Algorithm xxx: A Discrete Time Kalman Filter Package for Large Scale Problems

GERMÁN A. TORRES
Facultad de Matemática, Astronomía y Física - CIEM (CONICET)
Universidad Nacional de Córdoba

Data assimilation is the process of feeding a partially unknown prediction model with available information coming from observations, with the objective of correcting and improving the modeled results. One of the most important mathematical tools to perform data assimilation is the Kalman filter. The Kalman filter is esentially an algorithm of prediction-correction type that is optimal in the sense of minimizing the trace of the covariance matrix of the errors. Unfortunately the computational cost of applying the filter to large scale problems is enormous, and the programming of the filter is highly dependent on the model and the format of the data involved. The first objective of this paper is to present a set of Fortran 90 modules in order to implement reduced rank square root versions of the Kalman Filter, adapted for assimilation of a very big amount of variables. The second objective is to present a Kalman filter implementation whose code can be independent from model and observations, and as easy as possible for the user. A detailed description of the algorithms, structure, parallelization and examples of use are given.

Categories and Subject Descriptors: G.3 [**Mathematics of Computing**]: Probability and Statistics—*Time Series Analysis*; G.4 [**Mathematics of Computing**]: Mathematical Software—*Algorithm design and analysis*; J.2 [**Computer Applications**]: Physical Sciences and Engineering—*Earth and atmospheric sciences*

General Terms: Algorithms, Performance

Additional Key Words and Phrases: large scale problems, data assimilation, Kalman filter

1. INTRODUCTION.

The Kalman filter is a set of mathematical equations that combine information coming from a model output and observations, producing a better estimation (or analysis) of the dynamical system. Essentially, it implements a predictor-corrector type estimator that is optimal in the sense that it minimizes the estimated error covariance under some hypotheses. The state and the covariance matrix of forecast

---

Author's postal address: Facultad de Matemática, Astronomía y Física, Medina Allende s/n, Ciudad Universitaria, 5000, Córdoba, Argentina.

This work was carried out with the aid of a grant from the Inter-American Institute for Global Change Research (IAI) CRN II 2017 which is supported by the US National Science Foundation (Grant GEO-0452325) and with funds from the Secretaría de Ciencia, Tecnología e Innovación Productiva - SECYT (Grant A06/STICAMSUD/02).

Author's electronic address: torres@famaf.unc.edu.ar

errors are predicted (using the model), and when there are observations, a correction step is performed (see [Welch and Bishop 2001]).

The filter is named after Rudolph E. Kalman, who in 1960 published his paper describing a recursive solution to the discrete-data linear filtering problem [Kalman 1960]. The Kalman filter has been extensively applied in motion prediction (see [Azuma and Bishop 1994]), parameter estimation (see [Charalambous and Hibey 2001] and [Annan et al. 2005]), navigation and global positioning systems (see [Kim and Iltis 2002]), improvement of species concentrations in chemical transport models (see [Zhang et al. 1997], [Zhang et al. 1999], [van Loon et al. 2000], [Flemming et al. 2001], [Serafy et al. 2002] and [Segers 2002]), image treatment (see [Ertürk 2002] and [Kuo et al. 2002]), improvement of oceanographic models (see [Allen et al. 2002] and [Hoteit et al. 2004]), etc.

The size of the problem to face can be a restriction to the application of the Kalman filter. For example, in a chemistry transport model we could assimilate 20 species concentrations in a grid of $50 \times 50 \times 20$ cells, that is, the state vector would have $n = 10^6$ entries, so we should have a computer able to manage matrices of size $n \times n$, and operate with them. The most expensive part of the Kalman algorithm is the prediction of the covariance matrix of forecast errors, where we have to apply the tangent linear model $2n$ times. In complex models, a time step could take a few seconds, therefore a Kalman filter step is impossible to implement, and new techniques have been developed for large scale problems like reduced rank square root methods and ensemble methods based in Monte-Carlo estimations (see [Brown and Gaston 1995], [Chin et al. 1995], [Pham et al. 1998], [Segers et al. 2000], [Hoteit et al. 2001], [Hoteit et al. 2002], [Evensen 2003], [Asif 2004], [Hoteit and Pham 2004], [Treebushny and Madsen 2005], [Chen et al. 2005] and [Hanea et al. 2005]). According to the application, covariance matrices may have a sparse structure, which could simplify some array operations and speed up execution (for example, for observations the covariance matrices are usually taken diagonal). The user must decide whether or not to use reduced rank methods according to the complexity of the model.

Another common problem is related to implementation difficulties. Every time the filter is applied, the model needs to be called many times. Therefore the model source code has to be rearranged. Noise has to be added to generate samples. An interface is needed between the incoming observations and the data produced by the model. After the assimilation step the model has to begin with the produced analysis, and the complete implementation is full of subtle things related with the model and the observations. The major obstacle is that models are like black boxes, where the user changes configuration files, not the code itself. The source code of the model should not be changed unless the user knows exactly what is being done.

This paper proposes a modular assimilation environment in Fortran 90. It means that observations, model and assimilation are separated one from another. For example, if the observation stations change the location, there is no need to modify the main code, but only a module related with them. If we need to change the model, there is no need to transform all the code, but only the module related with the model. If we want to change the Kalman filter version, a change in the module related with the assimilation will suffice. This will allow the user to make minimum changes in all the codes around a modelling system (model, data formats, libraries, configuration files) in order to avoid programmer bugs. The choice of the language was made because a big number of large scale models are written either in Fortran 77 or in Fortran 90 (even when it is possible to mix languages, some users prefer not to do it). The modules can handle single and double precision and use the BLAS/LAPACK libraries for the matrix operations. For the case of choosing parallelization the libraries BLACS, SCALAPACK and MPI are used.

This paper also proposes the implementation of a reduced rank square root ensemble Kalman filter taking advantage of the Fortran language for the matrix-vector manipulation needed specially in 2D or 3D models. An example of a 3D system is presented. Comparisons of the model solution against the true and assimilated solution are given. Notice that the assimilation modules are prepared to include other versions of the filter, like the complete extended Kalman filter and the ensemble Kalman filter (useful for small and medium scale problems), and the RRSQRT Kalman filter (useful for large scale problems) as it will be shown later.

There are some available filtering packages that already implement the Kalman filter. Some of them are:

(1) STSA (The Time Series Analysis Toolbox for O-Matrix): this toolbox is a collection of O-Matrix functions for performing time series and statistics related analysis and visualization. It has capabilities for ARMA and ARFIMA, Bayesian, non-linear and spectral analysis related models. Time series filtering functions and spectral analysis functions are provided. Random number generators are included for both time series, and general statistical analysis. It is a commercial package using the O-Matrix language. See [STSA].

(2) BFL (Bayesian Filtering Library): this library provides an application independent framework for inference in Dynamic Bayesian Networks, that is, recursive information processing and estimation algorithms based on Bayes's rule, such as Extended Kalman filters, particle filter, etc. It is written in C++. For details see [BFL].

(3) KALMTOOL: it is a set of MATLAB tools for state estimation for nonlinear systems. The toolbox contains functions for Extended Kalman filtering as well as for two new filters called the DD1 filter and the DD2 filter. The toolbox specifically addresses the problem of not having observations available at all sampling instants. See [KALMTOOL].

(4) Bayes++ Bayesian Filter Classes: Bayes++ is an open source library of C++ classes. These classes represent and implement a wide variety of numerical algorithms for Bayesian Filtering of discrete systems. The classes provide tested and consistent numerical methods and the class hierarchy explicitly represents the variety of filtering algorithms and system model types. See [BAYES++].

(5) COSTA: it is an open source project designed to provide a free toolbox for data assimilation for models conforming to the COSTA interface. The current available implementations are: data assimilation methods (ensemble, reduced rank square root, ensemble square root and COFFEE methods) and parameter estimation methods (simplex, conjugate gradient and LBFGS methods). This package is prepared for parallel computing. For details and documentation please refer to [COSTA].

(6) PALM: this project aims to provide a general structure for a modular implementation of a data assimilation system. An assimilation algorithm is split up into independent units. The package uses C, Fortran and MPI allowing portability. Today, PALM software is used in an operational way in the French Operational oceanography project MERCATOR. See [Buis et al. 2003].

(7) ESMF (Earth System Modeling Framework): Developed by the team centered at NCAR, ESMF is an open source software for building climate, numerical weather prediction, data assimilation, and other Earth science software applications. Some of the features of this package are: Fortran 90 and (partial) C/C++ interfaces, portability, MPI and OpenMP support, a large set of test, infrastructure and superstructure for coupling and building Earth system components. See [ESMF], [Collins et al. 2005] and [Hill et al. 2004].

Some of the packages cited above have the problem that are commercial or are programmed in a high-level language. However, the list also presents highly sophisticated projects that have methods and implementations like parallelism, interfaces with other languages, modularity and portability.

The aim of the package presented in this paper is to have not only the features of parallelism, portability, and modularity, but mainly ease of use. The only things the user needs to code are the basic things related to the model and observations (like setting the number of observations, covariance matrices, model propagation, etc). Derived types could have been defined in the modules, but the idea was to keep the programs at the maximum level of simplicity to simplify the task of the user. The parallel version is just an additional module, and the user does not need to worry about communicators, array distribution or parallelization strategies.

The paper is structured as follows: in Section 2 a mathematical background is presented, with a brief explanation of the implemented methods. Section 3 is devoted to explain some versions of the Kalman filter. Section 4 refers to the design of the package, Section 5 shows a sample application and finally Section 6 is dedicated to the conclusions.

2. MATHEMATICAL BACKGROUND.

Let us define the following entities:

$$n \doteq \text{dimension of the model state (variables we want to assimilate)}, \tag{1}$$

$$p \doteq \text{number of observations (usually } p \ll n\text{)}, \tag{2}$$

$$\mathbf{x^t} \doteq \text{true state}, \mathbf{x^t} \in \mathbb{R}^n, \tag{3}$$

$$\mathbf{x^f} \doteq \text{background (or forecast) model state}, \mathbf{x^f} \in \mathbb{R}^n, \tag{4}$$

$$\mathbf{x^a} \doteq \text{analysis model state}, \mathbf{x^a} \in \mathbb{R}^n, \tag{5}$$

$$\mathbf{y} \doteq \text{vector of observations}, \mathbf{y} \in \mathbb{R}^p, \tag{6}$$

$$H \doteq \text{observation operator}, H : \mathbb{R}^n \longrightarrow \mathbb{R}^p, \tag{7}$$

$$\mathbf{H} \doteq \text{tangent observation operator}, \mathbf{H} : \mathbb{R}^n \longrightarrow \mathbb{R}^p, \tag{8}$$

$$\mathbf{R} \doteq \text{observation error covariance matrix}, \mathbf{R} \in \mathbb{R}^{p \times p}, \tag{9}$$

$$\mathbf{P^f} \doteq \text{background (or forecast) error covariance matrix}, \mathbf{P^f} \in \mathbb{R}^{n \times n}, \tag{10}$$

$$\mathbf{P^a} \doteq \text{analysis error covariance matrix}, \mathbf{P^a} \in \mathbb{R}^{n \times n}, \tag{11}$$

$$\overline{\mathbf{x}} \doteq \text{expected value of } \mathbf{x}. \tag{12}$$

The pair $\left(\mathbf{x^f}, \mathbf{P^f}\right)$ gives a previous knowledge (or background) of the system state with an estimation of the error. It can be obtained, for example, by a model. The pair $(\mathbf{y}, \mathbf{R})$ provides the observations with an estimation of the observation errors. They can be obtained, for example, from measurement stations. The objective is to generate a new pair $(\mathbf{x^a}, \mathbf{P^a})$ where the analysis $\mathbf{x^a}$ is as close as possible to the true state in the r.m.s. sense.

Assuming that we have:

—non-trivial errors: $\mathbf{P^f}$ and $\mathbf{R}$ are positive definite matrices,

—unbiased errors: the expectation of the background and observation errors are zero, that is, $\overline{\mathbf{x^f} - \mathbf{x^t}} = \overline{\mathbf{y} - H\left(\mathbf{x^t}\right)} = 0$,

—uncorrelated errors: observation and background errors are mutually uncorrelated, that is, $\overline{\left(\mathbf{x^f} - \mathbf{x^t}\right)\left(\mathbf{y} - H\left(\mathbf{x^t}\right)\right)^T} = 0$,

then it can be proved (see [Bouttier and Courtier 2002]) that the analysis defined by corrections to the background, which depends linearly on background observation departures and has a minimum variance estimate, is:

$$\mathbf{x^a} = \mathbf{x^f} + \mathbf{K}\left(\mathbf{y} - H\left(\mathbf{x^f}\right)\right), \tag{13}$$

where $\mathbf{K}$ is called the gain matrix and it is defined by:

$$\mathbf{K} = \mathbf{P^f}\mathbf{H}^T\left(\mathbf{H}\mathbf{P^f}\mathbf{H}^T + \mathbf{R}\right)^{-1}. \tag{14}$$

This is called the BLUE (Best Linear Unbiased Estimator).

The covariance matrix for analysis errors is given (for any $\mathbf{K}$) by:

$$\mathbf{P^a} = \left(\mathbf{I} - \mathbf{K}\mathbf{H}\right)\mathbf{P^f}\left(\mathbf{I} - \mathbf{K}\mathbf{H}\right)^T + \mathbf{K}\mathbf{R}\mathbf{K}^T, \tag{15}$$

and if $\mathbf{K}$ is the optimal least-squares gain, the expression is reduced to:

$$\mathbf{P^a} = \left(\mathbf{I} - \mathbf{K}\mathbf{H}\right)\mathbf{P^f}. \tag{16}$$

In practice, the user does not know $\mathbf{K}$ exactly and the formulation (16) can lead to an erroneous gain matrix, so it is convenient to work with the analysis error covariance matrix defined by (15).

The problem of finding the pair $(\mathbf{x^a}, \mathbf{P^a})$ is equivalent to the minimization of the following functional:

$$J\left(\mathbf{x}\right) = \left(\mathbf{x} - \mathbf{x^f}\right)^T {\mathbf{P^f}}^{-1}\left(\mathbf{x} - \mathbf{x^f}\right) + \left(\mathbf{y} - H\left(\mathbf{x}\right)\right)^T \mathbf{R}^{-1}\left(\mathbf{y} - H\left(\mathbf{x}\right)\right). \tag{17}$$

For the evolution in time of the whole system, we need a model to propagate the state vector and the forecast error covariance matrix in order to be able to apply the BLUE, and get a better approximation of the true state. Suppose that we have a model $M_{l\rightarrow l+1} : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ that takes forward the state vector from time step l to time step $l+1$, and also suppose that the tangent linear model $\mathbf{M}_{l\rightarrow l+1} : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ is available. Assuming the hypotheses for the BLUE, and the following ones:

—forecast errors: the model error $M_{l\rightarrow l+1}\left(\mathbf{x}_l^\mathbf{t}\right) - \mathbf{x}_{l+1}^\mathbf{t}$ is unbiased with known model error covariance matrix $\mathbf{Q}_l$,

- uncorrelated analysis and model errors: the analysis error $\mathbf{x}^{\mathbf{a}}_l - \mathbf{x}^{\mathbf{t}}_l$ and the model error $M_{l\rightarrow l+1}\left(\mathbf{x}^{\mathbf{t}}_l\right) - \mathbf{x}^{\mathbf{t}}_{l+1}$ are mutually uncorrelated,

we can prove (see [Bouttier and Courtier 2002]) that the optimal way (in the least square sense) to assimilate sequentially the observations is given by the Kalman filter algorithm:

$$
\begin{aligned}
\mathbf{x}^{\mathbf{f}}_{l+1} &= M_{l\rightarrow l+1}\left(\mathbf{x}^{\mathbf{a}}_l\right), && (18)\\
\mathbf{P}^{\mathbf{f}}_{l+1} &= \mathbf{M}_{l\rightarrow l+1}\mathbf{P}^{\mathbf{a}}_l\mathbf{M}^T_{l\rightarrow l+1} + \mathbf{Q}_l, && (19)\\
\mathbf{K}_{l+1} &= \mathbf{P}^{\mathbf{f}}_{l+1}\mathbf{H}^T_{l+1}\left(\mathbf{H}_{l+1}\mathbf{P}^{\mathbf{f}}_{l+1}\mathbf{H}^T_{l+1} + \mathbf{R}_{l+1}\right)^{-1}, && (20)\\
\mathbf{x}^{\mathbf{a}}_{l+1} &= \mathbf{x}^{\mathbf{f}}_{l+1} + \mathbf{K}_{l+1}\left[\mathbf{y}_{l+1} - H_{l+1}\left(\mathbf{x}^{\mathbf{f}}_{l+1}\right)\right], && (21)\\
\mathbf{P}^{\mathbf{a}}_{l+1} &= \left(\mathbf{I} - \mathbf{K}_{l+1}\mathbf{H}_{l+1}\right)\mathbf{P}^{\mathbf{f}}_{l+1}\left(\mathbf{I} - \mathbf{K}_{l+1}\mathbf{H}_{l+1}\right)^T + \\
&\quad + \mathbf{K}_{l+1}\mathbf{R}_{l+1}\mathbf{K}^T_{l+1}. && (22)
\end{aligned}
$$

The subscripts represent time evolution.

The equations (18)-(19) are the prediction part of the filter, whereas the equations (20)-(22) are the correction in order to minimize the variance of the analysis. As it is, the filter can be applied to small and medium problems, but it presents some problems regarding implementation for large scale problems:

- Storing: for atmospheric applications, we can have $n \approx 10^6$ and a full error covariance matrix of size $n \times n$ may be in excess of a teraword (see the assimilation system at the Data Assimilation Office [Lyster et al. 2003] as an example).
- Too many model evaluations: the equation (19) requires $2n$ evaluations of the tangent linear model. In some cases the tangent version of the model is obtained by an automatic differentiation package, or it is computed using two evaluations of the model. Therefore, the propagation of the forecast error covariance matrix costs too much.
- Matrix-vector manipulation: normally, the large scale models represent the system state as 3D matrices, so a transformation matrix→vector is needed to apply one filter step, and then a transformation vector→matrix to continue the propagation in time.
- Non-linearities: the extended Kalman filter (18)-(22) makes a linearization of the model, but this linearization has shown to be invalid in a number of applications (see [Evensen 1994; 1997]).

The next section is devoted to explain the main algorithms that are used in the assimilation modules.

3. VERSIONS OF THE KALMAN FILTER ALGORITHM.

3.1 EKF (Extended Kalman Filter).

The algorithm (18)-(22) without the time subscripts (for simplicitiy) is as follows:

$$
\begin{aligned}
\mathbf{x}^{\mathbf{f}} &= M\left(\mathbf{x}^{\mathbf{a}}\right), && (23)\\
\mathbf{P}^{\mathbf{f}} &= \mathbf{M}\mathbf{P}^{\mathbf{a}}\mathbf{M}^T + \mathbf{Q}, && (24)\\
\mathbf{K} &= \mathbf{P}^{\mathbf{f}}\mathbf{H}^T\left(\mathbf{H}\mathbf{P}^{\mathbf{f}}\mathbf{H}^T + \mathbf{R}\right)^{-1}, && (25)
\end{aligned}
$$

$$\begin{aligned}\mathbf{x^a} &= \mathbf{x^f} + \mathbf{K}\left(\mathbf{y} - H\left(\mathbf{x^f}\right)\right), &(26)\\ \mathbf{P^a} &= \left(\mathbf{I} - \mathbf{KH}\right)\mathbf{P^f}\left(\mathbf{I} - \mathbf{KH}\right)^T + \mathbf{KRK}^T. &(27)\end{aligned}$$

3.2 RRSQRTKF (Reduced Rank Square Root Kalman Filter).

The covariance matrices have good properties because they are symmetric and (semi) positive definite. Therefore they can be factorized and it is possible to compute the square root (for example, via the Cholesky decomposition, or the SVD). Whereas in some contexts the square root of a matrix P means $P^{1/2}$, from now on we will say that a matrix S is a square root of a matrix P if it holds $P = SS^T$. Compared with standard Kalman filtering algorithms, square root algorithms are known for their superior numerical properties (see [Bierman 1977] and [Kaminski et al. 1971]). "They are also more numerically robust than non-square-root forms because they are less susceptible to rounding errors and prevent the error covariance matrices from becoming negative definite" (see [Brown and Gaston 1995]). In [Paige 1985] the author suggests general representations of covariance matrices in linear filtering in which the covariance and information matrices are implicitly defined. He also develops numerically reliable algorithms (see also [Kourouklis 1977]).

Stability problems can be reduced using the square root form of covariance matrices, but we could have storing and timing difficulties. The solution for this is to take a square root covariance matrix with less columns. That is, given a covariance matrix $\mathbf{P}$:

$$\mathbf{P} = \mathbf{SS}^T, \quad \mathbf{S} \in \mathbb{R}^{n\times n} \qquad \longrightarrow \qquad \mathbf{P} \approx \mathbf{SS}^T, \quad \mathbf{S} \in \mathbb{R}^{n\times m}, \tag{28}$$

where $m \ll n$. With this formulation we still have the symmetry and semi positive definiteness of the covariance matrices. The integer m is usually called the number of modes.

Let us define:

$$\begin{aligned} m_a &\doteq \text{number of modes of the analysis error covariance matrix}, &(29)\\ m_f &\doteq \text{number of modes of the forecast error covariance matrix}, &(30)\\ m_q &\doteq \text{number of modes of the model error covariance matrix}, &(31)\\ m_r &\doteq \text{number of modes of the observation error covariance matrix}, &(32)\\ \mathbf{S^a} &\doteq \text{square root covariance matrix of analysis errors}, \mathbf{S^a} \in \mathbb{R}^{n\times m_a}, &(33)\\ \mathbf{S^f} &\doteq \text{square root covariance matrix of forecast errors}, \mathbf{S^f} \in \mathbb{R}^{n\times m_f}, &(34)\\ \mathbf{S^m} &\doteq \text{square root covariance matrix of model errors}, \mathbf{S^m} \in \mathbb{R}^{n\times m_q}, &(35)\\ \mathbf{S^o} &\doteq \text{square root covariance matrix of observation errors}, \mathbf{S^o} \in \mathbb{R}^{n\times m_r}, &(36)\end{aligned}$$

Then, covariance matrices in the Kalman filter algorithm (23)-(27) can be transformed in terms of the reduced rank square root as follows:

$$\begin{aligned}\mathbf{S^f}\mathbf{S^f}^T &\approx \mathbf{P^f} \approx \mathbf{MS^a}\mathbf{S^a}^T\mathbf{M}^T + \mathbf{S^m}\mathbf{S^m}^T = \left[\mathbf{MS^a} \mid \mathbf{S^m}\right]\left[\mathbf{MS^a} \mid \mathbf{S^m}\right]^T \Longrightarrow\\ \mathbf{S^f} &\approx \left[\mathbf{MS^a} \mid \mathbf{S^m}\right], &(37)\end{aligned}$$

$$\mathbf{K} = \mathbf{P^f}\mathbf{H}^T\left(\mathbf{HP^fH}^T + \mathbf{R}\right)^{-1} \approx \mathbf{S^f}\mathbf{S^f}^T\mathbf{H}^T\left(\mathbf{HS^f}\mathbf{S^f}^T\mathbf{H}^T + \mathbf{S^o}\mathbf{S^o}^T\right)^{-1} \Longrightarrow$$

$$\mathbf{K} \approx \mathbf{S^f}\left(\mathbf{HS^f}\right)^T\left(\left[\mathbf{HS^f} \mid \mathbf{S^o}\right]\left[\mathbf{HS^f} \mid \mathbf{S^o}\right]^T\right)^{-1}, \tag{38}$$

$$\begin{aligned}
\mathbf{S^a}\mathbf{S^a}^T &\approx \left(\mathbf{I}-\mathbf{KH}\right)\mathbf{P^f}\left(\mathbf{I}-\mathbf{KH}\right)^T+\mathbf{KRK}^T \\
&\approx \left(\mathbf{I}-\mathbf{KH}\right)\mathbf{S^f}\mathbf{S^f}^T\left(\mathbf{I}-\mathbf{KH}\right)^T+\mathbf{KS^o}\mathbf{S^o}^T\mathbf{K}^T \\
&= \left[\left(\mathbf{I}-\mathbf{KH}\right)\mathbf{S^f} \mid \mathbf{KS^o}\right]\left[\left(\mathbf{I}-\mathbf{KH}\right)\mathbf{S^f} \mid \mathbf{KS^o}\right]^T \Longrightarrow \\
\mathbf{S^a} &\approx \left[\left(\mathbf{I}-\mathbf{KH}\right)\mathbf{S^f} \mid \mathbf{KS^o}\right] \qquad (39)
\end{aligned}$$

Notice that:

—the number of forecast modes m_f is $m_a + m_q$,

—after the assimilation step, the square root of the analysis error covariance matrix (39) has a larger number of columns, namely, it has $m_a + m_q + m_r$ columns, and a truncation strategy is needed in order to be able to continue with the algorithm (in [Treebushny and Madsen 2003] a procedure based on the Lanczos decomposition algorithm is used, in [van Loon and Heemink 1997], [Segers et al. 2000] and [Segers 2002] a procedure based in the SVD is explained, in [Hoteit and Pham 2004] a reduced-order extended Kalman filter is proposed).

—generally, the number of observations p is much less than the dimension of the state space n. In this cases we should take $m_r = p$ in order to avoid loss of information.

Finally, the RRSQRTKF algorithm is:

$$\begin{aligned}
\mathbf{x^f} &= M\left(\mathbf{x^a}\right), && (40)\\
\mathbf{S^f} &= \left[\mathbf{MS^a} \mid \mathbf{S^m}\right], && (41)\\
\mathbf{K} &= \mathbf{S^f}\left(\mathbf{HS^f}\right)^T\left(\left[\mathbf{HS^f} \mid \mathbf{S^o}\right]\left[\mathbf{HS^f} \mid \mathbf{S^o}\right]^T\right)^{-1}, && (42)\\
\mathbf{x^a} &= \mathbf{x^f}+\mathbf{K}\left[\mathbf{y}-H\left(\mathbf{x^f}\right)\right], && (43)\\
\mathbf{S^a} &= \left[\left(\mathbf{I}-\mathbf{KH}\right)\mathbf{S^f} \mid \mathbf{KS^o}\right], && (44)\\
\mathbf{S^a} &\leftarrow \text{reduce } \mathbf{S^a} \text{ to } m_a \text{ columns}, && (45)
\end{aligned}$$

3.3 ENKF (ENsemble Kalman Filter).

The idea of the ensemble Kalman filter is to represent the error statistics using an ensemble of model states. Therefore, instead of forecasting the analysis error covariance matrix using the tangent linear model, the model states are propagated and the covariance matrix is recovered from them. It is very easy to implement, and there is no need to propagate full covariance matrices, but only a few model states (or modes) that contain the information about the system and its statistics. It captures non-linearities of the model (see [Evensen 1994; 2003]) and a tangent linear model is not necessary (sometimes not available). The only problem with this version is that the error in the Monte Carlo sampling decreases proportionally to $1/\sqrt{N}$ where N is the number of modes. The ENKF has been applied succesfully for several models (in [Allen et al. 2002] it is used in the European Regional Seas Ecosystem Model ERSEM, [Segers 2002] shows an implementation in the LOTOS model, in [Annan et al. 2005] an efficient method for parameter estimation and ensemble forecasting in climate modelling is developed).

Let us define m the number of members of the ensemble (or number of modes). The ENKF algorithm explained in [Evensen 2003] is:

$$\text{Generate (only the first time): } \xi^{\mathbf{a}i} \in \mathcal{N}\left(\mathbf{x^a}, \mathbf{P^a}\right), \quad i = 1 : m, \tag{46}$$

$$\text{Propagate: } \xi^{\mathbf{f}i} = M\left(\xi^{\mathbf{a}i}\right) + \eta^i, \quad \eta^i \in \mathcal{N}\left(\mathbf{0}, \mathbf{Q}\right), \quad i = 1 : m, \tag{47}$$

$$\text{Estimate: } \mathbf{x^f} = \frac{1}{m}\sum_{i=1}^{m} \xi^{\mathbf{f}i}, \tag{48}$$

$$\text{Estimate: } \mathbf{P^f} = \frac{1}{m-1}\sum_{i=1}^{m} \left(\xi^{\mathbf{f}i} - \mathbf{x^f}\right)\left(\xi^{\mathbf{f}i} - \mathbf{x^f}\right)^T, \tag{49}$$

$$\text{Gain matrix: } \mathbf{K} = \mathbf{P^f}\mathbf{H}^T\left(\mathbf{H}\mathbf{P^f}\mathbf{H}^T + \mathbf{R}\right)^{-1}, \tag{50}$$

$$\text{Correct: } \xi^{\mathbf{a}i} = \xi^{\mathbf{f}i} + \mathbf{K}\left(\mathbf{y}^i - H\left(\xi^{\mathbf{f}i}\right)\right), \quad \mathbf{y}^i \in \mathcal{N}\left(\mathbf{y}, \mathbf{R}\right), \quad i = 1 : m, \tag{51}$$

$$\text{Estimate: } \mathbf{x^a} = \frac{1}{m}\sum_{i=1}^{m} \xi^{\mathbf{a}i}, \tag{52}$$

$$\text{Estimate: } \mathbf{P^a} = \frac{1}{m-1}\sum_{i=1}^{m} \left(\xi^{\mathbf{a}i} - \mathbf{x^a}\right)\left(\xi^{\mathbf{a}i} - \mathbf{x^a}\right)^T, \tag{53}$$

3.4 RRSQRTENKF (Reduced Rank Square Root ENsemble Kalman Filter).

This version of the filter presents a version of the Ensemble Kalman filter, but considering square root of covariance matrices (see [Whitaker and Hamill 2002; Tippett et al. 2003]). This method is called EnSR and from now on and for the rest of the paper it will be identified by RRSQRTENKF . The algorithm (46)-(53) would be:

$$\text{Generate (only the first time): } \xi^{\mathbf{a}i} \in \mathcal{N}\left(\mathbf{x^a}, \mathbf{S^a}\mathbf{S^a}^T\right), \quad i = 1 : m, \tag{54}$$

$$\text{Propagate: } \xi^{\mathbf{f}i} = M\left(\xi^{\mathbf{a}i}\right) + \eta^i, \quad \eta^i \in \mathcal{N}\left(\mathbf{0}, \mathbf{S^m}\mathbf{S^m}^T\right), \quad i = 1 : m, \tag{55}$$

$$\text{Estimate: } \mathbf{x^f} = \frac{1}{m}\sum_{i=1}^{m} \xi^{\mathbf{f}i}, \tag{56}$$

$$\text{Estimate: } \mathbf{S^f} = \frac{1}{\sqrt{m-1}}\begin{pmatrix} \vdots & & \vdots \\ \xi^{\mathbf{f}1} - \mathbf{x^f} & \cdots & \xi^{\mathbf{f}m} - \mathbf{x^f} \\ \vdots & & \vdots \end{pmatrix}, \tag{57}$$

$$\text{Gain matrix: } \mathbf{K} = \mathbf{S^f}\left(\mathbf{H}\mathbf{S^f}\right)^T\left(\left[\mathbf{H}\mathbf{S^f} \mid \mathbf{S^o}\right]\left[\mathbf{H}\mathbf{S^f} \mid \mathbf{S^o}\right]^T\right)^{-1}, \tag{58}$$

$$\begin{aligned}\text{Correct: } & \xi^{\mathbf{a}i} = \xi^{\mathbf{f}i} + \mathbf{K}\left(\mathbf{y}^i - H\left(\xi^{\mathbf{f}i}\right)\right), \\ & \qquad \mathbf{y}^i \in \mathcal{N}\left(\mathbf{y}, \mathbf{S^o}\mathbf{S^o}^T\right), \quad i = 1 : m,\end{aligned} \tag{59}$$

$$\text{Estimate: } \mathbf{x^a} = \frac{1}{m}\sum_{i=1}^{m} \xi^{\mathbf{a}i}, \tag{60}$$

$$\text{Estimate: } \mathbf{S^a} = \frac{1}{\sqrt{m-1}} \begin{pmatrix} \vdots & & \vdots \\ \xi^{\mathbf{a}1} - \mathbf{x^a} & \cdots & \xi^{\mathbf{a}m} - \mathbf{x^a} \\ \vdots & & \vdots \end{pmatrix}, \tag{61}$$

Notice that both the ENKF and the RRSQRTENKF algorithms require the generation of random vectors with a prescribed distribution. This can be done every time step, but it is better to let the ensemble evolve according to the model and the observations. From the numerical experiments one can deduce that the ENKF and the RRSQRTENKF need a time interval until the ensemble members represent well the dynamical system.

Another remark is that the user could choose the number of columns of the reduced rank square root covariance matrices equal to the ensemble size. If this is not the case, the user should add a reduction step as in (45).

4. DESIGN.

4.1 Overview.

The package presented here presents a set of Fortran 90 modules that implement the Kalman filter adapted for large scale problems. The methods that are implemented are the Extended Kalman filter (identified as EKF), the Reduced Rank Square Root filter (identified as RRSQRTKF), the Ensemble Kalman filter (identified as ENKF) and the Reduced Rank Square Root Ensemble filter (identified as RRSQRTENKF). Each method and each necessary task into the assimilation (for example model and observations) is coded into modules that can be replaced according to the application.

A list of capabilities is described below:

—**Modularity**: as mentioned before, each task into an assimilation implementation is separated one from another. It means that observations, model and assimilation are different entities. For example, if the observation stations change the location, there is no need to modify the main code, but only a module related with them. If we need to change the model, there is no need to transform all the code, but only the module related with the model. If we want to change the Kalman filter version, a change in the module related with the assimilation will suffice. This will allow the user to make minimum changes in all the codes around a modelling system (model, data formats, libraries, configuration files) in order to avoid programmer bugs.

—**Simplicity**: there are no derived types of variables defined in the code. It is clear that derived types of variables are a useful language tool in order to make powerful codes, but in this case the intention was to determine the global variables and the specific functions associated to the assimilation. The users are encouraged to introduce all the new abstract types of variables and all the complexity they need in those modules that have to be edited, rather than adjust their codes to an existent structure. The code has been prepared for both an expert programmer as well as for a medium programmer.

—**Language**: the modules are programmed in Fortran 90. The choice of the language was made because a big number of large scale models are written either in Fortran 77 or in Fortran 90. Only Fortran standards have been used, and the code will work with almost any Fortran compiler. It has been succesfully compiled and executed with the Intel Fortran Compiler, Portland Fortran Compiler, GNU Fortran and g95, and its MPI wrappers.

—**Precision**: the modules can handle single and double precision. Some models have their outputs in single precision, others in double precision, so this is a useful feature. The switch between these two choices is done changing only one parameter in the whole code.

—**Parallelism**: repeated tasks like propagating states (or applying the observation operator, or the tangent observation operator, or the tangent model) are parallelized using the master-slave strategy with MPI (Message Passing Interface). In this case a set of independent tasks are sent to the processors. Once the task is performed, the slave receives a new task from the master. Linear algebra operations are performed using BLACS (Basic Linear Algebra Communication Subprograms) and SCALAPACK (Scalable LAPACK). The global matrices are distributed in a processor grid, then the operation is performed in each processor over local matrices. Finally, the global matrices are rebuilt from local pieces. The parallelism is included in a module and the user just needs to call the parallel routines implemented there avoiding communicators and memory distribution.

5. SAMPLE APPLICATION.

We illustrate the use of our software to solve a problem to assimilate CO concentrations in the area of Santiago de Chile. The user manual which accompanies the software contains a number of other illustrative examples as well as programming details.

5.1 Assimilation of CO.

The implementation was done using the MATCH model [MATCH]. The version of the filter applied was the Reduced Rank Square Root Kalman filter with 50 samples.

A grid of 41 × 41 is considered in the horizontal domain, and 16 levels in the vertical component. Then, the dimension of the space state is set to 26896. For the initialization, the model was run for a period of three days considering an atmosphere free of CO at the beginning. The initial state vector is set to the last output of the MATCH initial run, adding an error of 100%. The simulation period was 13 days starting at June 17th, 1999, and ending at June 30th of the same year, performing an analysis each 3 hours (the time step for observations). The meteorological fields were generated using the HIRLAM model [HIRLAM] with a resolution of 0.01 degrees ($\approx$ 1 km.) and at 1 hour of time resolution.

The CO emissions were generated by MODEM [MODEM]. See figure 1.

There are eight monitoring stations located at different positions in Santiago. The observations are taken from the measuring stations at the surface level, at intervals of 3 hours, where the analysis step is performed. In figure 1 we show the eigth monitoring stations and the domain of simulation: (1) Seminario, (2) Independencia-Recoleta and (5) Parque O'Higgins are in the city center of Santiago; (3) La Florida is in the east and south area; (4) Las Condes - Vitacura is monitoring the north-east sector; (6) Pudahuel-Cerro Navia and (7) Cerrillos register measurements at the west side of the city; and (8) El Bosque is located at the south. The error in the observations was set to 30% of the reported value.

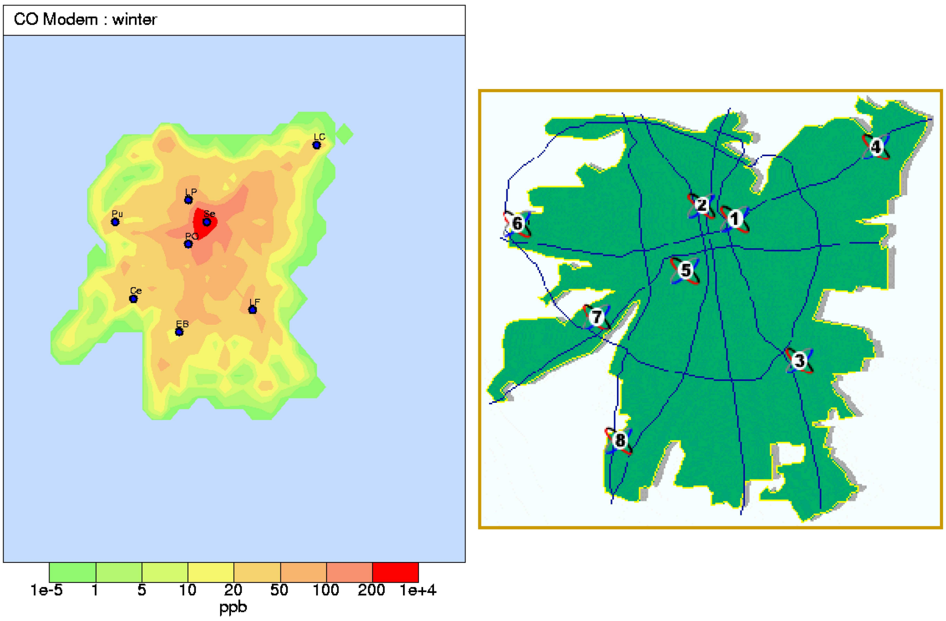

Fig. 1. Emissions generated by MODEM and stations.

After 100 hours of simulation running assimilation, we obtain the figure 2, where we can see a comparison of the model, observations, truth and assimilation.

6. CONCLUSIONS.

Due to the constant growth of the efficiency and speed of the computers, the investigation related to modelization has received a new impulse. New models and strategies have been proposed requiring more computer power. Therefore, new ways of improving forecasts can be implemented, as for example, the discrete time Kalman filter. In the last decades scientists have realized that improving only a model is not enough to get good forecasts, because always there are errors in the data. We can have a perfect model, but if the error is in the data, we have nothing to do. That is why information coming from observations must be used to correct the model. The idea behind this work is to provide a platform where researchers (from oceanography, climatology, etc.) can make use of the Kalman filter in their prediction models in an easy way, changing their source codes as little as possible in order to add assimilation material.

The package is oriented to large scale problems, although some versions of the Kalman filters withouth simplifications are also implemented. The modules are organized to separate tasks one from another. Model, observations and assimilation are considered as different objects, in which one of them can be changed without

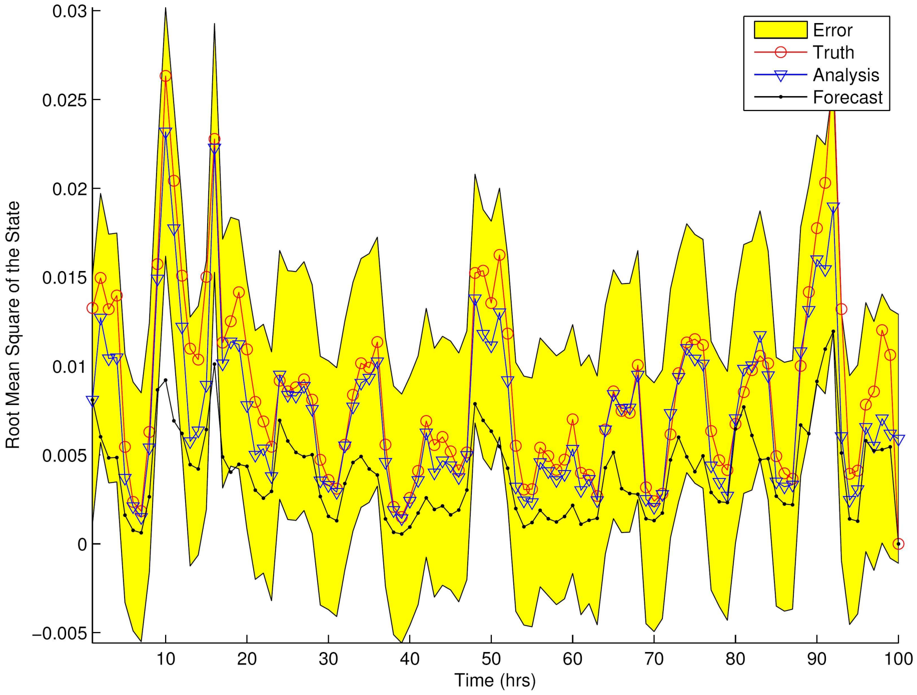

Fig. 2. Results of the CO assimilation.

altering all the code. Thus, the programming of an implementation is cleaner and one can avoid a lot of human errors.

Parallelism is implemented using MPI. In case of performing multiple runs of some operator, we have used the master-slave strategy. Linear algebra operations are optimized using the BLAS, LAPACK, PBLAS and SCALAPACK libraries.

The treatment of vectors and matrices takes advantage of the language features. This aspect is very important because most of the models in 3D use 3D matrices to represent the state in a domain, but the filter needs the representation of the state as a vector, so an efficient identification matrix/vector is required.

These assimilation libraries can be inserted in any model if observations are available. There are versions of the filter that do not require the tangent linear model (sometimes not available), but can capture the strong non linearities present in a lot of problems, for example, air pollution. These libraries do not deal with the settings of the covariance matrices because this strongly depends on the model and received observations. The most difficult part of the assimilation is how to set the covariance matrices of model and observation errors. The user must know the modeling system in detail in order to obtain an effective assimilation, and maybe to improve some parts of the code to gain efficiency in a particular problem.

Numerical tests in 0D, 1D, 2D and 3D have been implemented with satisfactory results in all the versions of the filter. In 0D, solving the ordinary differential equation as in the example provided in the user's manual, one can see how efficient the filter is in order to reduce the uncertainties. One can realize that the full Kalman filter produces discontinuities in the assimilated solution, and that discontinuities occur when an assimilation step is performed. Instead, the filters based on Monte Carlo methods (ENKF and RRSQRTENKF) produce smoother solutions. From the experiments one can see that the ensemble needs a period of time in order to learn how to represent better the system state. This also can be seen in 0D. For large scale problems it is impossible to apply the full filter. For the two 3D problems presented in the examples (see the user's manual), we have seen real applications using the proposed package. In the first case we were able to assimilate CO concentrations in the model MATCH, and in the second case we were able to assimilate O3 concentrations in the POLAIR3D model. In the last case a detailed description of how the package has to be implemented in the model is given.

Due to the need of improving models and modularizing the assimilation system (to get better, easier and cleaner codes), and due to the need of optimized packages for assimilation of large scale systems, these libraries are a helpful tool for the modeler, and a start point in order to tune the variables involved in the filter.

ACKNOWLEDGMENTS

We want to thank to the referees and the editors for their helpful suggestions to improve the submission of this paper, as well as to Ing. Karina Buchhorsts and the team at CMM (Center of Mathematical Modeling) in Santiago de Chile for their help in running the assimilation experiments.

REFERENCES

Allen, J., Eknes, M., and Evensen, G. 2002. An Ensemble Kalman Filter with a Complex Marine Ecosystem Model: Hindcasting Phytoplankton in the Cretan Sea. *Annales Geophysicae 20*, 1–13.

Annan, J., Hargreaves, J., Edwards, N., and Marsh, R. 2005. Parameters Estimation in an Intermediate Complexity Earth System Model Using an Ensemble Kalman Filter. *Ocean Modelling 8*, 135–154.

Asif, A. 2004. Fast Implementations of the Kalman-Bucy Filter for Satellite Data Assimilation. *IEEE Signal Processing Letters 11,* 2.

Azuma, R. and Bishop, G. 1994. Improving Static and Dynamic Registration in an Optimcal See-Through HMD. *Computer Graphics - SIGGRAPH 94 Conference Proceedings ed.*, 197–204.

BAYES++. http://bayesclasses.sourceforge.net/Bayes++.html.

BFL. http://www.orocos.org/bfl.

Bierman, G. 1977. *Factorization Methods for Discrete Sequential Estimation.* Academic Press, New York.

Bouttier, F. and Courtier, P. 2002. *Data Assimilation Concepts and Methods.* Meteorological Training Course Lecture Series ECMWF.

Brown, D. and Gaston, F. 1995. The Design of Parallel Square-Root Covariance Kalman Filters Using Algorithm Engineering. *the VLSI journal 20*, 101–119.

Buis, S., Declat, D., Gondet, E., Massart, S., Morel, T., and Thual, O. 2003. PALM: A Dynamic Parallel Coupler for Data Assimilation. *Geophyphisical Research Abstracts 5,* 05476.

Charalambous, C. and Hibey, J. 2001. Exact Filters for Newton-Raphson Parameter Estimation Algorithms for Continuous-Time Partially Observed Stochastic Systems. *Systems & Control Letters 42*, 101–115.

Chen, T., Morris, J., and Martin, E. 2005. Particle Filters for State and Parameter Estimation in Batch Processes. *Journal of Process Control 15*, 665–673.

Chin, T., Karl, W., and Willsky, A. 1995. A Distributed and Iterative Method for Square Root Filtering in Space-time Estimation. *Automatica 31,* 1, 67–82.

Collins, N., Theurich, G., DeLuca, C., Suarez, M., Trayanov, A., Balaji, V., Li, P., Yang, W., Hill, C., and da Silva, A. 2005. Design and Implementation of Components in the Earth System Modeling Framework. *International Journal of High Performance Computing Applications 19*, 341–350.

COSTA. http://www.costapse.org/.

Ertürk, S. 2002. Real-Time Digital Image Stabilization Using Kalman Filters. *Real-Time Imaging 8*, 317–328.

ESMF. http://www.esmf.ucar.edu/.

Evensen, G. 1994. Sequential Data Assimilation with a Non-Linear Quasi-Geostrophic Model Using Monte Carlo Methods to Forecast Errors Statistics. *J. Geophys. Res 99 C5*, 10143–10162.

Evensen, G. 1997. Advanced Data Assimilation for Strongly Non-Linear Dynamics. *Mon. Weather Rev. 125*, 1342–1354.

Evensen, G. 2003. The Ensemble Kalman Filter: Theoretical Formulation and Practical Implementation. *Ocean Dynamics 53*, 343–367.

Flemming, J., Reimer, E., and Stern, R. 2001. Long Term Evaluation of the Ozone Forecast by an Eulerian Model. *Phys. Chem. Earth (B) 26,* 10, 775–779.

Hanea, R., Velders, G., and Heemink, A. 2005. Comparison of Different Hybrid Kalman Filter Algorithms for a Large Scale Chemistry Transport Model. *Geophysical Research Abstracts 7,* 00108.

Hill, C., DeLuca, C., Balaji, V., Suarez, M., and da Silva, A. 2004. Architecture of the Earth System Modeling Framework. *Computing in Science and Engineering 6*, 18–28.

HIRLAM. http://hirlam.org/.

Hoteit, I. and Pham, D. 2004. An Adaptively Reduced-Order Extended Kalman Filter for Data Assimilation in the Tropical Pacific. *Journal of Marine Systems 45*, 173–188.

Hoteit, I., Pham, D., and Blum, J. 2001. A Semi-Evolutive Partially Local Filter for Data Assimilation. *Marine Pollution Bulletin 43,* 7-12, 164–174.

Hoteit, I., Pham, D., and Blum, J. 2002. A Simplified Reduced Order Kalman Filtering and Application to Altimetric Data Assimilation in Tropical Pacific. *Journal of Maryne Systems 36*, 101–127.

Hoteit, I., Triantafyllou, G., and Petihakis, G. 2004. Towards a Data Assimilation System for the Cretan Sea Ecosystem Using a Simplified Kalman Filter. *Journal of Marine Systems 45*, 159–171.

Kalman, R. 1960. A New Approach to Linear Filtering and Prediction Problems. *Transaction of the ASME - Journal of Basic Engineering 82 (Series D)*, 33–45.

KALMTOOL. http://www.iau.dtu.dk/research/control/kalmtool.html.

Kaminski, P., Bryson, A., and Schmidt, S. 1971. Discrete Square Root Filtering: A Survey of Current Techniques. *IEEE Trans. Autom. Control AC-16*, 727–736.

Kim, S. and Iltis, R. March 20-22, 2002. Performance Comparison of Particle and Extended Kalman Filter Algorithms for GPS C/A Code Tracking and Interference Rejection. *Conference on Information Sciences and Systems, Princeton University*.

Kourouklis, S. 1977. Implementing a General and Numerically Stable Aproach to the Kalman Filtering Problem. *Master's thesis, School of Computer Science, McGill University, Montreal, Quebec, Canada*.

Kuo, C., Chao, C., and Hsieh, C. 2002. An Efficient Motion Estimation Algorithm for Video Coding Using Kalman Filter. *Real-Time Imaging 8*, 253–264.

Lyster, P., Guo, J., Clune, T., and Larson, J. 2003. The Computational Complexity and Parallel Scalability of Atmospheric Data Assimilation Algorithms. *NASA Data Assimilation Office (DAO), Goddard Laboratory for Atmospheres*.

MATCH. http://www.smhi.se/sgn0106/if/FoUl/en/models/match/match.html.

MODEM. http://www.sectra.cl/contenido/metodologia/transporte_medioambiente/estimacion_emisiones_fuentes_moviles_modem.asp.

PAIGE, C. 1985. Covariance Matrix Representation in Linear Filtering. *In "Linear Algebra and Its Role in Systems Theory", B.N. Datta Ed., AMS Publns., Providence RI 47*, 309–321.

PHAM, D., VERRON, J., AND ROUBAUD, M. 1998. A Singular Evolutive Extended Kalman Filter for Data Assimilation in Oceanography. *Journal of Marine Systems 16,* 3-4, 323–340.

SEGERS, A. 2002. *Data Assimilation in Atmospheric Chemistry Models Using Kalman Filtering.* Delft Univ Pr.

SEGERS, A., HEEMINK, A., VERLAAN, M., AND VAN LOON, M. 2000. A Modified RRSQRT-Filter for Assimilating Data in Atmospheric Chemistry Models. *Environmental Modelling & Software 15*, 663–671.

SERAFY, G. E., VAN DER A, R., ESKES, H., AND KELDER, H. 2002. Assimilation of 3D Ozone Field in Global Chemistry-Transport Models Using Kalman Filter. *Adv. Space Res. 30,* 11, 2473–2478.

STSA. http://www.omatrix.com/stsa.html.

TIPPETT, M. K., ANDERSON, J. L., BISHOP, C. H., HAMILL, C. H., AND WHITAKER, J. S. 2003. Ensemble square root filters. *Mon. Weather Rev. 131,* 1485-1490.

TREEBUSHNY, D. AND MADSEN, H. 2003. A New Reduced Rank Square Root Kalman Filter Technique for Data Assimilation in Large Scale Modelling Systems. *Geophyphisical Research Abstracts 5,* 12800.

TREEBUSHNY, D. AND MADSEN, H. 2005. On the Construction of a Reduced Rank Square-Root Kalman Filter for Efficient Uncertainty Propagation. *Future Generation Computer Systems 21*, 1047–1055.

VAN LOON, M., BUILTJES, P., AND SEGERS, A. 2000. Data assimilation of ozone in the atmospheric transport chemistry model LOTOS. *Environmental Modelling & Software 15*, 603–609.

VAN LOON, M. AND HEEMINK, A. 1997. Kalman Filtering for Non Linear Atmospheric Chemistry Models: First Experiences. *Technical Report MAS-R9711, CWI, Amsterdam.*

WELCH, G. AND BISHOP, G. 2001. *An Introduction to the Kalman Filter.* SIGGRAPH 2001 - Course 8, ACM, Inc.

WHITAKER, J. S. AND HAMILL, T. M. 2002. Ensemble Data Assimilation without perturbed observations. *Mon. Weather Rev. 130,* 1913-1924.

ZHANG, X., HEEMINK, A., AND EIJKEREN, J. 1997. Data Assimilation in Transport Models. *Applied Mathematical Modelling 21,* 1, 2–14.

ZHANG, X., HEEMINK, A., JANSSEN, L., JANSSEN, P., AND SAUTER, F. 1999. A Computationally Efficient Kalman Smoother for the Evaluation of the CH_4 Budget in Europe. *Applied Mathematical Modelling 23*, 109–129.