Stochastic Methods for Sequential Data Assimilation in Strongly Nonlinear Systems

Dinh-Tuan Pham

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Stochastic methods for sequential data assimilation in strongly nonlinear systems

Dinh Tuan Pham

No 3597

December 30, 1998

THEME 4

ISSN 0249-6399

Stochastic methods for sequential data assimilation in strongly nonlinear systems

Dinh Tuan Pham

Thème 4 — Simulation et optimisation
de systèmes complexes
Projet Idopt

Rapport de recherche n ° 3597 — December 30, 1998 — 26 pages

Abstract: This paper considers several filtering methods of stochastic nature based on Monte-Carlo drawings, in view of the sequential data assimilation in non linear models. They include some known methods such as the particle filter and the ensemble Kalman filters and some other introduced by us: the second order particle filters and the singular evolutive interpolated filter. The aim is to study their behaviour in the simple non linear chaotic Lorenz system, in the hope of getting some insight on more complex models. It is seen that these filters perform satisfactory but our filters have the clear advantage in term of cost. 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). We have also introduced the use of the forgetting factor, which could enhance significantly the filter stability in this nonlinear context.

Key-words: Attractor. Nonlinear dynamical system. Data assimilation. Filtering. Lorenz model. Monte-Carlo method.

(Résumé : tsvp)

The author would like to thanks O. Talagrand and J. Verron for bringing his attention to this problem and with whom he has had many stimulating and fruitful discussions.

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN (France)
Téléphone : 04 76 61 52 00 - International: +33 4 76 61 52 00
Télécopie : 04 76 61 52 52 - International: +33 4 76 61 52 52

Résumé : Dans ce travail on considère quelques méthodes de filtrage de nature stochastique basées sur les tirages Monte-Carlo, pour l'assimilation séquentielle de données. Ceux-ci incluent les méthodes connues comme le filtrage particulaire et le filtrage d'ensemble de Kalman et d'autres méthodes introduites par nous: les filtres particulaires au second ordre et le filtre singulier évolutif interpolé de Kalman. Le but est d'étudier le comportement de ces filtres dans le système simple de Lorenz, dans l'espoir d'obtenir quelques idées de ce qui se produirai dans le cas de systèmes plus complexes. On voit que ces filtres ont des performances satisfaisantes, mais les nôtres ont l'avantage en terme de coût de calcul. Cela est réalisé grâce au concept de tirage exact au second ordre et à la correction sélective de erreurs selon les directions parallèles au sous espace tangent à l'attracteur (qui est de dimension faible). Nous avons également introduit l'utilisation du facteur oubli, qui peut renforcer de façon notable la stabilité des filtres, dans ce contexte non linéaire.

Mots-clé : Assimilation de données. Attracteur. Système dynamique non linéaire. Filtrage. Modèle de Lorenz. Méthode Monte-Carlo.

1 Introduction and motivation

The purpose of this work is to propose some sequential data assimilation methods and investigate their behavior and also that of some other known methods, 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, Evensen and van Leeuwen, 1994, Evensen, 1997, Burgers *et al.*, 1998) and some new filters which we introduce: the second order particle filters and the singular extended interpolated filter (Pham, 1996, 1997, Pham *et al.*, 1998a). All these methods are 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 cost factor is a big concern (since the state space has a very large number of dimensions, which could result in prohibitive computational cost). But in this paper we shall focus in the nonlinearity aspect and to 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 have some connection with atmospheric flows. More importantly, this system exhibits the main features of a nonlinear dynamics: the existence of an attractor and chaos. The first means that the system state in time will approach a special subset of the phase space called attractor and this happens regardless of the initial state (provided that it is not to far from the attractor). The second means that two nearby states, however close, will diverge in time, hence the system state in the far future is unpredictable (even that the system is deterministic).

Our works have been influenced by the paper of Miller *et al.* (1994) in which the data assimilation in the Lorenz system is investigated. However, these authors focus on the extended Kalman filters and the variationnal methods while we here focus on newer stochastic techniques. (However, we have been recently aware of the paper of Evensen, 1997, in which the author study his ensemble Kalman filter using a similar design). More importantly, unlike their paper we consider partial and not full observation. We feel that full observation of the state vector constitutes a too favorable situations, as we have been motivated by the application to oceanography and meteorology in which partial observation is the norm.

The paper is organized as follows. The next section introduces the sequential data assimilation framework and provide a brief review of the nonlinear optimal filter, the particle filter and the ensemble Kalman filter. Section 3 introduces some new filters and also the use of the forgetting factor and 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 a conclusion and an appendix describing the "second order exact sampling" which is the basis of our filter.

2 The particle filter and ensemble Kalman filter

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

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

where $\mathbf{x}^t(t)$ a vector representing the true system state at time t, $M(s,t)$ 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

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

where 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 the estimation of the state of the system at each observation time, using only the observations up to this time. In the linear case, this problem have been completely 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 for example Ghil and Manalotte-Rizzoli, 1991, for detail). However, this filter, beside being suboptimal, may produce instabilities of even divergence (Kushner, 1967, Evensen, 1992, Gauthier *et al.*, 1993, ...). The best way to deal with nonlinearities is, of course, to consider the optimal nonlinear filtering problem. Theoretical solution to this problem has been known but the practical implementation remains problematic, as it will be made clear below.

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 appropriately 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 which is the conditional density distribution function $d^a(\cdot|\ \mathbf{y}_1^o, \ldots, \mathbf{y}_k^o)$ of the state vector given all past observations up to time t_k. Similarly, instead of a forecast state vector at time t_k and its error covariance matrix, one now needs a *forecast* density, which is the conditional density distribution function $d^f(\cdot|\mathbf{y}_1^o, \ldots, \mathbf{y}_{k-1}^o)$ of the state vector given all past observations up to time t_{k-1}. The filtering can then be performed in two steps as in the linear case.

(i) 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 $\phi(\mathbf{z}|\ M(t_k, t_{k-1})\mathbf{x}, \mathbf{Q}_k)$ where

$$\phi(\mathbf{z}|\ \mathbf{m}, \boldsymbol{\Sigma}) = \frac{\exp[-\frac{1}{2}(\mathbf{z}-\mathbf{m})^T\boldsymbol{\Sigma}^{-1}(\mathbf{z}-\mathbf{m})]}{\sqrt{\det(2\pi\boldsymbol{\Sigma})}}$$

(^{T} denoting the transpose) is the density (at $\mathbf{z}$) of a Gaussian vector of mean $\mathbf{m}$ and covariance matrix $\mathbf{\Sigma}$. Thus,

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

(ii) The correction (or analysis) step: one computes the analysis density, which, from Bayes rule and the observation equation (2.2) assuming again Gaussian noise, is given by

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

The problem with the above formula is the needed integration with respect to the state vector. Furthermore the integration in (2.3) requires the evaluation of $M(t_k,t_{k-1})\mathbf{x}$ for a large number values of $\mathbf{x}$ while even one such evaluation is already quite costly in applications. The above filter is thus inapplicable in practice, but it can be used as starting point to construct suboptimal filters.

2.1 The particle filter

The basic idea is to approximate the analysis distribution at time t_{k-1} 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 $p_{1,k-1},\ldots p_{r+1,k-1}$, that is to approximate $d^a(\cdot|\mathbf{y}_1^o,\ldots,\mathbf{y}_{k-1}^o)$ by a convex combination of Dirac functions. Then in the case where there is no system noise, the function (2.3) is also a convex combination of Dirac functions and correspond to the discrete distribution based on the points $M(t_k,t_{k-1})\mathbf{x}_j^a(t_{k-1})$ with masses $p_{j,k-1}$, $j=1,\ldots,r+1$. However, if the system noise is present, the function (2.3) is again continuous. To obtain a discrete distribution, we shall resort to Monte-Carlo drawing. We draw $r+1$ realizations $\boldsymbol{\eta}_{1,k},\ldots,\boldsymbol{\eta}_{r+1,k}$ of a Gaussian random vector with zero mean and covariance matrix $\mathbf{Q}_k$. Then the distribution defined by the density (2.3) can be approximated by the discrete distribution located at

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

with the same probabilities $p_{1,k-1},\ldots p_{r+1,k-1}$. The new analysis distribution, defined by (2.4), then simply reduces to the discrete distribution based on the same points but having masses

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

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

(i) *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 set the probabilities $p_{1,0},\ \ldots,\ p_{r+1,0}$ to $1/(r+1)$. In practice, we recommend constituting a large data bases of states obtained from a long free run of the model, then just randomly pick $r+1$ particles out of this base.

(ii) *Forecast*: At time t_k, move the particles to $\mathbf{x}_j^f(t_k)$ defined by (2.5), keeping the same probabilities.

(iii) *Analysis*: Having observed $\mathbf{y}_k^o$, one corrects the probabilities according to (2.6) and set $\mathbf{x}_j^a$ to $\mathbf{x}_j^f$. The analysis state can be obtained as

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

and its error covariance matrix by

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

However, these computations are not necessary for the filter operation. The set of particles and its associated probabilities in fact provide a more complete picture about the state of the system. One can also compute the forecast vector and its error covariance matrix similarly, but again this is not necessary.

The above description corresponds to the simplest form of the particle filter. In practice it is often necessary to introduce a resampling step for it to work. The problem is that, the system being chaotic, the particles tend to become disperse and hence a lot of them would receive negligible probability (being far from the true state). Thus only few particles effectively participate in the filter operation. Resampling is performed to avoid this. Instead of setting $\mathbf{x}_j^a(t_k)$ to $\mathbf{x}_j^f(t_k)$ to, one redraw $r+1$ particles according to the analysis density (2.4) and reset the probabilities to $1/(r+1)$. There is a pitfall though. Since this density is a convex combination of Dirac functions, one actually draws $r+1$ particles out of a set of the same number of elements, so that one gets a lot of identical particles. If the system noise is present, this may not be very serious since they will be separated 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 so or be close so that one again effectively works with few particles. Therefore we propose to redraw the particle according *not to the discrete distribution* located at $\mathbf{x}_1^f(t_k)$, $\ldots$, $\mathbf{x}_{r+1}^f(t_k)$ with probabilities $p_{1,k}$, $\ldots$, $p_{r+1,k}$, *but to a continuous distribution* with a density. Using the popular kernel density estimation method (see for ex. Silverman, 1986), we propose to take as this density

$$\hat{d}^a(\mathbf{x}|\ \mathbf{y}_1^o, \ldots, y_k^o) = \sum_{j=1}^{r+1} p_{j,k} \phi[\mathbf{x} - \mathbf{x}_j^f(t_k)|\ \mathbf{0},\ h^2 \mathbf{P}^a(t_k)]$$

where $\mathbf{P}^a(t_k)$ is as in (2.7) and h is a tuning parameter. Then drawing $r+1$ particles according to this density is the same drawing $r+1$ elements 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 random perturbation $\boldsymbol{\xi}_{j,k}$, $(j = 1, \ldots, r+1)$, having zero means and covariance matrix $h^2\mathbf{P}^a(t_k)$

Note that resampling is only necessary if the set of probabilities $\{p_{1,k}, \ldots, p_{r+1,k}\}$ differ too much from the uniform probabilities. As measure of this discrepancy, we propose to take the entropy difference

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

This measure is always nonnegative and vanish only if and only if $p_{1,k} = \cdots = p_{r+1,k} = 1/(r+1)$. Thus, resampling is performed when it is greater than some threshold.

2.2 The ensemble Kalman filter

This filter has been introduced by Evensen (1994, 1997) and later modified by Burgers *et al.* (1998). It also makes use of a ensemble of states as in the particle filter. In fact the forecast steps are identical in both filters. But their analysis steps differ. In the first version of the ensemble Kalman filter, this step is the same as in the Kalman filter, except that the gain matrix is computed from the forecast error covariance matrix provided by the ensemble of states. In a later version (Burgers *et al.*, 1998), the analysis step is performed by "correcting" the ensemble states themselves and not the analysis state and is thus similar to our second order particle filter, described below. For each ensemble state $\mathbf{x}_j^f(t_k)$ (the notation is the same as in previous subsection), the correction is again performed as in the Kalman filter, but with the observation $\mathbf{y}_k$ being added a noise vector $\boldsymbol{\epsilon}_{j,k}$ having zero mean and covariance matrix $\mathbf{R}_k$.

Note that in this filer, as well as in all that follow, the probabilities associated with the ensemble states (or particles) do not change and hence equal $1/(r+1)$ where $r+1$ is the number of states. Thus resampling is never necessary. In fact it is implicit at the analysis stage through the addition of noise to the observation.

3 Some new filters and improvements

The particle method requires a very large number of particles in order to approximate adequately a continuous distribution by a discrete one, especially in a high dimensional state space. Monte-Carlo technique has notoriously slow convergence: 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 matrix contain a huge number of elements, so chance are that many of them are downright wrong.

The basic idea in our filters is to construct the particles in such a way to have sample mean and covariance matrix matching 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 since we also want our filter to perform well in linear models, it is natural to require the above condition.

3.1 The second order particle filter

Consider the forecast stage in the particle filter. If the particle $\mathbf{x}_1^a(t_{k-1}), \ldots, \mathbf{x}_{r+1}^a(t_{k-1})$ have 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 a linear model, the forecast distribution has mean

$$\mathbf{x}^f(t_k) = \frac{1}{r+1}\sum_{j=1}^{r+1}\mathbf{x}_j^f(t_k-), \qquad \mathbf{x}_j^f(t_k-) = M(t_k, t_{k-1})\mathbf{x}_j^a(t_{k-1}) \tag{3.1}$$

and covariance matrix

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

If the model is nonlinear, the above formula is only approximate, but one can do nothing about it except increasing r for better approximation for higher r. However, in all cases, the above matrix will differ from sample covariance matrix of the particles $\mathbf{x}_1^f(t_k), \ldots, \mathbf{x}_{r+1}^f(t_k)$, as defined by (2.5), because of sampling fluctuation in Monte-Carlo drawing.

To overcome the above problem, we introduce the concept of *second order exact sampling*. The sample $\boldsymbol{\eta}_{1,k}, \ldots, \boldsymbol{\eta}_{r+1,k}$ from a Gaussian distribution of zero mean and covariance matrix $\mathbf{Q}_k$ is said to be second order exact if it satisfies:

$$\frac{1}{r+1}\sum_{j=1}^{r+1}\boldsymbol{\eta}_{j,k} = \mathbf{0}, \qquad \frac{1}{r+1}\sum_{j=1}^{r+1}\boldsymbol{\eta}_{j,k}\boldsymbol{\eta}_{j,k}^T = \mathbf{Q}_k.$$

If one further introduces the linear constraints:

$$\sum_{j=1}^{r+1}\boldsymbol{\eta}_{j,k}[\mathbf{x}_j^f(t_k-) - \mathbf{x}^f(t_k)]^T = \mathbf{0} \tag{3.3}$$

then it can be verified that the particles $\mathbf{x}_1^f(t_k), \ldots, \mathbf{x}_{r+1}^f(t_k)$ have sample means $\mathbf{x}^f(t_k)$ and sample covariance matrix $\mathbf{P}_a^f(t_k)$, as defined in (3.2) and (3.3).

In the appendix we shall show how to perform a *second order exact sampling with linear constraints*. 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

$$\mathbf{C}(t_k-) = [\mathbf{x}_1^f(t_k-) - \mathbf{x}^f(t_k) \quad \cdots \quad \mathbf{x}_{r+1}^f(t_k-) - \mathbf{x}^f(t_k)] \tag{3.3'}$$

does not exceed r.

Turning to the analysis stage, we have adopted the correction of the particles themselves, as in Burger *et al.* (1998), and not of their probabilities, as in the particle filter. Note that if the analysis vector is corrected linearly with a gain matrix $\mathbf{K}_k$ as

$$\mathbf{x}^a(t_k) = \mathbf{x}^f(t_k) + \mathbf{K}_k[\mathbf{y}_k^o - H_k\mathbf{x}^f(t_k)], \tag{3.4}$$

then by linearizing the operator H_k as $H_k\mathbf{x}^f(t_k) = H_k\mathbf{x}^t(t_k) + \mathbf{H}_k[\mathbf{x}^f(t_k) - \mathbf{x}^t(t_k)]$, its error covariance matrix can be seen to be

$$\mathbf{P}^a(t_k) = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}^f(t_k)(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T. \tag{3.5}$$

Since the analysis vector should be the mean of the particles, one could correct the particles themselves by adding $\mathbf{K}_k[\mathbf{y}_k^o - H_k\mathbf{x}_j^f(t_k)]$ to $\mathbf{x}_j^f(t_k)$, $(j = 1, \ldots, r+1)$. But then the new particles would have sample covariance matrix the first term in the right hand side of (3.5) and not the whole right hand side. To correct this, we add a noise term. To be precise, we draw a second order exact sample $\tilde{\boldsymbol{\epsilon}}_{1,k}, \ldots, \tilde{\boldsymbol{\epsilon}}_{r+1,k}$ according to the Gaussian distribution with zero means and covariance matrix $\mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T$ with linear constraints:

$$\sum_{j=1}^{r+1} \tilde{\boldsymbol{\epsilon}}_{j,k}[\mathbf{x}_j^f(t_k) - \mathbf{x}^f(t_k)]^T = \mathbf{0} \tag{3.6}$$

and we correct the particles as

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

Then it can be check that the particles $\mathbf{x}_1^a(t_k), \ldots, \mathbf{x}_{r+1}^a(t_k)$ have sample mean $\mathbf{x}^a(t_k)$ and sample covariance matrix $\mathbf{P}^a(t_k)$, as given in (3.4) and (3.5).

Note that Burger *et al.* (1998) add noise directly to the observation and not to the particles as we do. But this is equivalent since adding a noise $\boldsymbol{\epsilon}_{j,k}$ to $\mathbf{y}_j^o$ amounts to adding $\mathbf{K}_k\boldsymbol{\epsilon}_{j,k}$ to the j-th particle and the last random vector has covariance matrix $\mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T$. The reason that we do our way is that the matrix $\mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T$ 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 that of the constraint matrix

$$\mathbf{C}(t_k) = [\mathbf{x}_1^f(t_k) - \mathbf{x}^f(t_k) \quad \cdots \quad \mathbf{x}_{r+1}^f(t_k) - \mathbf{x}^f(t_k)]^T \tag{3.6$'$}$$

does not exceed r.

The above arguments apply regardless the gain matrix one uses, but a natural choice is Kalman gain, given by $\mathbf{K}_k = \mathbf{P}^f(t_k)\mathbf{H}_k^T[\mathbf{H}_k\mathbf{P}^f(t_k)\mathbf{H}_k^T + \mathbf{R}_k]^{-1}$. This yields the best *linear* correction in the case where the operator H_k is linear. However, when H_k is non linear, we feel it is better to avoid linearization. Observed that the matrices $\mathbf{H}_k\mathbf{P}^f(t_k)\mathbf{H}_k^T$ and $\mathbf{P}^f(t_k)\mathbf{H}_k^T$, in the linear case, are no other than the sample covariance matrix of $\mathbf{y}_j^f(t_k) =$

$H_k \mathbf{x}_j^f(t_k)$, $j = 1, \ldots, r+1$ and sample 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

$$\mathbf{K}_k = \Big(\frac{1}{r+1}\sum_{j=1}^{r+1}[\mathbf{x}_j^f(t_k) - \mathbf{x}_j^f(t_k)][\mathbf{y}_j^f(t_k) - \mathbf{y}^f(t_k)]^T\Big) \Big(\frac{1}{r+1}\sum_{j=1}^{r+1}[\mathbf{y}_j^f(t_k) - \mathbf{y}_j^f(t_k)][\mathbf{y}_j^f(t_k) - \mathbf{y}^f(t_k)]^T + \mathbf{R}_k\Big)^{-1}. \quad (3.8)$$

3.2 The singular evolutive interpolated Kalman (SEIK) filter

We have developed this filter (Pham, 1996, 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 particle or ensemble filter using a *minimum number* of particles.

The principle of the SEEK filter is to make corrections only in the directions for which the error is amplified or not strongly attenuated by the system dynamic, relying on the above 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 a such 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 projects 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 form 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).

Turning to the second order particle filter, 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 particles $\mathbf{x}_1^a(t_k)$, $\ldots$, $\mathbf{x}_{r+1}^a(t_k)$. However, the above filter requires that r is not less than the rank of $\mathbf{Q}_k$ plus that of the matrix (3.3′) and the rank of $\mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T$ plus that of the matrix (3.6′) and to guarantee that such condition is satisfied one would need 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 particles as in (3.7), we simply redraw them second order exactly according to the Gaussian distribution with mean $\mathbf{x}^a(t_k)$, defined by (3.4) and covariance matrix $\mathbf{P}^a(t_k)$. This matrix has the same rank as $\mathbf{P}^f(t_k)$ (as will be seen later) 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 particles, we don't need $\mathbf{x}_1^f(t_k), \ldots, \mathbf{x}_{r+1}^f(t_k)$, except for the computation of $\mathbf{K}_k$ in (3.8). This is not a problem, since an alternative definition of $\mathbf{K}_k$ is possible and will be given shortly. Thus we don't have to draw $\boldsymbol{\eta}_{1,k}, \ldots, \boldsymbol{\eta}_{r+1,k}$.

As the covariance matrices involved in this filter all have rank r, it is much more efficient computationally to work with their factored form. Let $\mathbf{T}$ be a $(r+1) \times r$ full rank matrix with columns being orthogonal to the vector $[1 \ \cdots \ 1]^T$, then the right hand side of (3.2) can be written as $\mathbf{L}_k \mathbf{U}_{k-1} \mathbf{L}_k^T + \mathbf{Q}_k$ where

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

But, as state before, we change the system noise by projecting it onto the linear space spanned by the columns of $\mathbf{L}_k$. This yields $\mathbf{P}^f(t_k) = \mathbf{L}_k \tilde{\mathbf{U}}_{k-1} \mathbf{L}_k^T$ where

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

Note that the Kalman gain matrix is $\mathbf{P}^f(t_k)\mathbf{H}_k^T[\mathbf{H}_k^T\mathbf{P}^f(t_k)\mathbf{H}_k + \mathbf{R}_k]^{-1}$. This suggests taking as gain matrix, in the case where H_k in nonlinear, as

$$\mathbf{K}_k = \mathbf{L}_k\tilde{\mathbf{U}}_{k-1}(\mathbf{HL})_k^T[(\mathbf{HL})_k^T\tilde{\mathbf{U}}_{k-1}(\mathbf{HL})_k + \mathbf{R}_k]^{-1} \tag{3.9}$$

where $(\mathbf{HL})_k$ is the matrix $[H_k\mathbf{x}_1^f(t_k-) \ \cdots \ H_k\mathbf{x}_{r+1}^f(t_k-)]\mathbf{T}$.

Then, with a similar (and tedious) calculation as in the SEEK filter, one gets that that $\mathbf{P}^a(t_k) = \mathbf{L}_k\mathbf{U}_k\mathbf{L}_k^T$ where

$$\mathbf{U}_k = [\tilde{\mathbf{U}}_{k-1}^{-1} + (\mathbf{HL})_k^T\mathbf{R}_k^{-1}(\mathbf{HL})_k]^{-1}. \tag{3.10}$$

Hence $\mathbf{P}^a(t_k)$ has the same rank r as $\mathbf{P}^f(t_k)$, as mentioned earlier. Also $\mathbf{K}_k$ can be more conveniently computed as $\mathbf{L}_k\mathbf{U}_k(\mathbf{HL})_k^T\mathbf{R}_k^{-1}$.

A convenient choice for the matrix $\mathbf{T}$ is

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

In this case $\mathbf{L}_k$ reduces to 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.

It is worthwhile to relate the SEIK filter to the SEEK filter. Formula (3.1) and (3.2) together with the fact that the sample covariance matrix of $\mathbf{x}_1^a(t_{k-1}), \ldots, \mathbf{x}_1^a(t_{k-1})$ equal $\mathbf{P}^a(t_{k-1})$, is almost the same as 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 particles $\mathbf{x}_1^a(t_{k-1})$, $\ldots$, $\mathbf{x}_{r+1}^a(t_{k-1})$. Indeed, such interpolation would yield precisely the forecast and its error covariance matrix as given by (3.1) and (3.2). That is why the filter is called interpolated. The analysis step is the same as in the SEEK filter, excepted that the linearization of H_k is also avoided.

3.3 The singular second order particle filter

We have mentioned the difficulty of the second order particle filter that the number of particles should be greater than twice dimension of the phase space in order that the "rank conditions" can be guaranteed to hold. But although this space can be of very high dimension, the system state essentially lies on the attractor, which has a low dimension. Thus the "rank conditions" can still be approximately satisfied with a small number of particles.

The basic idea is to observe that the particles should lie near the attractor, hence 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 approximation through the singular value decomposition (SVD). One can, for example, write $\mathbf{C}(t_k-) = \mathbf{V}\mathbf{D}\mathbf{W}^T$ 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 $\mathbf{Q}_k$ 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, we just approximate $\mathbf{Q}_k$ by a matrix of rank $r' \leq r - r^*$. Similarly, we replace the constraint matrix $\mathbf{C}(t_k)$ by a rank r^* approximation $\tilde{\mathbf{C}}(t_k)$. If the matrix $\mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T$ 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 would approximate $\mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T$ by a matrix of rank $r'' \leq r - r^*$. Approximations to a covariance matrix may be obtained through an eigen-decomposition. For example, one can write $\mathbf{Q}_k = \mathbf{V}\mathbf{D}\mathbf{V}^T$ 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 the diagonal elements of $\mathbf{D}$ other than the r' largest to 0 yields a rank r' approximation to $\mathbf{Q}_k$.

The above method requires the choice of of three approximation ranks r^*, r' and r'' and a number of particles $r + 1 > r^* + \max(r', r'')$. 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 form formula (3.8). Hence r'' can be taken as r^* or less.

Since the gain matrix is of rank at most r, it is more efficiently computed as in section 3.2. Let $\mathbf{T}$ be the same matrix as defined there and put

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

Then $\mathbf{K}_k$ may be computed as in (3.8) but with $\mathbf{U}_{k-1}$ replaced $\tilde{\mathbf{U}}_{k-1}$ and $\mathbf{x}_j^f(t_k-)$ (in the definition of $(\mathbf{HL})_k$) replaced by $\mathbf{x}_j^f(t_k)$.

3.4 The use of the forgetting factor and an adaptative scale in the observation error covariance matrix

In our earlier studies (Pham *et al.*, 1998a, 1998b), we have found that the introduction of a forgetting factor can greatly improve the filter stability. This can be explained by the fact that in certain region of the phase space, the model can be strongly nonlinear and quite sensitive to the initial condition, hence relying too much on the model could lead to divergence (with respect to the true state). Thus, it is prudent to rely more on the observation in constructing the analysis vector, than as one do in the linear case. A simple way to achieve this is to use a gain matrix *computed as if* the observation noise has a smaller covariance matrix. More precisely, we replace, in the formula (3.8) and (3.9), $\mathbf{R}_k$ by $\rho\mathbf{R}_k$ where ρ is a positive coefficient not greater than 1. Note that replacing $\mathbf{R}_k$ in (3.9) would yield the same result as replacing $\tilde{\mathbf{U}}_{k-1}$ by $\tilde{\mathbf{U}}_{k-1}/\rho$. Therefore, in the case of the SEIK filter, the gain matrix can still be computed as $\mathbf{L}_k\mathbf{U}_k(\mathbf{HL})_k\mathbf{R}_k^{-1}$ but with $\mathbf{U}_k$ now given by $[\rho\tilde{\mathbf{U}}_{k-1}^{-1} + (\mathbf{HL})_k\mathbf{R}_k^{-1}(\mathbf{HL})_k^T]^{-1}$ and not (3.10). But here we only change the way the gain matrix is computed and not the forecast error covariance matrix $\mathbf{P}^f(t_k)$, as we have done in earlier works (Pham *et al.*, 1998a, 1998b). Thus the analysis error covariance matrix will be computed through the formula (3.5) and this yields, after some tedious algebra:

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

r where $\mathbf{U}_k$ is as above. We feel that the present approach is more satisfactory theoretically than the earlier one, since we only change the gain matrix is computed and nothing else. (But in our simulations their performance are quite similar.) Incidentally, in the case of the (singular) second order particle filter, the $\boldsymbol{\epsilon}_{j,k}$ are still be drawn according to the covariance matrix $\mathbf{R}_k$. We argue that is 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 which 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 toward the observation.

There is a difficulty with the filters considered in this paper when the small is noise is too small. To simplify things, consider the case where there is no system noise ($\mathbf{Q}_k = \mathbf{0}$) and the observation noise is of the form $\mathbf{R}_k = \sigma^2\tilde{\mathbf{R}}_k$ 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, the matrix $\mathbf{U}_k$ in (3.10) would reduce to $\sigma^2[(\mathbf{HL})_k^T\tilde{\mathbf{R}}_k^{-1}(\mathbf{HL})_k]^{-1}$, provided that the later is invertible. If it is not, one can still show that after several analysis step, the matrix $\mathbf{U}_k$ will become of the same order as σ^2 and hence so is $\mathbf{P}^a(t_k)$. The behavior of the second order particle and the ensemble filter are more subtle. But is can be seen form the formula (3.7) that, without the term $\tilde{\boldsymbol{\epsilon}}_{i,k}$, the set of particles will shrink to a single one. Thus in all case we will have an almost zero analysis error covariance matrix, but the analyis 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 they are computed, only reflects the error coming from the system and observation noises, but other kinds of error have been ignored. This includes the discrete approximation and the addition of noise in the resampling scheme in the particle filter, the sampling fluctuation in Monte-Carlo sampling, the linearization or interpolation error. When the system and observation noise are very small, the above kinds of error can no longer be neglected and the analysis error covariance matrix, as computed by the Kalman filter and its variates, can seriously underestimate the true filter error.

Note that the Kalman gain is not really affected by the small noise problem. As argued above, the matrix $\mathbf{U}_k$ and hence the analysis and forecast error covariance matrix, in the long run, will become of the same order as the scale factor σ^2 in the noise covariance matrix. Thus, looking at the formula (3.9) for the gain matrix, one see that σ^2 cancel out. It follows that 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 would be too low. However, the filters considered in this paper are based on a set of particles which should have this matrix as covariance matrix, hence these particles would cluster too closely around their means. As argued before, in the limiting case of no noise, the set of particles will shrink to a single one.

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

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

where $\mathbf{e_k} = \mathbf{y}_k^o - H_k \mathbf{x}^f(t_k)$ is the innovation factor and λ is a forgetting factor. Actually $\hat{\sigma}_k^2$, obtained this way, would somewhat overestimate the scale factor σ^2 in the noise covariance matrix, 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

$$dx/dt = s(y-x), \qquad dy/dt = (r-z)x - y, \qquad dz/dt = xy - bz.$$

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 mathod, with an integration step 0.005. Observations are made at interval 0.05 apart, on the x-coordinate only and with an observation error variance 2. This design is similar to that of Miller *et al.* (199?), except that observation are made more frequently, to compensate for the fact that only x-coordinate and not the full state vector is observed. The filter performance will be measured by the root mean square error (RMSE), that is $1/\sqrt{3}$ times the norm of the difference between the analysis and the true states.

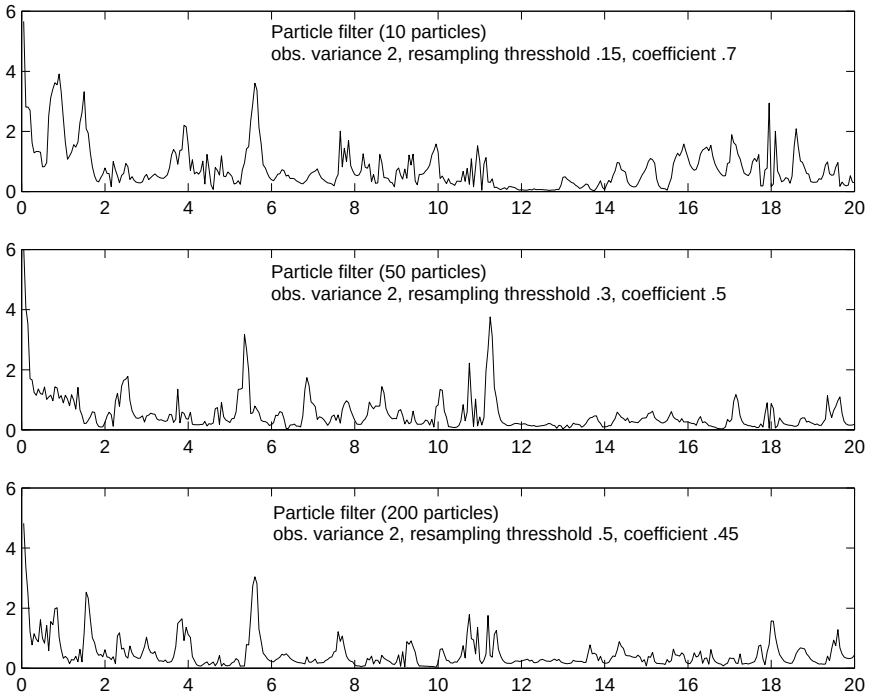

Figure 1: RMSE of the particle filter with 10, 50 and 200 particles

For the particle, the second order particle and the ensemble filters, we construct a data base of states as follows. We generate a trajectory of the Lorenz model starting at $(x, y, z) = (-0.587276, -0.563678, 16.8708)$ yielding 500 states (excluding the initial one) at interval 0.05 apart. We then retain only the last 400 states to avoid the transitory phase. The initial particles in the above filters are simply drawn randomly from this data base. The same data base 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 data base). For more detail on this EOF method, see for ex. Pham *et al.* (1998b). Only two first EOF are retained, yielding a rank two matrix. This is in line with the fact that the Lorenz attractor has 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 2.1) have been chosen to be .15 and .7, .3 and .5 and .5 and .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 what is reported here. Likewise better performance could be achieved with some other choices, since we haven't

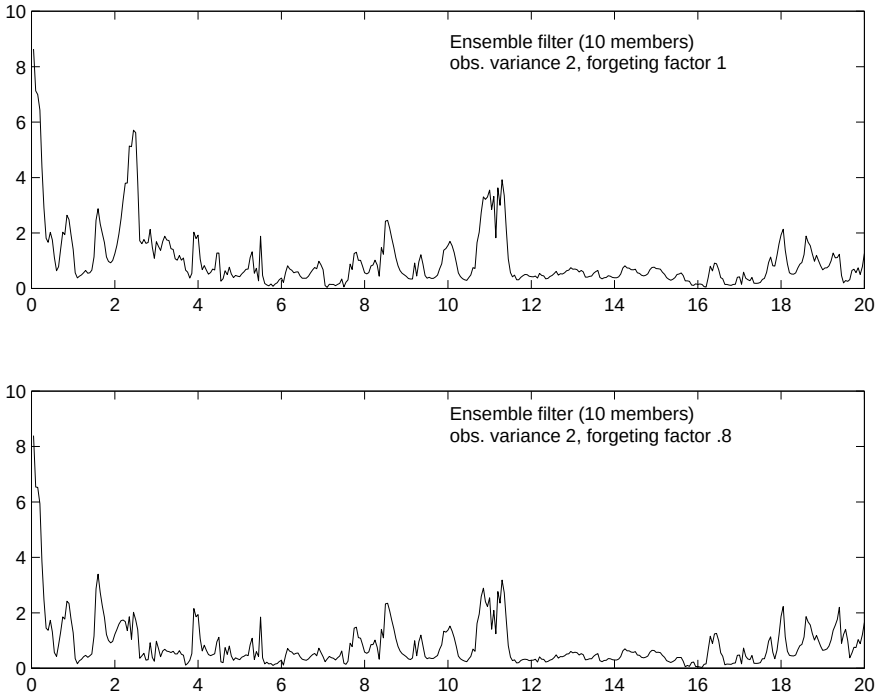

Figure 2: RMSE of the ensemble filter with and without forgetting factor

tried very hard to find out 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, but it also seems to be 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 ensemble filter with no forgetting factor and with a forgetting factor .8. One can see that the forgetting factor improves markedly the filter performance. As it can be seen in this plot and also in previous and subsequent plot, the filter sometime seems to lost track of the true system state, causing a sudden burst of large error (we suspect this happens when the system state enter a strong nonlinearity region). The use forgetting factor help to reduce the peak of these bursts and may even prevent them from happening. In this sense sense it enhance the filter stability.

Figure 3 plots the RMSE of the ensemble filter 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

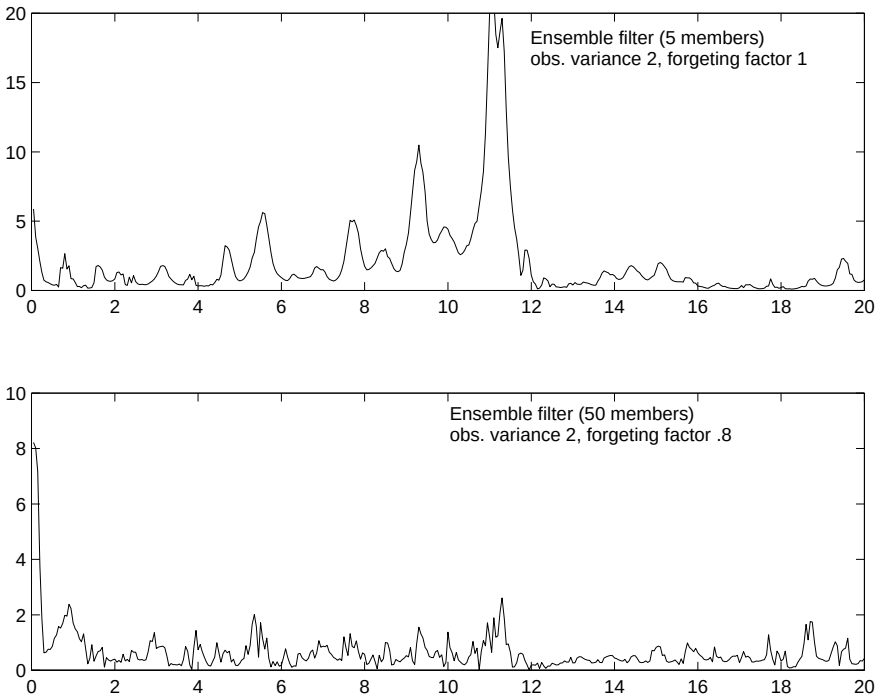

Figure 3: RMSE of the ensemble filter with 5 and 50 ensemble members

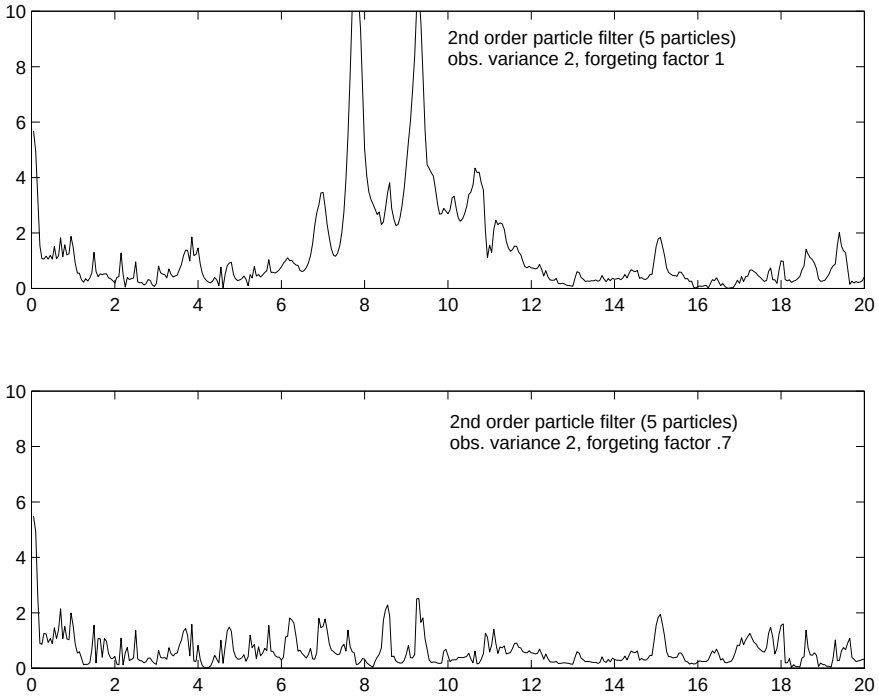

Figure 4: RMSE of the second order particle filter with and without forgetting factor

perform well with 5 ensemble members. Using a forgetting factor doesn't help. In fact the filter performs somewhat worse with a forgetting .9 or .8, that is why we chose to show the result with no forgetting factor. Not that the ensemble Kalman filter (the first version) has also been studied by Evensen (1997) in the context of the Lorenz model, and he has noted that good resuslt can be obtained using down to about 10 ensemble members. However, simulation results in Evensen (1997) cannot really be compared with ours because he use full observations (but with a larger time interval between them) and not partrial observations and besides we use the analysis scheme as in Burger *al.* (1998).

Figure 4 plots the RMSE of the second order filter with and without forgetting factor. Here again the effect of this factor is drastic, as it prevents too large filter error to happen around the middle of the experiment. Figure 5 plots the RMSE of this filter with different numbers of particles, 4 and 10 respectively, to show the effect of this number. Curiously, increasing the number of particles from 5 to 10 doesn't seem to increase filter performance. Its performance at 10 particles seems even slightly worse. With more particles, this filter behaves quite similar to the ensemble filter, as expected, since the condition that the drawing is second order exact and the constraints implemented by the second order particle filter would then be approximately satisfied by the ensemble filter anyway. But the performance

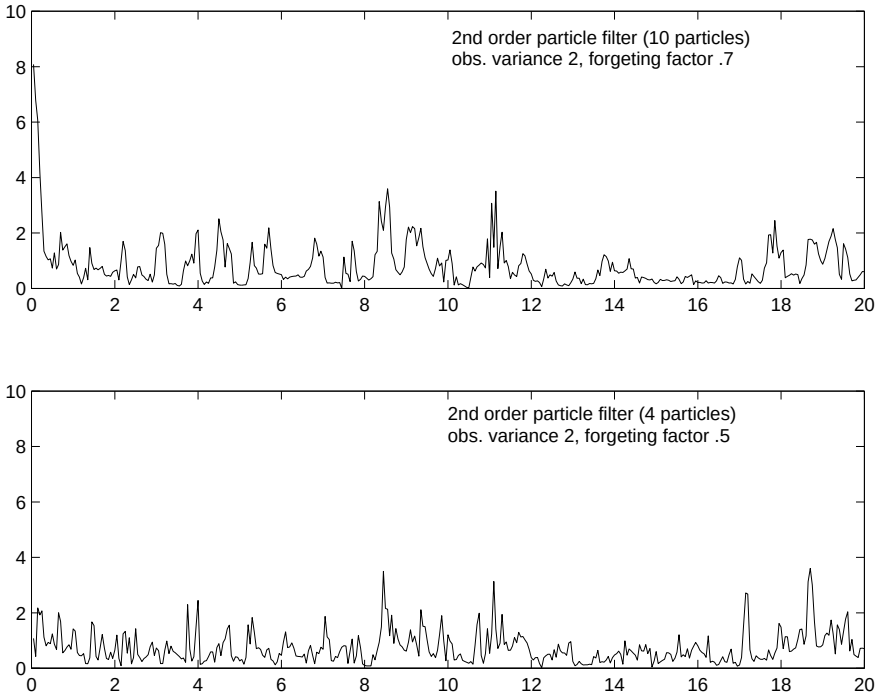

Figure 5: RMSE of the second order particle filter with and without forgetting factor

of both filters at 50 particles, for example, is still about the same as that of the second order particles filter with only 5 particles.

We don't have a sure explanation of the above finding. A plausible one is that the second order particle filter at 5 particles has almost attained it maximum capability. Indeed, this filter, as well as the ensemble Kalman filter, correct the analysis state in a *linear way* based on a gain matrix computed from the second order statistics only. The use of a more particles may improve the accuracy in evaluating these statistics, but the filter still hasn't exploit any other nonlinear characteristics (higher order moments, shape of the analysis density, ...). By contrast the particle filter theoretically can attain the optimal performance when the number of particles goes to infinity. Note that the "rank condition" in the second order particle filter requires that $r \geq 4$, that is one needs at least 5 particles. 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 suggests that one doesn't need a number of the particle more than the strict minimum. Note that the second plot of figure 5 (with 4 particles) corresponds actually to the singular second order particle filter, in which the constraint matrix has been reduced to a rank two matrix (in line with the fact that the Lorenz attractor has dimension about 2). The performance of this filter is somewhat worse than the one with 5 particles, but not much.

Finally, figure 6 shows that 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 compare very favorably to the second order particle filter. It is almost as good as the later with 5 particles and is even better to this filter with 4 particles. Yet, it requires only 3 particles.

5 Conclusion and discussion

We have introduced some new filter of stochastic nature for data assimilation in non linear systems. They have in common with previously introduced filters, such as the particle filter and the ensemble Kalman filter, in the use Monte-Carlo drawings to mimic the behavior of the system. Our contributions consists in three points

(i) The concept of second order exact sampling with linear constraints. This permits to reduce the number of particles to a strict minimum and yet not causing any degradation of performance.

(ii) The exploitation of the (existence of) low dimensional attractor of the system. This permits to reduce further the number of particles, 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 particles may be reduced to just the dimension of the attractor plus 1. Again, this does not degrade performance.

(iii) The use of the forgetting factor to enhance filter stability. Our simulation studies shows that filter instability is the most vexing problem in nonlinear data assimilation. The

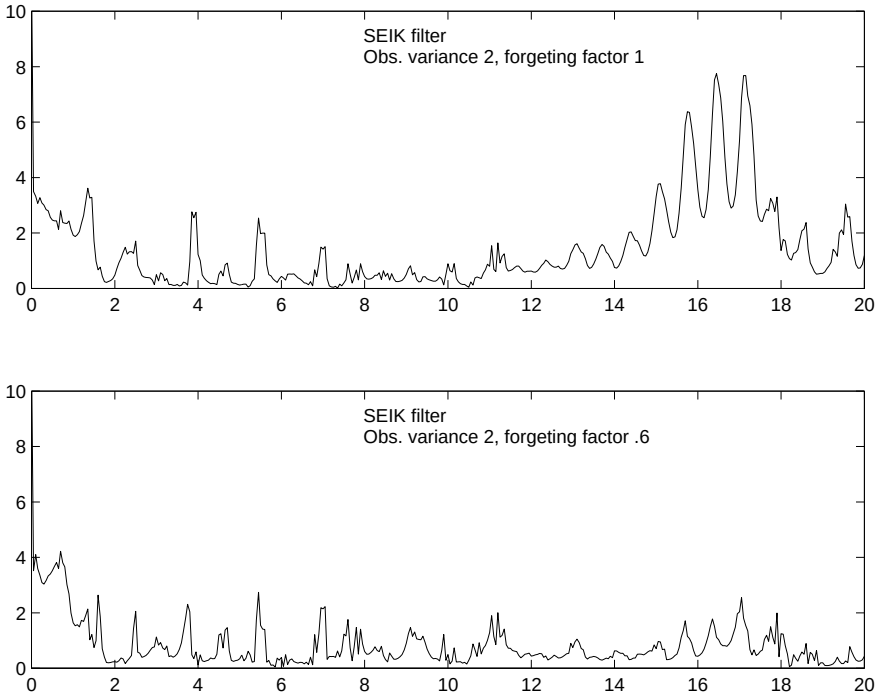

Figure 6: RMSE of the SEIK filter with and without forgetting factor

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 its magnitude and duration.

In term of performance, our new filters are comparable and comparable to the ensemble Kalman filter, all with appropriate forgetting factor. The advantage the new filters is cost. In this respect the SEIK filter is the best. The particle filter, however, can outperform these filters (and also the ensemble Kalman filter), 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.

6 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

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

$$[(\mathbf{x}_1-\mathbf{m}) \quad \cdots \quad (\mathbf{x}_{r+1}-\mathbf{m})]\mathbf{\Gamma} = \mathbf{0} \tag{A.2}$$

where $\mathbf{C}$ is a given matrix of $r+1$ rows.

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 (A1) and the constraint (A2) can be grouped into

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

Let r' be the rank of $\mathbf{\Sigma}$, then one may decompose it into the form $\mathbf{L}\mathbf{L}^T$ where $\mathbf{L}$ is a full rank matrix having r' columns. Since $\mathbf{x}_1, \ldots, \mathbf{x}_{r+1}$ are realizations of a random vector $\mathbf{x}$ having zero mean and covariance matrix $\mathbf{\Sigma}$, they must lie on the linear space spanned by the column of $\mathbf{L}$. Indeed, any vector orthogonal to the columns of $\mathbf{L}$ must be also orthogonal to the $\mathbf{x}_j$, since its scalar product with $\mathbf{x}$ has zero mean and variance. Thus, $\mathbf{x}$ may be written in the form $\mathbf{L}\mathbf{y}$ where $\mathbf{y}$ is a certain random vector of zero mean and unit covariance matrix, since it equals $(\mathbf{L}\mathbf{L}^T)^{-1}\mathbf{L}^T\mathbf{x}$, it is clear that $\mathbf{y}$ has zero mean and unit covariance matrix.

Thus, it suffices to draw a sample $\mathbf{y}_1, \ldots, \mathbf{y}_{r+1}$ conditionally from the Gaussian distribution of zero mean and unit covariance matrix, subject to the constraints that

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

Then the desired sample is simply $\mathbf{L}\mathbf{y}_1, \ldots \mathbf{L}\mathbf{y}_{r+1}$.

The first constraint in (A.4) means that the rows vectors of $[\mathbf{y}_1 \quad \cdots \quad \mathbf{y}_{r+1}]$ belong to the linear space of all vectors in $\mathbb{R}^{r+1}$ 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

$$\begin{bmatrix} \mathbf{y}_1^T \\ \vdots \\ \mathbf{y}_{r+1}^T \end{bmatrix} = [\mathbf{v}_1 \quad \cdots \quad \mathbf{v}_{r-r''}] \begin{bmatrix} \mathbf{v}_1^T \\ \vdots \\ \mathbf{v}_{r-r''}^T \end{bmatrix} \begin{bmatrix} \mathbf{y}_1^T \\ \vdots \\ \mathbf{y}_{r+1}^T \end{bmatrix} = [\mathbf{v}_1 \quad \cdots \quad \mathbf{v}_{r-r''}]\mathbf{W}$$

Without the constraint (A.4), the elements of the matrix $[\mathbf{y}_1 \quad \cdots \quad \mathbf{y}_{r+1}]$ are simply independent standard normal variables, hence, the columns vectors of the matrix $\mathbf{W} = [\mathbf{v}_1 \quad \cdots \quad \mathbf{v}_{r-r''}]^T[\mathbf{y}_1 \quad \cdots \quad \mathbf{y}_{r+1}]^T$ are independent random Gaussian vectors in $\mathbb{R}^{r-r''}$ with zero mean and unit covariance matrix. The first constraint in (A.4) simply impose that the matrix $[\mathbf{y}_1 \quad \cdots \quad \mathbf{y}_{r+1}]^T$ must equal $[\mathbf{v}_1 \quad \cdots \quad \mathbf{v}_{r-r''}]\mathbf{W}$, 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 $\mathbf{\Omega} = \mathbf{W}/(r+1)$ satisfies $\mathbf{\Omega}^T\mathbf{\Omega} = \mathbf{I}$, 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 an *uniform* random orthogonal matrix, which we define as a random matrix obeying the conditional Gaussian distribution with elements being independent standard normal variables subjected to the constraint that its columns are of unit norm and orthogonal among themselves. The distribution of such matrix is called uniform because it is invariant with respect to orthogonal transformation: if $\mathbf{\Omega}$ is an uniform random orthogonal matrix then so is $\mathbf{\Omega}\mathbf{O}$, for any orthogonal matrix $\mathbf{O}$.

We now propose an efficient method to draw an uniform random orthogonal matrix. Let $\mathbf{\Omega}$ be a such a matrix, of size $(r - r'') \times r'$. Denote by $\mathbf{z}_{r-r''}$ its last column. This vector is uniformly distributed over the unit sphere of $\mathbb{R}^{r-r''}$. Construct a $(r-r'') \times (r-r''-1)$ matrix $\mathcal{H}(\mathbf{z}_{r-r''})$ which completes it to form an orthonormal matrix. This means that the columns of $\mathcal{H}(\mathbf{z}_{r-r''})$ form a basis of the orthogonal complement to the linear subspace spanned by $\mathbf{z}_{r-r''}$. Thus one can write $\mathbf{\Omega}$ as $[\mathcal{H}(\mathbf{z}_{r-r''})\mathbf{\Omega}_{r-r''-1} \quad \mathbf{z}_{r-r''}]$ where $\mathbf{\Omega}_{r-r''-1} = \mathcal{H}(\mathbf{z}_{r-r''})^T\mathbf{\Omega}$ 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 $\mathcal{H}(\mathbf{z}_{r-r''-1})$ be a $(r - r'' - 1) \times (r - r'' - 2)$ matrix which completes it to form an orthonormal matrix. Then again $\mathbf{\Omega}_{r-r''-1} = [\mathcal{H}(\mathbf{z}_{r-r''-1})\mathbf{\Omega}_{r-r''-2} \quad \mathbf{z}_{r-r''-1}]$ where $\mathbf{\Omega}_{r-r''-2}$ is an $(r - r'' - 2) \times (r' - 2)$ uniform random orthogonal matrix. Continue this way, one gets

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

where $\mathbf{z}_k$ is a uniform random vector on the unit sphere of $\mathbb{R}^k$, $\mathcal{H}(\mathbf{z}_k)$ a $k \times (k-1)$ matrix which 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 1/2 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 $\mathbf{\Omega}_{r-r''}$.

The above matrix $\mathcal{H}(\cdot)$ can also be used to generate an orthonormal basis of the linear space of all vectors in $\mathbb{R}^{r+1}$ orthogonal to the column of $\mathbf{\Gamma}$. 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 $\mathcal{H}(\mathbf{z}_{r+1}) \cdots \mathcal{H}(\mathbf{z}_{r+1-i})$ form a basis of the linear space of all vectors in $\mathbb{R}^{r+1}$ 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 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 the vector for any vector $\mathbf{c}_{i+1}$,

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

is no other than its orthogonal projection onto the linear space of all vectors in $\mathbb{R}^{r+1}$ orthogonal to $\mathbf{c}_0$, $\ldots$, $\mathbf{c}_i$. Choose $\mathbf{c}_{i+1}$ to be a column of $\mathbf{\Gamma}$ such that the above projection is not zero (which is clearly possible if $i < r''$), so that one can normalized the vector

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

to get a vector $\mathbf{z}_{r-i}$ of unit norm. Then since $[\mathcal{H}(\mathbf{z}_{r-i}) \;\; \mathbf{z}_{r-i}]$ is an orthonormal matrix, the columns of $[\mathcal{H}(\mathbf{z}_{r+1}) \cdots \mathcal{H}(\mathbf{z}_{r+1-i})][\mathcal{H}(\mathbf{z}_{r-i}) \;\; \mathbf{z}_{r-i}]$ clearly also form an orthogonal basis of the linear space of all vectors in $\mathbb{R}^{r+1}$ orthogonal to $\mathbf{c}_0$, $\ldots$, $\mathbf{c}_i$. But we already know that the last column is proportional to the orthogonal projection of $\mathbf{c}_{i+1}$ onto the linear space of all vectors in $\mathbb{R}^{r+1}$ orthogonal to $\mathbf{c}_0$, $\ldots$, $\mathbf{c}_i$, therefore the other columns, that are those of $\mathcal{H}(\mathbf{z}_{r+1}) \cdots \mathcal{H}(\mathbf{z}_{r-i})$, form an orthogonal basis of the linear space of all vectors in $\mathbb{R}^{r+1}$ orthogonal to $\mathbf{c}_0$, $\ldots$, $\mathbf{c}_{i+1}$. This shows that the vector $\mathbf{z}_{r-i}$ satisfies the requirement. One continues this way until $i = r''$, at this stage all columns of $\mathbf{\Gamma}$ belongs to the linear space spanned by $\mathbf{c}_0$, $\ldots$, $\mathbf{c}_{r''}$.

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

(i) *Initialization*: Pick a non zero 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.

(ii) *Recursion:* for $i = 0, \ldots, r'' - 1$, pick a non zero column of the matrix $\mathcal{H}^T(\mathbf{z}_{r+1-i})\mathbf{\Gamma}^{(i)}$ 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 $\mathcal{H}(\mathbf{z}_{r+1}) \cdots \mathcal{H}(\mathbf{z}_{r+1-r''})\mathbf{\Omega}$. Recall that $\mathbf{\Omega} = \mathbf{\Omega}_{r-r''}$, hence the last matrix 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 $\mathcal{H}(\mathbf{z}_k)$. 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

$$\mathbf{I} - \frac{1}{1 + s z_{k,k}} \begin{bmatrix} z_{1,k} \\ \vdots \\ z_{k-1,k} \\ z_{k,k} + s \end{bmatrix} \begin{bmatrix} z_{1,k} & \cdots & z_{k-1,k} & z_{k,k} + s \end{bmatrix},$$

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 last column $-s\mathbf{z}_k$. Thus one may take as $\mathcal{H}(\mathbf{z}_k)$ 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., van Leuuven, P. J. and Evensen, G. (1998) Analysis scheme in the Ensemble Kalman filter. *Mon. Wea. Rev.*. **126**, 1719–1724.

Del Moral, P. Noyer, J. C., Rigal, G. and Salut, G (1995) Résolution particulaire et traitement non-linéaire du signal: application Radar/Sonar. (Particle resolution and non-linear signal processing with RADAR/SONAR applications) *Traitement du Signal*, 12, 4, 287–301

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

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.

Evensen G. (1997) Advanced data assimilation in strongly nonlinear dynamics. *Mon. Wea. Rev.*, **125**, 1342–1354.

Evensen G., van Leeuwen, P. J. (1994) Assimilation of Geostat Altimeter Data for the Agulhas Current using the Ensemble Kalmal Filter with a Quasi-Geostrophic Model. *Mon. Wea. Rev.*. **124**, 85–96.

Gauthier, P., Courtier P. and Moll, P. (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*, 23, 141-265.

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

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

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

Ide K., A. F. Bennett, P. Courtier, M. Ghil and A.C. Lorenc (1995) Unified notation for data assimilation: operational, sequential and variational. Submitted to the *J. Met. Soc. Japan*.

Pham D. T. (1996) A Singular Evolutive Interpolated Kalman Filter for Data Assimilation in Oceanography. Technical Report, Project Idopt INRIA-CNRS, RT 163, September 1996.

Pham, D.T (1997) Dimension, Predictability and Reduced Rank Kalman Filtering in Data Assimilation. Proceeding of the third bilateral French-Russian conference: Predictability of Atmospheric and Oceanic Circulations. Nancy, 9-11 April (1997)

Pham, D.T., Verron, J., Gourdeau, L. (1998a) Filtres de Kalman singuliers évolutif pour l'assimilation de données en océnographie. *Comptes Rendus Aca. Sci.* Science de la terre et des planètres, **326**, 255–260

Pham D. T., J. Verron, J. and Roubaud, M. C. (1998b): A Singular Evolutive Extended Kalman Filter for Data Assimilation in Oceanography. **16**, 3, 4, 323–340.

Silverman, B. W. (1986) *Density Estimation for Statistics and Data Analysis.* London: Chapman & Hall

Unit´e de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LES NANCY
Unit´e de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unit´e de recherche INRIA Rhône-Alpes, 655, avenue de l' Europe, 38330 MONTBONNOT ST MARTIN
Unit´e de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unit´e de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399