

## The Variational Kalman filter and an efficient implementation using limited memory BFGS

H. Auvinen<sup>1,\*</sup>, J. M. Bardsley<sup>2</sup>, H. Haario<sup>1</sup> and T. Kauranne<sup>1</sup>

<sup>1</sup> *Department of Mathematics and Physics  
Lappeenranta University of Technology  
Lappeenranta, Finland*

<sup>2</sup> *Department of Mathematical Sciences  
University of Montana  
Missoula, Montana 59812, USA*

### SUMMARY

In the field of state space estimation and data assimilation, the Kalman filter (KF) and the extended Kalman filter (EKF) are among the most reliable methods used. However, KF and EKF require the storage of, and operations with, matrices of size  $n \times n$ , where  $n$  is the size of the state space. Furthermore, both methods include inversion operations for  $m \times m$  matrices, where  $m$  is the size of the observation space. Thus Kalman filter methods become impractical as the dimension of the system increases. In this paper, we introduce a Variational Kalman filter (VKF) method to provide a low storage, and computationally efficient, approximation of the KF and EKF methods. Furthermore, we introduce a Variational Kalman smoother (VKS) method to approximate the fixed-lag Kalman smoother (FLKS) method. Instead of using the KF formulae, we solve the underlying maximum a posteriori optimization problem using the limited memory BFGS (LBFGS) method. Moreover, the LBFGS optimization method is used to obtain a low storage approximation of state estimate covariances and prediction error covariances. A detailed description of the VKF and VKS methods with LBFGS is given. The methodology is tested on linear and nonlinear test examples. The simulated results of the VKF method are presented and compared to KF and EKF, respectively. The convergence of BFGS/LBFGS methods is tested and demonstrated numerically.

KEY WORDS: Kalman filter, Bayesian inversion, large-scale optimization

### 1. Introduction

Several variants of the Kalman filter (KF) and the extended Kalman filter (EKF) have been proposed to reduce their computational complexity for large-dimensional problems. The Reduced Rank Kalman Filter or Reduced Order extended Kalman filter (see, e.g., [23], [5], [30], [8], [11], [24], [28]) project the dynamical state vector of the model onto a low dimensional subspace. The success of the approach depends on a judicious choice of the reduction operator. Moreover, since the reduction operator

---

\*Correspondence to: H. Auvinen, Department of Mathematics and Physics, Lappeenranta University of Technology, email: harri.auvinen@lut.fi

is typically fixed in time, the dynamics of the system may not be correctly captured; see [9] for more details.

There exist various Ensemble Kalman Filter (EnKF) algorithms—first proposed in [7]—that are widely used in the field of data assimilation. The idea behind these methods is to form an ensemble of state vectors that represent the state estimate covariance. Each of the members of the ensemble is then propagated forward in time by the full nonlinear evolution model in order to approximate component-wise covariances of prediction error. EnKF can be used on large-scale data assimilation problems because it is highly parallelizable.

In [1], we have shown how high dimensional KF and EKF may be carried out approximatively using the limited memory BFGS (LBFGS) optimization algorithm. The resulting methods were effective and exhibited low storage and computational cost characteristics. In this paper, we introduce an alternative approximation, the Variational Kalman filter (VKF), for KF and EKF. Furthermore, we introduce a Variational Kalman smoother (VKS) method to approximate the fixed-lag Kalman smoother (FLKS) method. In the variational approach, we solve an equivalent maximum a posteriori optimization problem using LBFGS, which replaces the explicit computation and use of the Kalman gain matrix, in order to obtain state estimates and covariance approximations.

The idea of using the LBFGS method in variational data assimilation is not new (see e.g. [25], [10], [26], [27] [28], [11], [24]). In many of these references, the LBFGS Hessian or inverse Hessian is used as a preconditioner, and even as an approximate error covariance matrix for the background term in variational data assimilation. However, in the VKF method presented here, the LBFGS method is further used for matrix inversion, in order to propagate effectively the state estimate covariance information forward in time.

The paper is organized as follows. We introduce the notations used in this paper in Section 2, with discussion of the Kalman filter. In Section 3, we present our methods for approximating the Kalman filter and fixed-lag Kalman smoother. The convergence of LBFGS/BFGS methods are studied numerically in Section 4. To test and compare these methods, we present results from a number of numerical experiments in Section 5, and we end with discussion and conclusions in Sections 6 and 7, respectively.

## 2. The Kalman Filter

Consider the following coupled system of discrete, linear, stochastic difference equations

$$\mathbf{x}_k = \mathbf{M}_k \mathbf{x}_{k-1} + \boldsymbol{\varepsilon}_k^p, \quad (1)$$

$$\mathbf{y}_k = \mathbf{K}_k \mathbf{x}_k + \boldsymbol{\varepsilon}_k^o. \quad (2)$$

In the first equation,  $\mathbf{x}_k$  denotes the  $n \times 1$  state vector of the system at time  $k$ ;  $\mathbf{M}_k$  is the  $n \times n$  linear evolution operator; and  $\boldsymbol{\varepsilon}_k^p$  is a  $n \times 1$  random vector representing the prediction error and is assumed to characterize errors in the model and in the corresponding numerical approximations. In the second equation,  $\mathbf{y}_k$  denotes the  $m \times 1$  observed data vector;  $\mathbf{K}_k$  is the  $m \times n$  linear observation operator; and  $\boldsymbol{\varepsilon}_k^o$  is an  $m \times 1$  random vector representing the observation error. The error terms are

assumed to be independent and Normally distributed, with zero mean and with covariance matrixes  $\mathbf{C}_{\varepsilon_k^p}$  and  $\mathbf{C}_{\varepsilon_k^o}$ , respectively.

The task is to estimate  $\mathbf{x}_k$  and its error covariance  $\mathbf{C}_k$  at time point  $k$  given  $\mathbf{y}_k$ ,  $\mathbf{K}_k$ ,  $\varepsilon_k^o$ ,  $\mathbf{M}_k$ ,  $\varepsilon_k^p$  and estimates  $\mathbf{x}_{k-1}^{est}$  and  $\mathbf{C}_{k-1}^{est}$  of the state and covariance at time point  $k - 1$ . The Kalman filter is the standard approach taken for such problems. It has the form.

**The Kalman filter algorithm**

- Step 0:** Select initial guess  $\mathbf{x}_0^{est}$  and covariance  $\mathbf{C}_0^{est}$ , and set  $k = 1$ .
- Step 1:** Compute the evolution model estimate and covariance:
  - (i) Compute  $\mathbf{x}_k^p = \mathbf{M}_k \mathbf{x}_{k-1}^{est}$ ;
  - (ii) Compute  $\mathbf{C}_k^p = \mathbf{M}_k \mathbf{C}_{k-1}^{est} \mathbf{M}_k^T + \mathbf{C}_{\varepsilon_k^p}$ .
- Step 2:** Compute Kalman filter estimate and covariance:
  - (i) Compute the Kalman Gain  $\mathbf{G}_k = \mathbf{C}_k^p \mathbf{K}_k^T (\mathbf{K}_k \mathbf{C}_k^p \mathbf{K}_k^T + \mathbf{C}_{\varepsilon_k^o})^{-1}$ ;
  - (ii) Compute the Kalman filter estimate  $\mathbf{x}_k^{est} = \mathbf{x}_k^p + \mathbf{G}_k (\mathbf{y}_k - \mathbf{K}_k \mathbf{x}_k^p)$ ;
  - (iii) Compute the estimate covariance  $\mathbf{C}_k^{est} = \mathbf{C}_k^p - \mathbf{G}_k \mathbf{K}_k \mathbf{C}_k^p$ .
- Step 3:** Update  $k := k + 1$  and return to Step 1.

A nonlinear extension of KF, known as the extended Kalman filter (EKF), is obtained when (1), (2) are replaced by

$$\mathbf{x}_k = \mathcal{M}(\mathbf{x}_{k-1}) + \varepsilon_k^p, \tag{3}$$

$$\mathbf{y}_k = \mathcal{K}(\mathbf{x}_k) + \varepsilon_k^o, \tag{4}$$

where  $\mathcal{M}$  and  $\mathcal{K}$  are possibly nonlinear functions. EKF is obtained by the following modification of the KF algorithm: in Step 1, (i) use the nonlinear model  $\mathbf{x}_k^p = \mathcal{M}(\mathbf{x}_k^{est})$  to compute the prior, but employ the linearized approximations,

$$\mathbf{M}_k = \frac{\partial \mathcal{M}(\mathbf{x}_{k-1}^{est})}{\partial \mathbf{x}}, \quad \text{and} \quad \mathbf{K}_k = \frac{\partial \mathcal{K}(\mathbf{x}_k^p)}{\partial \mathbf{x}}, \tag{5}$$

for the covariance calculations, and otherwise employ the same formulae as above.

We note that  $\mathbf{M}_k$  and  $\mathbf{K}_k$  can be computed or estimated in a number of ways. For example, the numerical scheme that is used in the solution of either the evolution or the observation model defines a tangent linear code (see, e.g., [15]), which can be used to compute (5). A common, but also more computationally expensive, approach is to use finite differences to approximate (5).

The Kalman filter is expensive to implement due to the fact that it is necessary to store  $n \times n$  matrices and invert  $m \times m$  matrices at each step. Our task is to overcome these limitations. For this we will need a variational formulation of the Kalman filter that is set forth in the following section. We make the (reasonable) assumptions that multiplication by the evolution and observation matrices  $\mathbf{M}_k$  and  $\mathbf{K}_k$  and by the covariance matrices  $\mathbf{C}_{\varepsilon_k^p}$  and  $\mathbf{C}_{\varepsilon_k^o}$  is efficient, both in terms of storage and CPU time.

3. The Variational Kalman filter method

In a general assimilation problem, where  $n$  is the size of state space and  $m$  is the size of the observation space, the standard formulation of the Kalman filter (KF)

and the extended Kalman filter (EKF) require the storage and multiplication of  $n \times n$  matrices and the inversion of  $m \times m$  matrices. Thus the computational cost of KF/EKF increases rapidly as the size of the problem becomes large. Due to this fact there are several interesting assimilation problems, where the standard formulation of KF or EKF is impractical to implement.

Bayes' Theorem can be used to formulate the Kalman filter as a sequential maximum a posterior iteration. To see this, we recall Bayes' formula

$$p_{\mathbf{x}|\mathbf{y}}(\mathbf{x}) = \frac{p_{\mathbf{y}|\mathbf{x}}(\mathbf{y}|\mathbf{x})p_{\mathbf{x}}(\mathbf{x})}{p_{\mathbf{y}}(\mathbf{y})}, \quad (6)$$

where  $\mathbf{x}$  is the vector of unknowns,  $\mathbf{y}$  the measurements,  $p_{\mathbf{x}}$  denotes the prior density, and  $p_{\mathbf{y}|\mathbf{x}}$  is the density of the likelihood function. The maximum a posterior (MAP) estimate is obtained by maximizing (6). Equivalently, one can minimize

$$\ell(\mathbf{x}|\mathbf{y}) := -\log p_{\mathbf{y}|\mathbf{x}}(\mathbf{y}|\mathbf{x}) - \log p_{\mathbf{x}}(\mathbf{x}). \quad (7)$$

For the linear model (2) at time  $k$ , the function  $\ell$  assumes the form

$$\ell(\mathbf{x}|\mathbf{y}_k) = \frac{1}{2}(\mathbf{y}_k - \mathbf{K}_k\mathbf{x})^T \mathbf{C}_{\varepsilon_k^o}^{-1}(\mathbf{y}_k - \mathbf{K}_k\mathbf{x}) + \frac{1}{2}(\mathbf{x} - \mathbf{x}_k^p)^T (\mathbf{C}_k^p)^{-1}(\mathbf{x} - \mathbf{x}_k^p), \quad (8)$$

where  $\mathbf{C}_{\varepsilon_k^o}$  and  $\mathbf{C}_k^p$  are the covariance matrices of the measurement noise  $\varepsilon_k^o$  and of the prior  $\mathbf{x}_k^p$ , respectively. The Kalman filter estimate and its covariance  $\mathbf{x}_k^{est}$  and  $\mathbf{C}_k^{est}$  are precisely the minimizer and inverse Hessian of  $\ell(\mathbf{x}|\mathbf{y}_k)$ , respectively.

The advantage of the variational formulation of the Kalman filter is that it suggests the use of an optimization algorithm for computing estimates of  $\mathbf{x}_k^{est}$  and  $\mathbf{C}_k^{est}$ . For large-scale problems, this can be very advantageous.

In particular, we advocate using the limited memory BFGS algorithm (LBFGS) for the minimization problem (7). Given specific choices of initial guess, stopping criteria, and number of stored vectors, LBFGS will yield estimates of both  $\mathbf{x}_k^{est}$  and  $\mathbf{C}_k^{est}$ . The storage requirement for the covariance approximation—which we denote  $\mathbf{B}_k^\#$ —is  $2rn$ , where  $r$  is the number of stored LBFGS vectors (typically on the order of 10), and multiplication by  $\mathbf{B}_k$  is order  $n$ . See the Appendix for the details of the LBFGS algorithm for a quadratic cost function, as well as the limited memory formulas for  $\mathbf{B}_k^\#$  and  $(\mathbf{B}_k^\#)^{-1}$ .

However, it is  $(\mathbf{C}_k^p)^{-1}$  that is needed in the optimization problem in the next VKF iteration (see (8)). Thus we apply LBFGS a second time to an auxiliary optimization problem

$$\arg \min_{\mathbf{u}} \frac{1}{2} \langle \mathbf{A}\mathbf{u}, \mathbf{u} \rangle - \langle \mathbf{b}, \mathbf{u} \rangle, \quad (9)$$

where  $\mathbf{A} = \mathbf{M}_k \mathbf{B}_{k-1}^\# \mathbf{M}_k^T + \mathbf{C}_{\varepsilon_k^p}$  and  $\mathbf{b}$  is the zero vector. This gives an approximation  $\mathbf{B}_k^*$  of  $(\mathbf{C}_k^p)^{-1}$ .

We summarize the Variational Kalman filter algorithm as follows:

### The Variational Kalman filter algorithm

**Step 0:** Select initial guess  $\mathbf{x}_0^\#$  and covariance  $\mathbf{B}_0^\# = \mathbf{C}_0^{est}$ , and set  $k = 1$ .

**Step 1:** Compute the evolution model estimate and covariance:

- (i) Compute  $\mathbf{x}_k^p = \mathbf{M}_k \mathbf{x}_{k-1}^\#$ ;
- (ii) Define  $\mathbf{C}_k^p = \mathbf{M}_k \mathbf{B}_{k-1}^\# \mathbf{M}_k^T + \mathbf{C}_{\varepsilon_k^p}$ ;
- (iii) Compute LBFGS approximation  $\mathbf{B}_k^*$  of  $(\mathbf{C}_k^p)^{-1}$ ;

**Step 2:** Compute Variational Kalman filter and covariance estimates:

- (i) Minimize  $\ell(\mathbf{x}|\mathbf{y}_k) = (\mathbf{y}_k - \mathbf{K}_k \mathbf{x})^T (\mathbf{C}_{\varepsilon_k^o})^{-1} (\mathbf{y}_k - \mathbf{K}_k \mathbf{x}) + (\mathbf{x} - \mathbf{x}_k^p)^T \mathbf{B}_k^* (\mathbf{x} - \mathbf{x}_k^p)$  using LBFGS, and define  $\mathbf{x}_k^\#$  and  $\mathbf{B}_k^\#$  to be the LBFGS minimizer and inverse Hessian approximations;

**Step 3:** Update  $k := k + 1$  and return to Step 1.

Note that in Step 1, (ii) and Step 2, (i) the optimizations are quadratic and therefore only quadratic LBFGS is needed, (see appendix for details). For practical applications, a judicious choice of the initial inverse Hessian is needed in order to obtain accurate results efficiently. In the numerical examples of this work, we have used  $\mathbf{B}_0^{-1} = \beta \mathbf{I}$  with  $\beta$  chosen so that  $\beta \mathbf{I}$  approximates the diagonal of the covariance matrix of interest. For more discussion on the choice of  $\mathbf{B}_0^{-1}$  (the preconditioner) see [20], [28].

### 3.1. The nonlinear Variational Kalman filter method

As in the case of EKF, we need a linearization to propagate the covariance information from one observation time to the next. However, the direct linearization as in EKF is impractical for large dimensions. Rather, in the case of a non-linear evolution model we should use the adjoint operator, if available, in Step 1, (ii) of VKF. Furthermore, if the adjoint operator is coded in an implicit form, *i.e.* in the software of the model, we get full benefit from a limited memory presentation of  $\mathbf{C}_k^{est}$ . This, indeed, is the situation in many operational codes for weather forecasting.

Supposing that the linearization  $\mathbf{M}_k$  of  $\mathcal{M}_k$  is available, then Step 2, (i) can be written in the VKF algorithm as:

**Step 2:** Compute the Variational Kalman filter estimate and covariance:

- (i) Minimize  $\ell(\mathbf{x}|\mathbf{y}_k) = (\mathbf{y}_k - \mathcal{K}(\mathbf{x}))^T (\mathbf{C}_{\varepsilon_k^o})^{-1} (\mathbf{y}_k - \mathcal{K}(\mathbf{x})) + (\mathbf{x} - \mathbf{x}_k^p)^T \mathbf{B}_k^* (\mathbf{x} - \mathbf{x}_k^p)$  using LBFGS, and define  $\mathbf{x}_k^\#$  and  $\mathbf{B}_k^\#$  to be the LBFGS minimizer and inverse Hessian approximations;

Especially, if the *tangent linear*  $\mathbf{M}_k^{tl}$  and corresponding *adjoint code*  $\mathbf{M}_k^*$  [15] are available for the evolution model  $\mathcal{M}$ , Step 1, (ii) can be written as:

**Step 1:** Compute the evolution model estimate and covariance:

- (ii) Define  $\mathbf{C}_k^p = \mathbf{M}_k^{tl} \mathbf{B}_{k-1}^\# \mathbf{M}_k^* + \mathbf{C}_{\varepsilon_k^p}$ ;

This feature of the method is one of the major advantages compared to EKF, since the time consuming linearization of the evolution model can be avoided. The traditional linearization requires  $n$  evolution model function calls, but inside VKF the required number of tangent linear and adjoint code evaluations is around 15-60, which is far less than  $n$  in large scale problems.

Similar possibilities exist for use the tangent linear code  $\mathbf{K}_k^{tl}$  of the observation model  $\mathcal{K}$  in Step 2, (i):

**Step 2:** Compute the evolution model estimate and covariance:

- (i) Minimize  $\ell(\mathbf{x}|\mathbf{y}_k) = (\mathbf{y}_k - \mathbf{K}_k^{tl} \mathbf{x})^T (\mathbf{C}_{\varepsilon_k^o})^{-1} (\mathbf{y}_k - \mathbf{K}_k^{tl} \mathbf{x}) + (\mathbf{x} - \mathbf{x}_k^p)^T \mathbf{B}_k^* (\mathbf{x} - \mathbf{x}_k^p)$  using LBFGS, and define  $\mathbf{x}_k^\#$  and  $\mathbf{B}_k^\#$  to be the LBFGS minimizer and inverse Hessian approximations;

In EKF the computational advantage of using  $\mathbf{K}_k^{tl}$  instead is lost, since the Kalman gain is typically a full matrix.

### 3.2. The Variational Kalman smoother method

Next we introduce a Variational Kalman smoother (VKS) method, which can be used afterwards to smooth the results of the Variational Kalman filter. The idea is to simulate a fixed-lag Kalman smoother (FLKS) method and take full benefit from the limited memory covariance approximation form of the VKF method. In general, such post-processing improves the quality of the VKF results.

The VKF method provides an estimate  $\mathbf{x}_k^{est}$  and a corresponding limited memory approximation of the covariance matrix  $\mathbf{C}_k^{est}$  after each time step  $k$ . In VKS, we use these results from the previous  $[k_0, k_0+1, \dots, k]$  time steps, where the parameter  $k_0 = k - lag$  determines the length of the time interval. In case of linear evolution model  $\mathbf{M}_k$ , we couple the results together by using the following cost function:

$$J(\mathbf{x}_{k_0}) = \sum_{t=k_0}^k (\mathbf{M}_t \mathbf{x}_{k_0} - \mathbf{x}_t^{est})^T (\mathbf{C}_t^{est})^{-1} (\mathbf{M}_t \mathbf{x}_{k_0} - \mathbf{x}_t^{est}), \quad (10)$$

where  $\mathbf{M}_t \mathbf{x}_{k_0}$  is a model trajectory from  $\mathbf{x}_{k_0}$ . The minimization of the cost function is done by using the 4d-Var method (see [6], [15]), using LBFGS.

In the nonlinear case, the evolution model  $\mathcal{M}_t$  is used instead of  $\mathbf{M}_t$  in the cost function formulation (10). Furthermore, the gradient of (10) can be computed efficiently by using the adjoint of the evolution model, but in principle, the linearization of  $\mathcal{M}_t$  can be used again as well. Since the smoothing process improves the accuracy of the estimate at time  $k_0$ , it is possible to outperform EKF in retrospective analysis.

During VKF iterations, the inverse Hessian limited memory BFGS formula is used to represent  $\mathbf{C}_k^{est}$ . In the VKS cost function (10) we instead need the inverse of  $\mathbf{C}_t^{est}$ . In practice this detail is handled by using the direct Hessian limited memory BFGS formula (see, e.g., [20] and the appendix). The direct Hessian limited memory BFGS formula provides the  $(\mathbf{C}_t^{est})^{-1}$  required.

## 4. Monitoring the quality of the BFGS approximation

The quality of approximations produced by Quasi-Newton methods to Hessian matrices, such as the covariance matrices in our case, has been studied at least since 1970 [12]. Normally this is carried out by monitoring the matrix norm of the difference between an approximation and a known Hessian matrix.

As discussed above, the Kalman filter is a statistical procedure, that repeatedly applies the Bayes rule to create the distribution of the state vector. We will therefore employ the chi-square test to monitor the goodness of the LBFGS updates as approximations of the covariances of known distributions. The chi-square distribution provides a scalar valued test for multinormality: if  $\mathbf{x}$  is an  $n$ -dimensional Gaussian random vector with zero mean and covariance  $\mathbf{C}$ , then  $\mathbf{x}^T \mathbf{C}^{-1} \mathbf{x} \sim \chi_n^2$ . We monitor how well the approximative covariances fulfill this test, when a sample of vectors  $\mathbf{x}$  has been generated using a 'true' covariance matrix.

We start with a covariance  $\mathbf{C}^{true}$  and sample a set of vectors  $\mathbf{x}_i$  from  $N(0, \mathbf{C}^{true})$ ,  $i = 1, \dots, m$ . Next, we compute the approximations  $\mathbf{C}_{iter}^{-1}$  of the inverse of the covariance using the LBFGS optimization with an increasing number of iterations, and calculate  $qttest(i) := \mathbf{x}_i \mathbf{C}_{iter}^{-1} \mathbf{x}_i'$ , for every  $i$ . These values are compared with the chi-square significance values at, e.g., the 0.5, 0.75 and 0.95 levels, to see

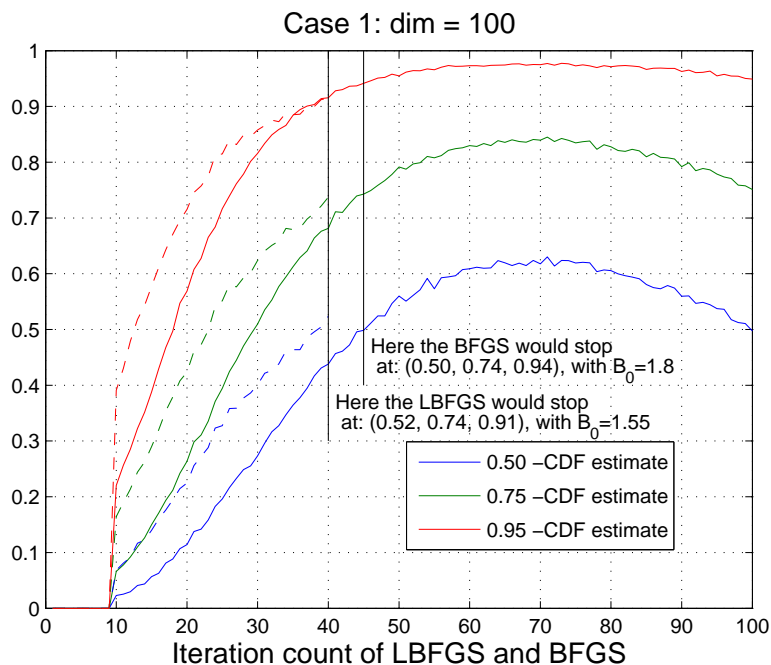


Figure 1. Plot of the chi-square test confidence limit estimates of the BFGS (-) and LBFGS (- -) as a function of the iteration count.

which percentage of the vectors sampled from the true distribution lie inside those confidence regions of the respective approximative multinormal distributions. It is known that for a linear-quadratic problem the BFGS algorithm forms an exact inverse Hessian with full  $n$  updates, so the test values are expected to reach the true limits with  $n$  iterations. Accordingly, we perform tests with both the limited memory BFGS and full BFGS, but with the LBFGS method we stop at the convergence limit, when the stopping criterion has been chosen to be close to machine precision. The full BFGS runs are performed with the LBFGS code, by keeping all search directions in the memory. During LBFGS runs we keep all other search directions in memory except the first one.

In the test examples, we use  $m = 1000$  samples and repeat the procedure 10 times to get the mean values of the results.

We consider two different cases: in the first example, the true covariance  $\mathbf{C}_i^{true} \in \mathbb{R}^{m \times m}$  is defined with singular values proportional to  $1/k$ , where  $k = 4, 5, \dots, m + 4$ . In the second test, the covariance is adopted from the EKF process of the Lorenz95 case (for more details, see the next Chapter), with 100 variables.

In the above tests we allow the BFGS optimization method to continue iteration beyond the normal stopping point. If the typical stopping criteria are used, the LBFGS method will terminate after about 45 iterations in these cases, with 100 variables. We also perform a true limited memory chi-square test with LBFGS, where we stop the optimization process at the convergence limit.

The results for the first test case are given in Figure 1. With too few updates, the calculated values are below the chi-square test values, so the approximate covariance gives a too narrow distribution. With increasing updates, the test values are first

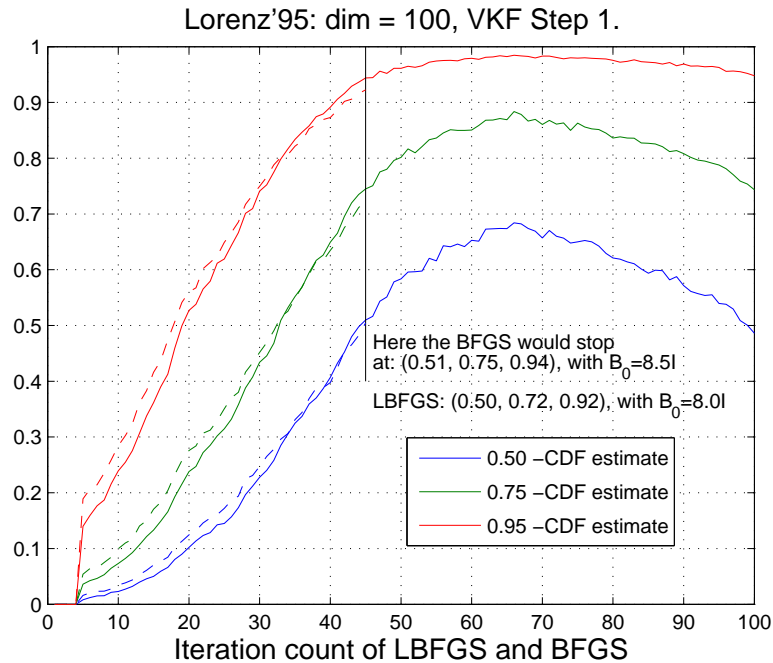


Figure 2. Plot of the chi-square test confidence limit estimates of the and BFGS (-) and LBFGS (- -) as a function of the iteration count.

exceeded, then we can see how the correct values are reached with full updates. The choice of the initial inverse matrix  $B_0$  naturally has an impact on the process. To demonstrate this, we use slightly different  $B_0$  values:  $B_0 = 1.8\mathbf{I}$ , with BFGS and  $B_0 = 1.55\mathbf{I}$ , with LBFGS.

Next we adopt the covariance  $\mathbf{C}^{est}$  from the EKF process in the Lorenz'95 case, with 100 variables. Then we perform the chi-square test over the VKF Step 1, (ii) where we approximate  $(\mathbf{C}^p)^{-1} = (\mathbf{M}\mathbf{C}^{est}\mathbf{M}^T + \mathbf{C}_\varepsilon)^{-1}$  by using BFGS and LBFGS methods. The corresponding results for the second test example, the Lorenz95 case, are plotted in Figure 2. The initial inverse Hessian  $B_0 = 8.5\mathbf{I}$ , with BFGS and  $B_0 = 8.0\mathbf{I}$ , with LBFGS.

We can see that BFGS method relative quickly finds the right confidence limits, with a suitable initial inverse Hessian. After the normal stopping point of the optimization process BFGS exceeds its designated confidence limits, but afterwards converges to the right values as the iteration count reaches the dimension of the problem. We note, that for practical applications the most important values are located near the normal stopping point of the optimization process, where the method seems to provide reasonably accurate results.

Next, we study the accuracy of the covariance approximation over one full VKF time step, i.e., both Step 1 and Step 2. For simplicity, we use the same number of iterations during both steps. Again, we take the covariance  $\mathbf{C}^{est}$  and also  $\mathbf{x}^p$  from the EKF process in the Lorenz'95 case, with 100 variables. First we perform VKF Step 1, (ii) where we approximate  $(\mathbf{C}^p)^{-1} = (\mathbf{M}\mathbf{C}^{est}\mathbf{M}^T + \mathbf{C}_\varepsilon)^{-1}$  by using LBFGS method. Then we use this approximation in VKF Step 2, (i) to compute the Variational Kalman filter covariance approximation with the LBFGS method using initial

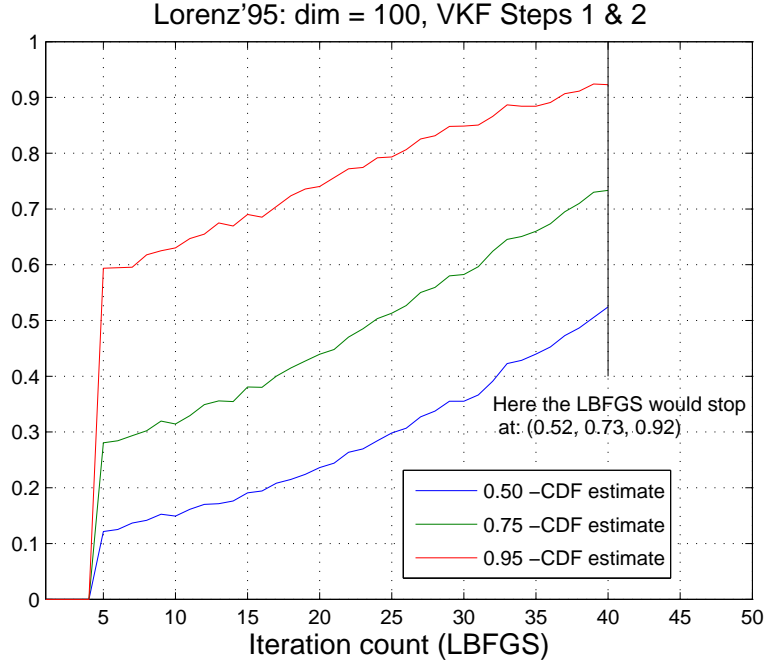


Figure 3. Plot of the chi-square test confidence limit estimates for LBFSGS (-) over the VKF steps. The LBFSGS estimates are plotted as a function of the iteration count.

inverse Hessian  $B_0 = 0.12\mathbf{I}$ . In order to analyze the accuracy of the approximated covariance  $\mathbf{C}_{iter}^{-1}$ , we compute the corresponding operations with the EKF method, namely EKF Steps 1, (ii), Step 2, (i) and Step 2, (iii) to obtain  $\mathbf{C}^{true}$ . Finally, we perform similar chi-square tests for these covariances as earlier. The results are shown in Figure 3. They indicate that the LBFSGS method provides reasonably accurate approximations after 40 iterations already. Please see the Appendix for analytical result about the convergence rate of LBFSGS.

## 5. Numerical Experiments

In this section, we test the variational Kalman filter on two examples.

### 5.1. An Example with a Large-Scale Linear Evolution Model

The first example is intended to test a large dimensional situation. We consider the following forced heat equation model

$$\frac{\partial x}{\partial t} = -\frac{\partial^2 x}{\partial u^2} - \frac{\partial^2 x}{\partial v^2} + \alpha \exp \left[ -\frac{(u-2/9)^2 + (v-2/9)^2}{\sigma^2} \right], \quad (11)$$

where  $x$  is a function of  $u$  and  $v$  over the domain  $\Omega = \{(u, v) \mid 0 \leq u, v \leq 1\}$  and  $\alpha \geq 0$ . We generate synthetic data using (11) with  $\alpha > 0$  and assume that the evolution model is given by (11) with  $\alpha = 0$ , which gives a model bias. The problem can be made arbitrarily large-scale via a sufficiently fine spatial discretization.

However, the well-behaved nature of solutions of (11) calls for further experiments with a different test case.

We discretize the model (11) using a uniform  $N \times N$  computational grid and the standard finite difference schemes of both the time and spatial derivatives. This gives the time stepping equation  $\mathbf{x}_{k+1} = \mathbf{M}\mathbf{x}_k + \mathbf{f}$ , where  $\mathbf{M} = \mathbf{I} - \Delta t \mathbf{L}$ . Here  $\mathbf{L}$  is given by the standard finite difference discretization of the two-dimensional Laplacian operator with homogeneous Dirichlet boundary conditions,  $\Delta t$  is chosen to guarantee stability, and  $\mathbf{f}$  is the constant vector determined by the evaluation of the forcing term in (11) at each of the points of the computational grid. We define  $\mathbf{K}_k = \mathbf{K}$  for all  $k$  in (2), where  $\mathbf{K}$  is the full weighting matrix, which has the following grid representation

$$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}.$$

Such an observation matrix could model, for example, an array of square heat sensors on the bottom of a metal plate that have dimension  $2/N \times 2/N$  with the edges aligned with the grid lines and equally spaced at  $n^2/64$  locations.

We first generate synthetic data using the stochastic equations

$$\mathbf{x}_{k+1} = \mathbf{M}\mathbf{x}_k + \mathbf{f} + N(\mathbf{0}, (0.5\sigma_{\text{ev}})^2 \mathbf{I}), \quad (12)$$

$$\mathbf{y}_{k+1} = \mathbf{K}\mathbf{x}_{k+1} + N(\mathbf{0}, (0.8\sigma_{\text{obs}})^2 \mathbf{I}), \quad (13)$$

with  $\alpha = 3/4$  in (11) and where  $\sigma_{\text{ev}}^2$  and  $\sigma_{\text{obs}}^2$  are chosen so that the signal to noise ratios, defined by  $\|\mathbf{x}_0\|^2/n^2\sigma_{\text{ev}}^2$  and  $\|\mathbf{K}\mathbf{x}_0\|^2/n^2\sigma_{\text{obs}}^2$  respectively, are both 50. The initial condition used for the data generation is

$$[\mathbf{x}_0]_{ij} = \exp[-((u_i - 1/2)^2 + (v_j - 1/2)^2)],$$

where  $(u_i, v_j)$  is the  $ij$ th grid point.

For the implementation of KF, we used the biased models

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{M}\mathbf{x}_k + N(\mathbf{0}, \sigma_{\text{ev}}^2 \mathbf{I}), \\ \mathbf{y}_{k+1} &= \mathbf{K}\mathbf{x}_{k+1} + N(\mathbf{0}, \sigma_{\text{obs}}^2 \mathbf{I}), \end{aligned}$$

with initial conditions  $\mathbf{x}_0 = \mathbf{0}$  and  $\mathbf{C}_0^{\text{est}} = 0.001\mathbf{I}$  in Step 0 of the filter. We compare the results obtained with VKF and KF, where  $n = 2^j$  with  $j$  taken to be the largest positive integer so that memory issues do not arise in the MATLAB implementation for the standard KF. For the computer on which the simulations were done (a laptop with 2G RAM memory and a 2.13 GHz processor) the largest such  $j$  was 5, making  $N = 32$ . We note that in our implementation of the LBFGS method, we have chosen to take only 10 LBFGS iterations with 9 saved vectors.

In our first test, the goal is only to show that the results obtained with VKF are comparable to those obtained with KF. For this purpose, we present a plot in Figure 4 of the relative error vector, which has  $k$ th component

$$[\mathbf{relative\_error}]_k := \frac{\|\mathbf{x}_k^{\text{est}} - \mathbf{x}_k\|}{\|\mathbf{x}_k\|},$$

for both VKF and the standard KF. We see that results obtained using the two approaches yield quite similar relative error curves. Inside the VKF method, the initial inverse Hessian parameters during approximation of  $\mathbf{C}_k^{\text{est}}$  and  $(\mathbf{C}_k^p)^{-1}$  were

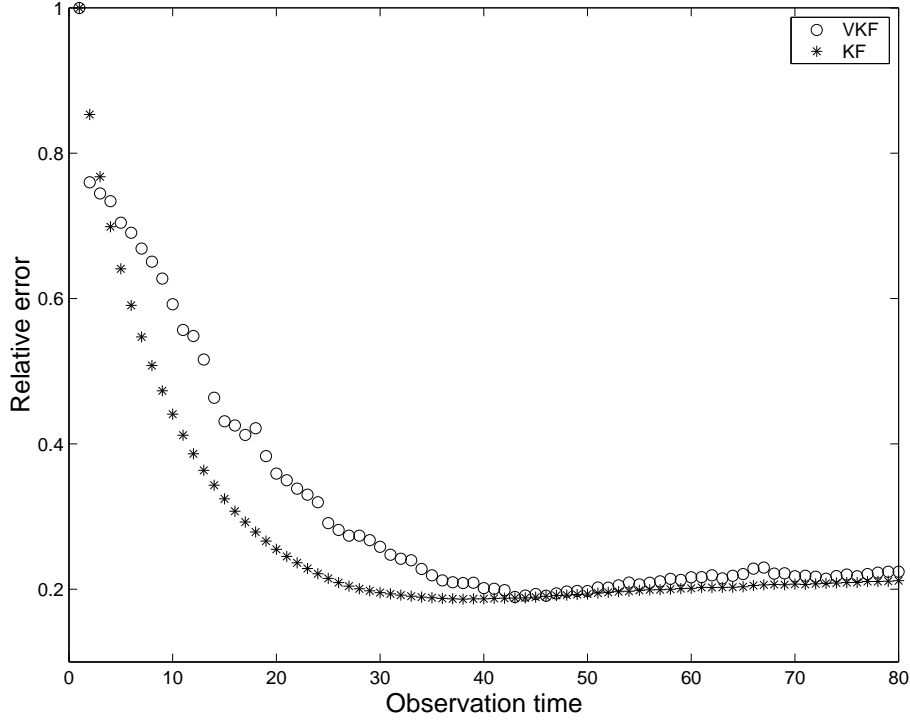


Figure 4. Relative error curves for KF (\*) and VKF (o). The horizontal axis represents the time of the observations.

$\mathbf{B}_0^{-1} = \mathbf{I}$  and  $\mathbf{B}_0^{-1} = 4000\mathbf{I}$ , respectively for all  $k$ . At the beginning of the filtering period KF provides more accurate results, but later the difference decreases. Both curves eventually begin to increase once the forcing term, which is not used in the state space model in KF, has a prominent effect on the data; in early iterations, it is overwhelmed by the diffused initial temperature. We also mention that in the large number of test runs we did using this model, our implementation of the VKF was on average about 10 times faster than the standard KF.

For more comparisons, we perform similar tests using different values for  $\sigma_{ev}^2$  and  $\sigma_{obs}^2$ , namely, so that the signal to noise ratios mentioned above are both 10. The same initial inverse Hessian parameters were used inside the VKF method as in the previous test. The relative error curves are exhibited in Figure 5. In this case, VKF provides better results at the beginning of the filtering period. This might be due to a regularization effect, implicitly implemented via the use of a truncated LBFGS algorithm.

Lastly, we define  $\sigma_{ev}^2$  and  $\sigma_{obs}^2$  to be as in the original experiment, but take  $\alpha = 2$ . This has the effect that the state space model used within VKF and KF is less accurate. In this case the initial inverse Hessian parameters were  $\mathbf{B}_0^{-1} = \mathbf{I}$  for  $\mathbf{C}_k^{est}$  and  $\mathbf{B}_0^{-1} = 2000\mathbf{I}$  for  $(\mathbf{C}_k^p)^{-1}$ . We now obtain the solution curves appearing in Figure 6. It seems that as the underlying evolution becomes less accurate, while the noise level remains moderately low, KF provides better results.

Satisfactory results can also be obtained for much larger scale problems. To show this, we take  $j = 8$  which gives  $N = 256$ ,  $n = 65536$ . So the number of unknowns in

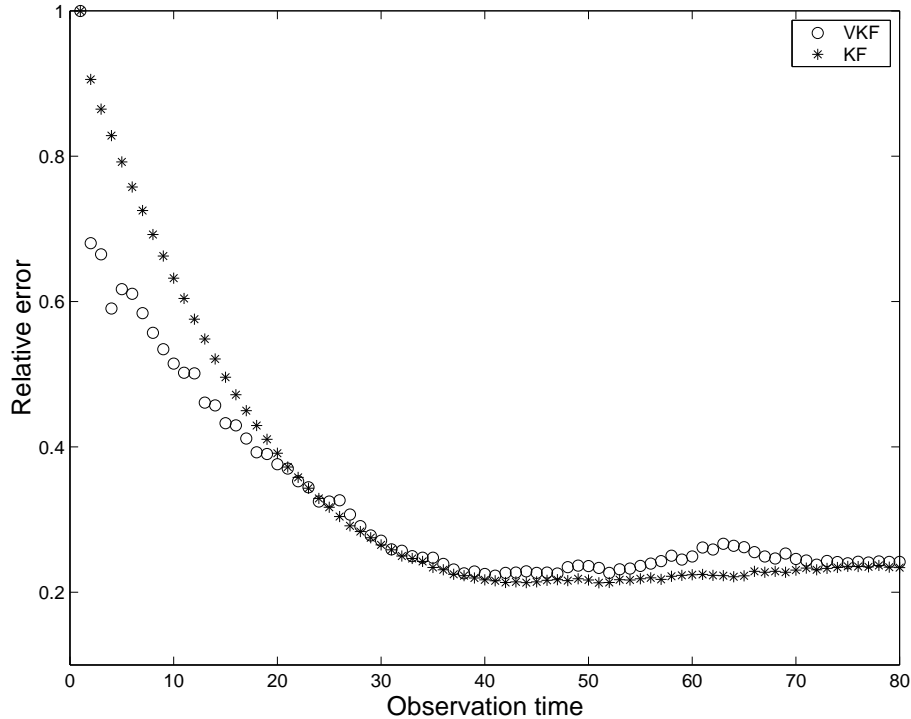


Figure 5. Relative error curves for KF (\*) and VKF (o). The horizontal axis represents the observation time.

this problem is then 65536. Otherwise we fix the parameter values to be the same as above. However, the stability condition of the time stepping scheme requires a much smaller step now. The respective error plot is given in Figure 7. We can not include any error curve for the standard KF because memory issues prevent it on our computer for both  $N = 128$  ( $n = 16384$ ) or  $N = 256$  ( $n = 65536$ ). However, in this case we compare results with the LBFGS-KF method [1]. We note that the results here for VKF are rather similar to those presented for LBFGS-KF in [1].

### 5.2. An Example with a Small-Scale, Nonlinear Evolution Model

Our second example produces chaotic, unpredictable behavior. We consider the non-linear Lorenz'95 model introduced and analyzed in [17, 18], given by

$$\frac{\partial x^i}{\partial t} = (x^{i+1} - x^{i-2})x^{i-1} - x^i + 8, \quad i = 1, 2, \dots, 40, \quad (14)$$

with periodic state space variables, i.e.  $x^{-1} = x^{n-1}$ ,  $x^0 = x^n$  and  $x^{n+1} = x^1$ . In the present tests we use the dimension  $n = 40$ . The model shares many characteristics with realistic atmospheric models (cf. [18]), and is often used as a test case for various weather forecasting schemes.

Next, we apply EKF, VKF and VKS to the problem of estimating the state variables from data generated using the nonlinear, chaotic evolution model (14). The data was generated by integrating the model using a fourth order Runge-Kutta (RK4) method with time-step  $\Delta t = 0.025$ . The discussion in [18] suggests

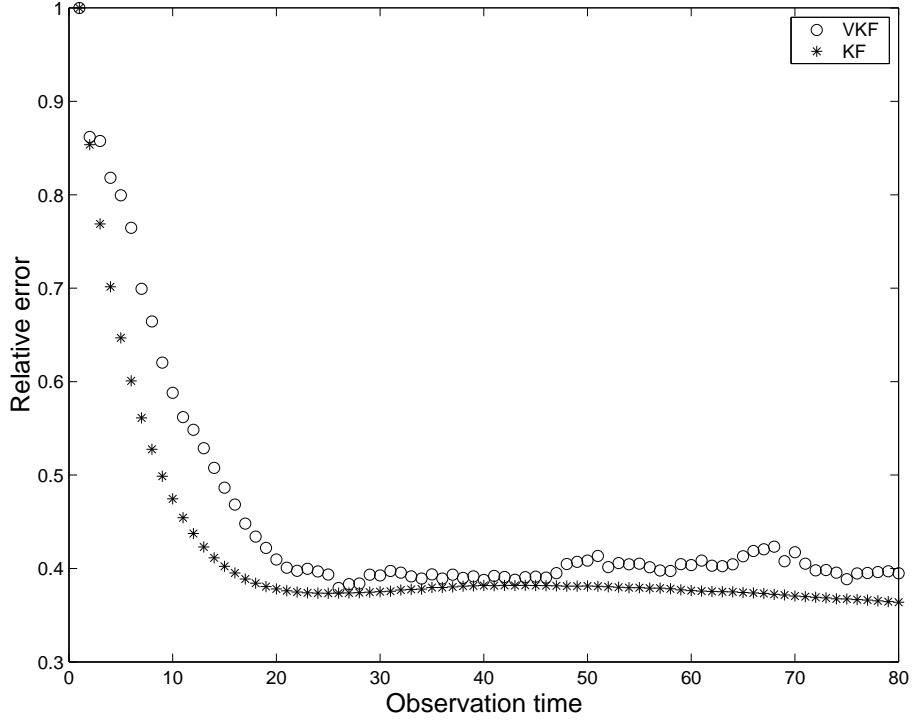


Figure 6. Relative error curves for KF (\*) and VKF (o). The horizontal axis represents the observation time.

that when using (14) as a test example for weather forecasting algorithms, the characteristic time scale is such that the above  $\Delta t$  corresponds to 3 hours. The “truth” was generated by taking 42920 time steps of the RK4 method, i.e., 5365 days. The initial state for the data generation was  $x^{20} = 8 + 0.008$  and  $x^i = 8$  for all  $i \neq 20$ .

The observed data is then computed using this true data. In particular, after a 365 day long initial period, the true data is observed at every other time step and at the last 3 grid points in each set of 5; that is, the observation matrix is  $m \times n$ , with nonzero entries

$$[\mathbf{K}]_{rs} = \begin{cases} 1 & (r, s) \in \{(3j + i, 5j + i + 2) \mid i = 1, 2, 3, j = 0, 1, \dots, 7\}, \\ 0 & \text{otherwise.} \end{cases}$$

The observation error is simulated using Gaussian noise  $N(\mathbf{0}, (0.15 \sigma_{\text{clim}})^2 \mathbf{I})$  where  $\sigma_{\text{clim}}$  is a standard deviation of the model state used in climatological simulations,  $\sigma_{\text{clim}} := 3.6414723$ . The data generation codes were written in MATLAB and were transcribed by us from the `scilab` codes written by the author of [16].

For the application of EKF and VKF, we employ the coupled system

$$\mathbf{x}_{k+1} = \mathcal{M}(\mathbf{x}_k) + N(\mathbf{0}, (0.05 \sigma_{\text{clim}})^2 \mathbf{I}), \tag{15}$$

$$\mathbf{y}_{k+1} = \mathbf{K} \mathbf{x}_{k+1} + N(\mathbf{0}, (0.15 \sigma_{\text{clim}})^2 \mathbf{I}), \tag{16}$$

where  $\mathcal{M}(\mathbf{x}_k)$  is obtained by taking two steps of the RK4 method applied to (14) from  $\mathbf{x}_k$  with time-step 0.025. We note that this coincides with the data generation

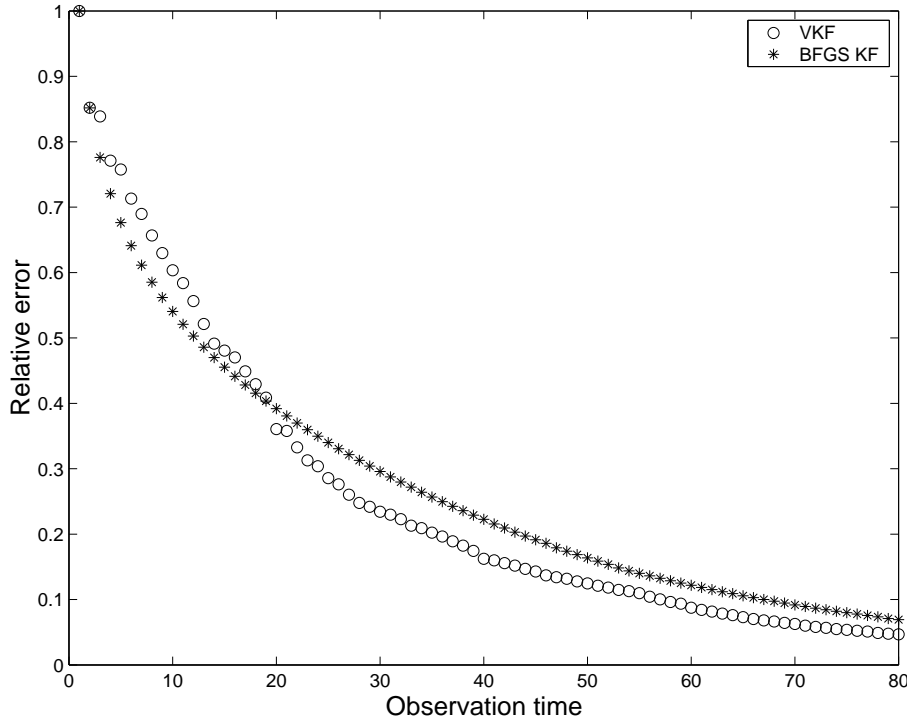


Figure 7. Relative error curves for LBFKS-KF (\*) and VKF (o). The horizontal axis represents the observation time.

scheme, if the noise term is removed and the above initial condition is used. Due to the fact that  $\mathcal{M}$  is a nonlinear function, EKF must be used (see equations (3) and (4)). Since  $\mathcal{K} := \mathbf{K}$  in (4) is linear,  $\mathbf{K}_k = \mathbf{K}$  for all  $k$  in (5). However a linearization of the nonlinear evolution function  $\mathcal{M}$  is required. The computation of  $\mathbf{M}_k$  in (5) is performed by a routine in one of the `scilab` codes mentioned above, adopted for our use in MATLAB.

The initial condition is defined by  $[\mathbf{x}_{t_0}]_i = [\mathbf{x}_{t_0}^{true}]_i + N(0, (0.3 \sigma_{\text{clim}})^2)$  for all  $i$ , and the initial covariance was taken to be  $\mathbf{C}_0^{est} = (0.13 \sigma_{\text{clim}})^2 \mathbf{I}$ . In our implementation of the LBFKS method within VKF, we computed 15 iterations with 14 saved vectors and the initial inverse Hessian parameters were  $\mathbf{B}_0^{-1} = 0.15 \mathbf{I}$  for  $\mathbf{C}_k^{est}$  and  $\mathbf{B}_0^{-1} = 10 \mathbf{I}$  for  $(\mathbf{C}_k^p)^{-1}$ .

To present the accuracy of the state estimates  $\mathbf{x}_k^{est}$  obtained by EKF, VKF and VKS we plot the vector with components

$$[\mathbf{rms}]_k = \sqrt{\frac{1}{40} \|\mathbf{x}_k^{est} - \mathbf{x}_k^{true}\|^2} \quad (17)$$

in Figure 8. We observe that all three methods yield comparable results.

In order to compare the forecasting abilities of the two approaches, we compute the following forecast statistics at every 8th observation. Take  $j \in \mathcal{I} := \{8i \mid i = 1, 2, \dots, 100\}$  and define

$$[\mathbf{forecast\_error}_j]_i = \frac{1}{40} \|\mathcal{M}_{4i}(\mathbf{x}_j^{est}) - \mathbf{x}_{j+4i}^{true}\|^2, \quad i = 1, \dots, 20, \quad (18)$$

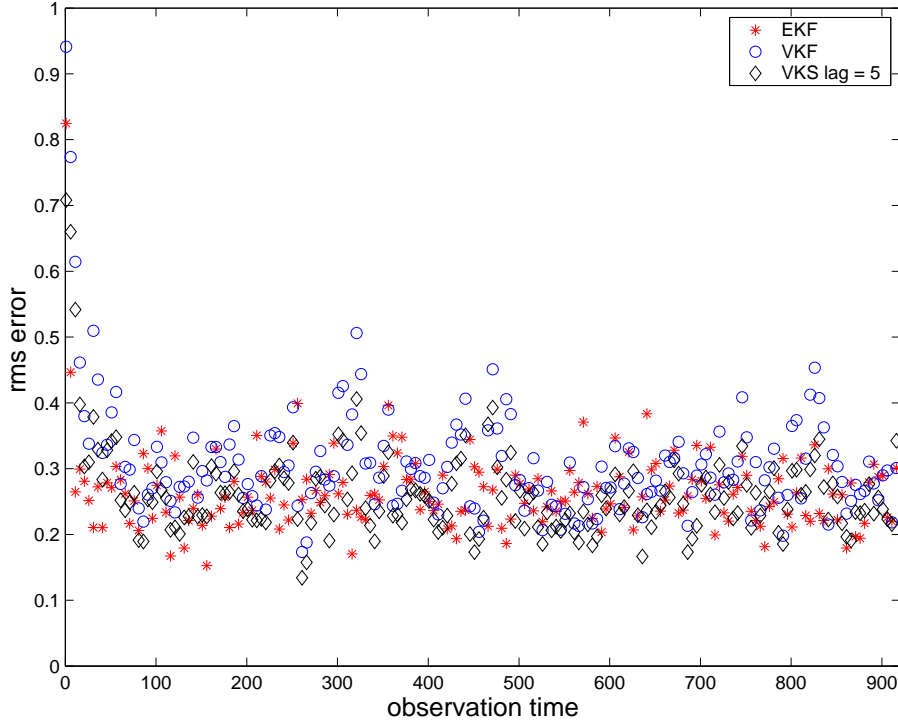


Figure 8. Plot of residual mean square error for VKF (o), EKF (\*) and VKS (◊) applied to (14).

where  $\mathcal{M}_n$  denotes a forward integration of the model by  $n$  time steps with the RK4 method. Thus this vector gives a measure of forecast accuracy given by the respective filter estimate up to 80 time steps, or 10 days out. We may define the forecast skill vector

$$[\text{forecast\_skill}]_i = \frac{1}{\sigma_{\text{clim}}} \sqrt{\frac{1}{100} \sum_{j \in \mathcal{I}} [\text{forecast\_error}_j]_i^2}, \quad i = 1, \dots, 20, \quad (19)$$

plotted in Figure 9. The results show that the forecasting skills of EKF and VKF are quite similar. This suggests that the quality of the VKF estimates is as high as those obtained using EKF. The forecast of VKS method is computed  $lag = 5$  observation times afterwards and therefore it actually is not a forecast, but it demonstrates the improvement of the accuracy of retrospective analysis. Figure 9 also illustrates the fact that the Lorenz 95 model (14) is truly chaotic.

In our test cases a linear or linearized model matrix  $\mathbf{M}_k$  has been available. This may not true in many important examples. But in numerical weather forecasting, for example, a tangent linear code often is available, providing an efficient way to compute the matrix vector product  $\mathbf{M}_k \mathbf{x}$ .

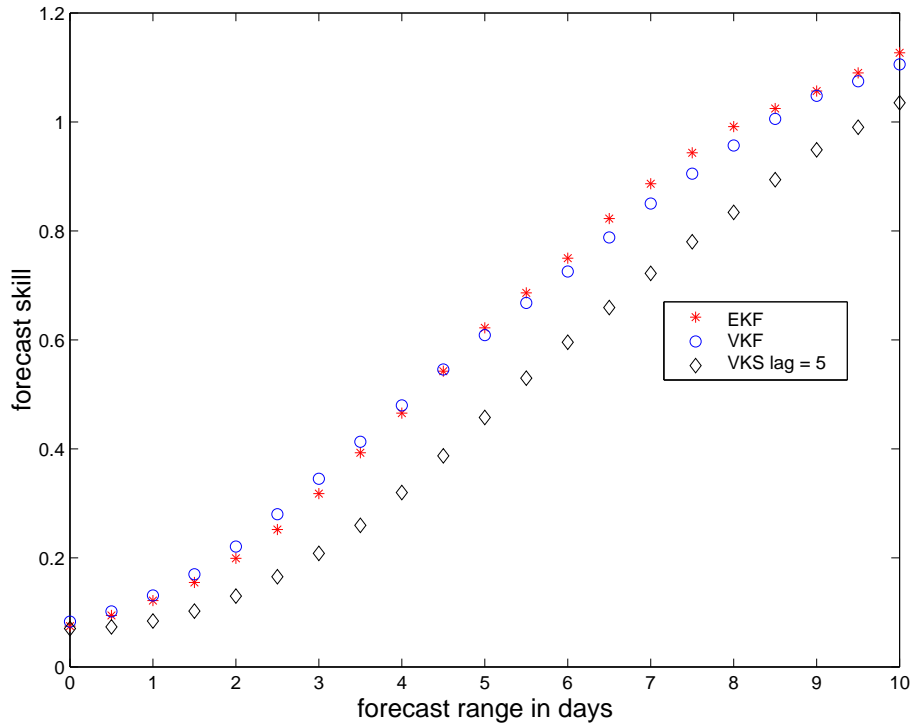


Figure 9. Plot of forecast skill vector for VKF (o) and EKF (\*) applied to (14). This plot also contains a retrospective forecast skill of the VKS ( $\diamond$ ) method.

## 6. Discussion

Fisher and Courtier [10] adopt several low-rank corrections derived from a 3D-Var minimization with LBFGS to approximate background and forecast error covariances, and also use them for preconditioning the minimizations. They observe that such methods perform reasonably well, but that in a 2D weather model, even 52 leading singular vectors fail to capture most of the difference between the variances of background and analysis error. Veersé et al. [28] adopt diagonal preconditioning based on LBFGS, too, and also apply it to the inverse Hessian approximation. They do not involve the Kalman formulas that would propagate the covariance in time.

VKF is related in many ways to various Reduced Rank Kalman filters, see [23], [8], or Reduced Order Kalman filters and Proper Orthogonal Decomposition (POD) filters, see [24]. Like these methods, VKF uses a low-dimensional subspace to approximate the forecast and analysis error covariance matrices.

Dee [23] and Fisher [8] select a suitable subspace for the low-rank correction beforehand and keep it over the assimilation window. Tian et al. [24] use a proper orthogonal decomposition (POD) of an ensemble for spanning the four-dimensional search basis of an ensemble method and save considerably in the dimension of the control space. They are able to carry out a minimization over a set of entire 4D model trajectories, but they keep the same ensemble for the entire assimilation window, unlike VKF.

A popular alternative to variational data assimilation is the family of many

different Ensemble Kalman Filter algorithms, first proposed in [7]. Ensemble Kalman Filters avoid the need to use tangent linear and adjoint codes, and rely instead on generating an ensemble of state vectors that are then propagated with the full nonlinear evolution model forward in time.

Ensemble methods are relatively simple to implement, although many advanced versions use sophisticated manipulation to glean information about error covariance from the ensemble. Gejadze et al. [11] have implemented an ensemble method with a BFGS search direction and gradient basis. Their formulation of the problem is control theoretic, just like in 4D-Var, which renders the control problem to an initial state control, unlike the final state control formulation adopted by VKF and EKF alike. Their method is capable of handling strong nonlinearities in the model and produce very good approximate analysis error covariance matrices. VKF is even more similar to the Maximum Likelihood Ensemble Filter MLEF of Zupanski [32] in its principle of maximizing the likelihood in Step 2, (i) of VKF. MLEF, like VKF, follows the mode of the set of search directions, rather than the mean. However, unlike VKF, it uses a fixed set of search directions throughout the assimilation window.

Ensemble Kalman Filters like many Reduced Rank Kalman Filters tend to suffer from covariance leaks. This means that if the size of the sample, or the dimension of the reduced basis, is small compared to the dimension of the state vector, progressively less and less of true model error covariance stays in the subspace spanned by the ensemble, or the low-rank subspace, see [9]. The reasons to this phenomenon are not fully understood but it seems plausible that non-linear system dynamics simply do not leave any low-dimensional sub-space invariant, even if a basis of that subspace is transformed with the nonlinear model. VKF, on the other hand, compensates for this defect by discontinuously updating its subspace by picking up new 3D-Var and 4D-Var minimization directions every time there are new observations to process, or at any pre-defined time intervals.

The computational complexity of VKF closely parallels that of 4D-Var, and even that of Ensemble Kalman filters. In its Step 1, (i) VKF simply computes a short forecast with the full nonlinear model. In Step 1, (ii) an artificial 4D-Var minimization is carried out with fixed tangent linear and adjoint models. This is otherwise similar to the inner minimization loop of an incremental formulation of 4D-Var, see [21], except that the minimization is formulated as a final state control problem. It can therefore be expected to converge in some fifty iterations in an operational weather model. Since the approximate error covariance matrix is assembled using the LBFGS update formula in the course of the minimization, there is no extra cost compared to incremental 4D-Var, apart from conducting the minimization with the LBFGS method, as opposed to the method of Conjugate Gradients, which requires that a set of search directions and gradients is stored and manipulated.

In Step 2, the essence of VKF involves just a single 3D-Var minimization in Step 2, (i) again using LBFGS. We can therefore conclude that the computational complexity of VKF is similar to the classical 4D-Var method, see [15], where there is only a single minimization loop that uses LBFGS minimization, instead of the nested loop structure of incremental 4D-Var.

VKF also mimics the algorithmic structure of 4D-Var rather faithfully. It involves a 4D-Var type sequence of steps, including computing the tangent linear and adjoint models during every forecast step. Linearization does not need to be repeated,

since the time-dependent minimization is carried out with the linearized model by definition in Step 1, (ii). Repeated linearizations may be needed in Step 2, (i), but this will involve only the observation operator, as this step mimics 3D-Var and the forecast model is not involved. If the tangent linear and adjoint models are available because of a previously adopted 4D-Var, VKF will be relatively straightforward to implement.

Parallelization is an issue not discussed in the current paper, but please see [3] for some analysis. The formulation of VKF used here is a serial algorithm because of the sequential minimizations involved, although the forecast model can obviously be run in parallel, just as in current operational practice. In this respect, too, VKF is a close relative to current implementations of 4D-Var. However, it is possible to replace Step 1, (ii) by propagating 3D-Var minimization directions only forward in time, as described in [3]. Another possibility is to adopt an Ensemble version of VKF.

Some recent Ensemble Kalman methods have adopted a hybrid approach, where an Ensemble Kalman Filter is used to produce a dynamic error covariance matrix, while the state is transported with a 4D-Var method to retain its smoothness [31]. An earlier hybrid method combines a 3D-Var minimization with an Ensemble Kalman Filter [13]. In hybrid methods, the error covariance matrix is a weighted combination of an EnKF error covariance matrix and a static background error covariance matrix. In such a case, the ensemble can be computed in parallel, but the variational assimilation remains sequential.

VKF can be seen as a hybrid method, too, but a deterministic one. It possesses characteristics very similar to hybrid EnKF methods, and the parallel version discussed in [3] behaves computationally in a manner very close to the hybrid method of [31].

Hybrid methods involving 4D-Var will need a tangent linear and adjoint model, just like VKF. As to operational implementation, we feel that VKF is well worth implementing if the NWP model used has already been provided with a tangent linear and an adjoint code. In this case, the modifications needed to the analysis suite are mostly just some matrix and vector algebra. If 4D-Var has not been implemented, and the tangent linear and adjoint codes have not been produced, Ensemble Kalman filters will definitely be easier to implement and may better meet the needs of such NWP centers.

## 7. Conclusions

The standard implementations of KF and EKF become exceedingly time and memory consuming as the dimension of the underlying state space increases. Several variants of KF and EKF have been proposed to reduce the dimension of the system, thus making implementation in high dimensions possible. The Reduced Rank Kalman Filter or Reduced Order extended Kalman filter (see, e.g., [23], [5], [30], [8], [11], [24], [26], [27], [28]) project the dynamical state vector of the model onto a lower dimensional subspace. The success of this approach depends on a judicious choice of the reduction operator. Moreover, since the reduction operator is typically fixed in time, they can suffer from “covariance leaks” [9]. A typical cause to this is that a nonlinear system does not generally leave any fixed linear subspace invariant.

In this paper, we propose the use of the limited memory BFGS (LBFGS) minimization method in order to circumvent the computational complexity and memory issues of standard KF and EKF. In particular, we replace the  $n \times n$ , where  $n$  is the dimension of the state space, covariance matrices within KF and EKF with low storage approximations obtained using LBFGS. The large-scale matrix inversions required in KF and EKF implementations are also approximated using LBFGS. We call the resulting method the Variational Kalman filter (VKF). In order to test these methods, we consider two test cases: a large-scale linear and a small scale nonlinear one. The VKF is applied in the large-scale linear case and is shown to be effective. In fact, our method exceeds the speed of standard KF by an order of magnitude, and yields comparable results when both methods can be applied. Furthermore, it can be used on much larger scale problems. In the nonlinear, small scale case, VKF and VKS are implemented and are also shown to give results that are comparable to those obtained using standard EKF. We believe that these results suggest that our approach deserves further consideration for large-scale linear and non-linear data assimilation problems. We note that in truly high-dimensional non-linear cases we need a tangent linear and corresponding adjoint code of the evolution model in order to get full benefits of the VKF method. In many important application fields, for example in numerical weather forecasting, such codes already are available both for the evolution model and for the observation model.

## 8. Acknowledgements

This work was supported by the Academy of Finland (application number 213476, Finnish programme for Centre of Excellence in research 2006–2011, with Tekes funding decision 40084/06). The first author would like to thank Vilho, Yrjö and Kalle Väisälä Foundation for their support. We are thankful for M. Leutbecher for providing us with the Scilab version of the Lorenz95 code, that served as starting point for the Matlab model implementation. Finally, the second author would like to thank both the University of Montana and the University of Helsinki for their support during his stay in Finland, where this work was, in part, undertaken. We also thank our anonymous reviewers for suggestions that lead to the section on monitoring the quality of LBFGS approximations to the true error covariance matrix.

## 9. Appendix

For completeness, we present the LBFGS method for a quadratic minimization and the limited memory formulations for the Hessian and inverse Hessian matrices. The LBFGS Minimization Algorithm for  $q(\mathbf{u}) = \frac{1}{2}\langle \mathbf{A}\mathbf{u}, \mathbf{u} \rangle - \langle \mathbf{b}, \mathbf{u} \rangle$  reads as

```

ν := 0;
u0 := initial guess;
B0-1 := initial inverse Hessian approximation;
begin quasi-Newton iterations
  gν := ∇q(uν) = A uν - b;
  Bν-1 := LBFGS approximation to A-1;

```

$\mathbf{v}_\nu = \mathbf{B}_\nu^{-1} \mathbf{g}_\nu;$   
 $\tau_\nu = \langle \mathbf{g}_\nu, \mathbf{v}_\nu \rangle / \langle \mathbf{v}_\nu, \mathbf{A} \mathbf{v}_\nu \rangle;$   
 $\mathbf{u}_{\nu+1} := \mathbf{u}_\nu - \tau_\nu \mathbf{v}_\nu;$   
 end quasi-Newton iterations

### 9.1. The Limited Memory Approximation for $\mathbf{A}^{-1}$

The BFGS matrix  $\mathbf{B}_\nu^{-1}$  is computed using recursion

$$\mathbf{B}_{\nu+1}^{-1} = \mathbf{V}_\nu^T \mathbf{B}_\nu^{-1} \mathbf{V}_\nu + \rho_\nu \mathbf{s}_\nu \mathbf{s}_\nu^T,$$

where

$$\begin{aligned}
 \mathbf{s}_\nu &:= \mathbf{u}_{\nu+1} - \mathbf{u}_\nu, \\
 \mathbf{d}_\nu &:= \nabla q(\mathbf{u}_{\nu+1}) - \nabla q(\mathbf{u}_\nu), \\
 \rho_\nu &:= 1/\mathbf{d}_\nu^T \mathbf{s}_\nu, \\
 \mathbf{V}_\nu &:= \mathbf{I} - \rho_\nu \mathbf{d}_\nu \mathbf{s}_\nu^T.
 \end{aligned}$$

However, for large-scale problems the storage of the full matrix  $\mathbf{B}_\nu^{-1}$  is infeasible, which motivates the limited storage version of the algorithm. At iteration  $\nu$ , suppose that the  $j$  vector pairs  $\{\mathbf{s}_i, \mathbf{d}_i\}_{i=\nu-j}^{\nu-1}$  are stored. Then the LBFGS approximation of the inverse Hessian is given by

$$\begin{aligned}
 \mathbf{B}_\nu^{-1} &= (\mathbf{V}_{\nu-1}^T \cdots \mathbf{V}_{\nu-j}^T) \mathbf{B}_0^{-1} (\mathbf{V}_{\nu-j} \cdots \mathbf{V}_{\nu-1}) \\
 &+ \rho_{\nu-j} (\mathbf{V}_{\nu-1}^T \cdots \mathbf{V}_{\nu-j+1}^T) \mathbf{s}_{\nu-j} \mathbf{s}_{\nu-j}^T (\mathbf{V}_{\nu-j+1} \cdots \mathbf{V}_{\nu-1}) \\
 &+ \rho_{\nu-j+1} (\mathbf{V}_{\nu-1}^T \cdots \mathbf{V}_{\nu-j+2}^T) \mathbf{s}_{\nu-j+1} \mathbf{s}_{\nu-j+1}^T (\mathbf{V}_{\nu-j+2} \cdots \mathbf{V}_{\nu-1}) \\
 &+ \vdots \\
 &+ \rho_{\nu-1} \mathbf{s}_{\nu-1} \mathbf{s}_{\nu-1}^T.
 \end{aligned} \tag{20}$$

Assuming exact arithmetic and that  $j = n$ , we have that  $\mathbf{u}_\nu$  converges to the unique minimizer of  $q$  in at most  $n$  iterations, and if  $n$  iterations are performed  $\mathbf{B}_{n+1}^{-1} = \mathbf{A}^{-1}$  [20]. In the implementation in this paper, however,  $j \ll n$  and LBFGS iterations are stopped once a prespecified maximum number of iterations or gradient norm stopping tolerance is reached.

It is proven in [19] that for quadratic minimization problems and exact line searches LBFGS is equivalent to preconditioned conjugate gradient with fixed preconditioner  $\mathbf{B}_0$ . Thus its convergence rate is given by [20]

$$\|\mathbf{u}_k - \mathbf{u}^*\|_{\tilde{\mathbf{A}}}^2 \leq \left( \frac{\lambda_{N-k} - \lambda_1}{\lambda_{N-k} + \lambda_1} \right)^2 \|\mathbf{u}_0 - \mathbf{u}^*\|_{\tilde{\mathbf{A}}}^2,$$

where  $\mathbf{u}^* = \mathbf{A}^{-1} \mathbf{b}$ ,  $\tilde{\mathbf{A}} = \mathbf{B}_0^{1/2} \mathbf{A} \mathbf{B}_0^{1/2}$  is  $N \times N$  with eigenvalues  $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N$ , and  $\|\mathbf{v}\|_{\tilde{\mathbf{A}}} = \mathbf{v}^T \tilde{\mathbf{A}} \mathbf{v}$ .

### 9.2. A Low Storage Approximation of $\mathbf{A}$

The required formulas are given in [4], and take the following form. Let

$$\mathbf{S}_\nu = [\mathbf{s}_{\nu-j}, \dots, \mathbf{s}_{\nu-1}], \quad \mathbf{D}_\nu = [\mathbf{d}_{\nu-j}, \dots, \mathbf{d}_{\nu-1}],$$

then

$$\mathbf{B}_\nu = \xi_\nu \mathbf{I} - [\xi_\nu \mathbf{S}_\nu \quad \mathbf{D}_\nu] \begin{bmatrix} \xi_\nu \mathbf{S}_\nu^T \mathbf{S}_\nu & \mathbf{L}_\nu \\ \mathbf{L}_\nu^T & -\mathbf{D}_\nu \end{bmatrix}^{-1} \begin{bmatrix} \xi_\nu \mathbf{S}_\nu^T \\ \mathbf{D}_\nu^T \end{bmatrix}, \tag{21}$$

where  $\mathbf{L}_\nu$  and  $\mathbf{D}_\nu$  are the  $j \times j$  matrices

$$(\mathbf{L}_\nu)_{i,j} = \begin{cases} \mathbf{s}_{\nu-j-1+i}^T \mathbf{d}_{\nu-j-1+j} & \text{if } i > j, \\ 0 & \text{otherwise.} \end{cases}$$

and

$$\mathbf{D}_\nu = \text{diag}(\mathbf{s}_{\nu-j}^T \mathbf{d}_{\nu-j}, \dots, \mathbf{s}_{\nu-1}^T \mathbf{d}_{\nu-1}).$$

We note that when  $\xi_\nu = 1$  for all  $\nu$  in (21), we have an exact equality between  $\mathbf{B}_\nu$  in (21) and (20). However, we have found that a more accurate Hessian approximation is obtained if, following [20], we use the scaling  $\xi_\nu = \mathbf{d}_{\nu-1}^T \mathbf{d}_{\nu-1} / \mathbf{s}_{\nu-1}^T \mathbf{d}_{\nu-1}$  instead.

We note that the middle matrix in (21) has size  $2j \times 2j$ , which is of reasonable size provided  $j$  is not too large, and its inversion can be carried out efficiently using a Cholesky factorization that exploits the structure of the matrix (for details see [4]).

#### REFERENCES

1. H. Auvinen, J. M. Bardsley, H. Haario, and T. Kauranne, *Large-Scale Kalman Filtering Using the Limited Memory BFGS Method*. Submitted 2008 to ETNA.
2. Harri Auvinen, Heikki Haario and Tuomo Kauranne, *Optimal approximation of Kalman filtering with a temporally local 4D-Var in operational weather forecasting*, Proceedings of the 11th ECMWF Workshop on Use of High-Performance Computing in Meteorology (W. Zwiefelhofer, G. Mozdzyński (eds.)), World Scientific 2005.
3. Harri Auvinen, Heikki Haario and Tuomo Kauranne, *Variational Kalman Filtering on Parallel Computers*, Proceedings of the 12th ECMWF Workshop on Use of High-Performance Computing in Meteorology (W. Zwiefelhofer, G. Mozdzyński (eds.)), World Scientific 2007.
4. Richard H. Byrd, Jorge Nocedal, and Robert B. Schnabel, *Representations of Quasi-Newton Matrices and Their Use in Limited Memory Methods*, Mathematical Programming, Volume 63, Numbers 1-3, January, 1994, pp. 129-156.
5. Cane M.A., R.N. Miller, B. Tang, E.C. Hackert and A.J. Busalacchi, *Mapping tropical Pacific sea level: data assimilation via reduced state kalman filter*. J. Geophys. Res., Vol. 101, pp. 599-617, 1996.
6. J. Derber: A variational continuous assimilation technique. *Mon. Weather Rev.* **117** 2437-2446 (1989).
7. G. Evensen, *Sequential data assimilation with a non-linear quasi-geostrophic model using Monte Carlo methods to forecast error statistics*. J. Geophys. Res., Vol. 99, pp. 10143-10162, 1994.
8. M. Fisher, *Development of a simplified Kalman filter*. ECMWF Technical Memorandum 260, 1998.
9. Michael Fisher and Erik Andersson, *Developments in 4D-Var and Kalman Filtering*, ECMWF Technical Memorandum 347, 2001.
10. Michael Fisher and Philippe Courtier, *Estimating the covariance matrices of analysis and forecast error in variational data assimilation*, ECMWF Technical Memorandum 220, 1995.
11. I. Yu. Gejadze, F.-X. Le Dimet and V. Shutyaev, *On analysis error covariances in variational data assimilation*. SIAM J. Sci. Comput. 30, pp. 1847-1874, 2008.
12. J. Greenstadt, *Variations on Variable-Metric Methods*. Math. Comput. 24, pp. 1-22, 1970.
13. T. M. Hamill and C. Snyder, *A hybrid ensemble Kalman filter-3D variational analysis scheme*. Mon. Weather Rev. 128, pp. 2905-2919, 2000.
14. R. E. Kalman, *A New Approach to Linear Filtering and Prediction Problems*, Transactions of the ASME – Journal of Basic Engineering, 82 (Series D), 1960, pp. 35-45.
15. F.-X. LeDimet and O. Talagrand: Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects. *Tellus A series*, Vol. 38, pp. 97-110 (1986).
16. Martin Leutbecher, *A data assimilation tutorial based on the Lorenz-95 system*, European Centre for Medium-Range Weather Forecasts Web Tutorial, [www.ecmwf.int/newsevents/training/lecture\\_notes/pdf\\_files/ASSIM/Tutorial.pdf](http://www.ecmwf.int/newsevents/training/lecture_notes/pdf_files/ASSIM/Tutorial.pdf)
17. E. N. Lorenz, *Predictability: A problem partly solved*, Proc. Seminar on Predictability, Vol. 1, ECMWF, Reading, Berkshire, UK, pp. 1-18, 1996.
18. E. N. Lorenz and K. A. Emanuel, *Optimal Sites for Supplementary Weather Observations: Simulation with a Small Model*, Journal of Atmospheric Science, 1998, pp. 399-414.

19. Jorge Nocedal, *Updating Quasi-Newton Matrices with Limited Storage*, Mathematics of Computation, Volume 35, No. 151, 1980, pp. 773-782.
20. Jorge Nocedal and Stephen Wright, *Numerical Optimization*, Springer 1999.
21. Rabier, F., Järvinen, H., Klinker, E., Mahfouh, J.-F. and Simmons, A., *The ECMWF operational implementation of four dimensional variational assimilation. Part I: Experimental results with simplified physics*. Q. J. R. Meteorol. Soc. 126, pp. 1143-1170, 2000.
22. Clive D. Rodgers, *Inverse Problems for atmospheric sounding: Theory and Practice*, World Scientific, 2000.
23. Dee D.P., *Simplification of the Kalman filter for meteorological data assimilation* Quart. J. Roy. Meteor. Soc., vol. 117, pp. 365-384, 1990.
24. Tian, X., Xie, Z. and Dai, A., An ensemble-based explicit four-dimensional variational assimilation method. J. Geophys. Res., Vol. 113, pp. D21124 (1-13), 2008.
25. Yang W., Navon I. M., Courtier P., *A new Hessian preconditioning method applied to variational data assimilation experiments using NASA general circulation models*. Mon. Weather Rev. 124, pp. 1000-1017, 1996.
26. Veersé, F., *Variable-storage quasi-Newton operators as inverse forecast/analysis error covariance matrices in variational data assimilation*. INRIA Report No 3685, April 26, 1999.
27. Veersé, F., *Variable-storage quasi-Newton operators for modeling error covariances*. In Proceedings of the Third WMO International Symposium on Assimilation of Observations in Meteorology and Oceanography, 7-11 June 1999, Quebec City, Canada, World Meteorological Organization, Geneva.
28. Veersé, F., Auroux, D. and Fisher, M., *Limited-memory BFGS diagonal preconditioners for a data assimilation problem in meteorology*. Optimization and Engineering, 1, pp. 323-339, 2000.
29. Curtis R. Vogel, *Computational Methods for Inverse Problems*, SIAM 2002.
30. Voutilainen A., T. Pyhälähti, K. Kallio, J. Pulliainen, H. Haario, J. Kaipio) *A filtering Approach for Estimating Lake Water Quality from Remote Sensing data*. Int. J. Appl. Earth Observation & Geoinformation, ISSN 0303-2434, Vol 9, 1, 50-64, February 2007.
31. F. Zhang, M. Zhang and J. A. Hansen, *Coupling Ensemble Kalman Filter with Four-dimensional Data Assimilation*. Advances in Atmospheric Sciences 26, pp. 1-8, 2009.
32. Zupanski, M., *Maximum likelihood ensemble filter: theoretical aspects*. Mon. Weather Rev. 133, pp. 1710-1726, 2005.