Electronic Transactions on Numerical Analysis.
Volume 35, pp. 217-233, 2009.

ISSN 1068-9613.

ETNA
Kent State University
http://etna.math.kent.edu

LARGE-SCALE KALMAN FILTERING USING THE LIMITED MEMORY BFGS METHOD*

H. AUVINEN†, J. M. BARDSLEY‡, H. HAARIO†, AND T. KAURANNE†

Abstract. The standard formulations of the Kalman filter (KF) and extended Kalman filter (EKF) require the storage and multiplication of matrices of size $n \times n$, where n is the size of the state space, and the inversion of matrices of size $m \times m$, where m is the size of the observation space. Thus when both m and n are large, implementation issues arise. In this paper, we advocate the use of the limited memory BFGS method (LBFGS) to address these issues. A detailed description of how to use LBFGS within both the KF and EKF methods is given. The methodology is then tested on two examples: the first is large-scale and linear, and the second is small scale and nonlinear. Our results indicate that the resulting methods, which we will denote LBFGS-KF and LBFGS-EKF, yield results that are comparable with those obtained using KF and EKF, respectively, and can be used on much larger scale problems.

Key words. Kalman filter, Bayesian estimation, large-scale optimization

AMS subject classifications. 65K10, 15A29

1. Introduction.

The Kalman filter (KF) for linear dynamical systems and the extended Kalman filter (EKF) for nonlinear but smoothly evolving dynamical systems are popular methods for use on state space estimation problems. As the dimension of the state space becomes very large, as is the case, for example, in numerical weather forecasting, the standard formulations of KF and EKF become computationally intractable due to matrix storage and inversion requirements.

Computationally efficient variants of KF and EKF have been proposed for use on such large-scale problems. The Reduced Rank Kalman Filter or Reduced Order extended Kalman filter (see, e.g., [4, 7, 30]) project the dynamical state vector of the model onto a lower dimensional subspace. The success of the approach depends upon a judicious choice of the reduction operator. Moreover, since the reduction operator is typically fixed in time, the dynamics of the system may not be correctly captured; see [9] for more details.

In the context of numerical weather forecasting, a great deal of attention has been given to the filtering problem. The current state of the art is 4D-Var (see, e.g., [9, 23]), which utilizes a variational formulation of an initial value estimation problem [11, 14, 17]. 4D-Var has been shown to be identical to a Kalman smoother when the model is assumed to be perfect [16]. The resulting quadratic minimization problem is very large-scale (10^4-10^7 unknowns) and so efficient numerical optimization methods are needed. Similar to the methods in the previous paragraph, the partial orthogonal decomposition is used in [5] to reduce the dimensionality of the 4D-Var minimization problem. A more standard approach is to implement a preconditioned conjugate gradient method [8, 12, 22, 26]. In this context, a number of different preconditioners have been tested.

In this paper, we take a different approach. In particular, we focus our attention on the Kalman filter itself, using the limited memory BFGS (LBFGS) [22] iterative method for the required large-scale matrix storage and inversion within KF and EKF. More specifically, suppose $\mathbf{Ax} = \mathbf{b}$ is a system, with symmetric positive definite matrix $\mathbf{A}$, that requires solution,

*Received April 11, 2008. Accepted for publication August 29, 2009. Published online on November 30, 2009. Recommended by P. Van Dooren. This work was supported by a faculty exchange grant from the University of Montana.

†Department of Mathematics and Physics, Lappeenranta University of Technology, Lappeenranta, Finland ({harri.auvinen,heikki.haario,tuomo.kauranne}@lut.fi).

‡Department of Mathematical Sciences, University of Montana, Missoula, Montana 59812, USA (bardsleyj@mso.umt.edu).

and or a low storage approximation of $\mathbf{A}^{-1}$ is needed, within the KF or the EKF algorithms. In such cases, LBFGS is a natural choice, since it generates both a sequence of approximations of $\mathbf{A}^{-1}\mathbf{b}$, and a sequence of symmetric positive definite matrices $\{\mathbf{B}_k^{-1}\}$ approximating $\mathbf{A}^{-1}$. This choice is further supported by the result that $\mathbf{A}^{-1}\mathbf{b}$ is reached within a finite number of LBFGS iterations (assuming exact arithmetic) [20, 21], and that if k and full memory BFGS is used $\mathbf{B}_{n+1}^{-1} = \mathbf{A}^{-1}$ [22, Chapter 8]. A result regarding the accuracy of the LBFGS Hessian approximations to $\mathbf{A}^{-1}$ is also given in [21]; it depends upon the eigenvalues of $\mathbf{A}^{1/2}\mathbf{B}^k\mathbf{A}^{1/2}$; see Appendix A. The use of LBFGS and its success in the examples that we consider can be motivated by the fact that the covariance matrices being approximated are approximately low rank. In applications of interest [6], covariance information is contained in slowly varying, low dimensional subspaces, making accurate low-rank approximations possible.

The idea of using the LBFGS method in variational data assimilation is not new; see, e.g., [10, 13, 17, 25, 27, 28, 29, 31]. In many of these references, the LBFGS Hessian or inverse Hessian is used as a preconditioner for conjugate gradient iterations, and even as an approximate error covariance matrix for the background term in 3D- and 4D-Var variational data assimilation. However, in the 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. Moreover, we apply our methodology to the Kalman filter itself, not to the variation formulation used by the 3D- and 4D-Var methods [17]. LBFGS can also be incorporated in a fully variational formulation of the Kalman filter; see [2].

As has been stated, the approach presented here uses the LBFGS algorithm directly within the context of the Kalman filter. The equivalence of LBFGS and a certain preconditioned conjugate gradient method (see [21] and Appendix A) suggests that our approach and those cited above are similar. One advantage of the cited approaches, however, is that they can be incorporated into existing 3D- and 4D-Var codes used in practice.

An application of a similar methodology that could be used in conjunction with 3D- and 4D-Var is presented in [1, 2]. The aim of the current paper is to demonstrate the use of LBFGS within the standard (non-variational) formulation of the linear or extended Kalman filter.

The paper is organized as follows. We present KF and EKF in Section 2, and then in Section 3 we present LBFGS-KF and LBFGS-EKF. We test these methods with two numerical experiments in Section 4. Conclusions are then given in Section 5, and implementation details of the LBFGS algorithm are contained in Appendix A.

2. The Kalman filter. We consider the coupled system of discrete, linear stochastic difference equations given by

$$\mathbf{x}_k = \mathbf{M}_k\mathbf{x}_{k-1} + \varepsilon_k^p, \tag{2.1}$$
$$\mathbf{y}_k = \mathbf{K}_k\mathbf{x}_k + \varepsilon_k^o. \tag{2.2}$$

In the first equation, $\mathbf{x}_k$ denotes the $n \times 1$ state of the system at time k; $\mathbf{M}_k$ is the $n \times n$ linear evolution operator; and ε_k^p is a $n \times 1$ random vector known as the prediction error and is assumed to characterize errors in the model and corresponding numerical approximations. In the second equation, $\mathbf{y}_k$ denotes the $m \times 1$ observed data; $\mathbf{K}_k$ is the $m \times n$ linear observation operator; and ε_k^o is an $m \times 1$ random vector known as the observation error. The prediction error ε_k^p and observation error ε_k^o are assumed to be independent and normally distributed, with zero means and symmetric positive definite covariance matrices $\mathbf{C}_{\varepsilon_k^p}$ and $\mathbf{C}_{\varepsilon_k^o}$, respectively.

We assume, in addition, that we have in hand estimates of both the state $\mathbf{x}_{k-1}^{est}$ and its positive definite covariance matrix $\mathbf{C}_{k-1}^{est}$ at time $k-1$. Moreover, we assume that $\mathbf{x}_{k-1}^{est}$, ε_k^p, and ε_k^o are independent random vectors. The goal is to then estimate $\mathbf{x}_k^{est}$ and its covariance $\mathbf{C}_k^{est}$. In Bayesian terms, equation (2.2) provides the likelihood function in the estimation step, while (2.1) gives the prior. By the assumptions on ε_k^p and ε_k^o, the prior mean and the prior covariance matrix are directly obtained from (2.1),

$$\begin{aligned}\mathbf{x}_k^p &= \mathbf{M}_k \mathbf{x}_{k-1}^{est},\\ \mathbf{C}_k^p &= \mathbf{M}_k \mathbf{C}_{k-1}^{est} \mathbf{M}_k^{\mathrm{T}} + \mathbf{C}_{\varepsilon_k^p}.\end{aligned}$$

The full negative-log posterior density given (up to an additive constant) by Bayes' theorem, takes the form

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

and hence, we have

$$\mathbf{x}_k^{est} = \arg\min_{\mathbf{x}} \ell(\mathbf{x}|\mathbf{y}_k), \tag{2.3}$$

$$\mathbf{C}_k^{est} = \nabla^2 \ell(\mathbf{x}|\mathbf{y}_k)^{-1}. \tag{2.4}$$

Equations (2.3) and (2.4) motivate the variational Kalman filter, which is the subject of [1]. However, they can also be used to derive the Kalman filter. In particular, noting that (2.3) and (2.4) can be alternatively written (see [24] for detail)

$$\begin{aligned}\mathbf{x}_k^{est} &= \mathbf{x}_k^p + \mathbf{G}_k(\mathbf{y}_k - \mathbf{K}_k\mathbf{x}_k^p),\\ \mathbf{C}_k^{est} &= \mathbf{C}_k^p - \mathbf{G}_k\mathbf{K}_k\mathbf{C}_k^p,\end{aligned}$$

where

$$\mathbf{G}_k = \mathbf{C}_k^p\mathbf{K}_k^{\mathrm{T}}(\mathbf{K}_k\mathbf{C}_k^p\mathbf{K}_k^{\mathrm{T}} + \mathbf{C}_{\varepsilon_k^o})^{-1},$$

we have the following standard formulation of the Kalman filter.

The Kalman Filter

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

Note that it is typical to take the initial covariance $\mathbf{C}_k^{est}$ to be diagonal.

The extended Kalman filter (EKF) is the extension of KF when (2.1), (2.2) are replaced by

$$\mathbf{x}_k = \mathcal{M}(\mathbf{x}_{k-1}) + \varepsilon_k^p, \tag{2.5}$$

$$\mathbf{y}_k = \mathcal{K}(\mathbf{x}_k) + \varepsilon_k^o, \tag{2.6}$$

where $\mathcal{M}$ and $\mathcal{K}$ are (possibly) nonlinear functions. EKF is obtained by the following simple modification of the above algorithm: in Step 1 (i), use instead $\mathbf{x}^p = \mathcal{M}(\mathbf{x}_k^{est})$, and define

$$\mathbf{M}_k = \frac{\partial\mathcal{M}(\mathbf{x}_{k-1}^{est})}{\partial\mathbf{x}}, \quad \text{and} \quad \mathbf{K}_k = \frac{\partial\mathcal{K}(\mathbf{x}^p)}{\partial\mathbf{x}}. \tag{2.7}$$

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, which can be used to compute (2.7); see, e.g., [12, 15]. However, a more common, but also more computationally expensive, approach is to use finite differences to approximate (2.7).

3. Using LBFGS for large-scale Kalman filtering. When the model size n is large, the Kalman filter is known to be prohibitively expensive to implement. This motivates several alternative approaches—most notably the 4D-Var method—used in large-scale applications such as numerical weather forecasting [8, 9, 11, 12, 17, 23, 26] and oceanography [4]. We instead focus our attention on the Kalman filter itself, using the limited memory BFGS (LBFGS) [22] iterative method for the required large-scale matrix storage and inversion within KF and EKF.

First, we give a general description of the LBFGS method for minimizing

$$q(\mathbf{u}) = \frac{1}{2}\langle \mathbf{A}\mathbf{u}, \mathbf{u}\rangle - \langle \mathbf{b}, \mathbf{u}\rangle, \tag{3.1}$$

where $\mathbf{A}$ is an $n \times n$ symmetric positive definite matrix and $\mathbf{b}$ is an $n \times 1$ vector. It is given by

The LBFGS method for quadratic minimization
$\nu := 0$;
$\mathbf{u}_0 :=$ initial guess;
$\mathbf{B}_0^{-1} :=$ initial inverse Hessian approximation;
begin quasi-Newton iterations
  $\mathbf{g}_\nu := \nabla q(\mathbf{u}_\nu) = \mathbf{A}\mathbf{u}_\nu - \mathbf{b}$;
  $\mathbf{v}_\nu = \mathbf{B}_\nu^{-1}\mathbf{g}_\nu$;
  $\tau_\nu = \langle \mathbf{g}_\nu, \mathbf{v}_\nu\rangle / \langle \mathbf{v}_\nu, \mathbf{A}\mathbf{v}_\nu\rangle$;
  $\mathbf{u}_{\nu+1} := \mathbf{u}_\nu - \tau_\nu \mathbf{v}_\nu$;
  $\mathbf{B}_\nu^{-1} :=$ LBFGS approximation to $\mathbf{A}^{-1}$;
end quasi-Newton iterations

In all of the examples considered in this paper, $\mathbf{B}_0$ was taken to be the identity matrix. The limited memory formulations for $\mathbf{B}_\nu^{-1}$, and corresponding formulas for $\mathbf{B}_\nu$, are found in Appendix append1. The stopping criteria for the LBFGS iterations is discussed in Section 4.

Some insight into the convergence properties of the LBFGS method can be obtained by an appeal to its connection with the well-known conjugate gradient (CG) method, which described in detail in [20, 21, 22, Section 9.1]. In particular, CG can be formulated as a *memoryless* BFGS method. Moreover, in the presence of exact arithmetic, LBFGS and iterates from a certain preconditioned CG iteration are identical [21], and hence finite convergence is guaranteed. Thus it seems reasonable to suspect that LBFGS will have convergence properties similar to that of CG, which are well-known and have been extensively studied. In particular, the early convergence of CG iterates within the dominant subspaces corresponding to the largest eigenvalues of the coefficient matrix is likely shared by LBFGS iterates.

Next, we describe how LBFGS was used to make the Kalman filter more efficient. We make the reasonable assumption that multiplication by the evolution and observation matrices $\mathbf{M}_k$ and $\mathbf{K}_k$, and by the covariance matrices $\mathbf{C}_{\boldsymbol{\varepsilon}_k^p}$ and $\mathbf{C}_{\boldsymbol{\varepsilon}_k^o}$, is efficient, both in terms of storage and CPU time. Additional computational challenges arise for sufficiently large n due to the storage requirements for $\mathbf{C}_k^{est}$, which becomes a full matrix as the iterations proceed. The same is also true for $\mathbf{C}_k^p$. However, given that

$$\mathbf{C}_k^p = \mathbf{M}_k \mathbf{C}_k^{est} \mathbf{M}_k^{\mathrm{T}} + \mathbf{C}_{\boldsymbol{\varepsilon}_k^p}, \tag{3.2}$$

storage issues are restricted to those for $\mathbf{C}_k^{est}$; typically the matrix $\mathbf{C}_{\varepsilon_k^p}$ is assumed to be diagonal.

A low storage approximation of $\mathbf{C}_k^{est}$ can be obtained by applying the LBFGS algorithm to the problem of minimizing (3.1) with $\mathbf{A} = \mathbf{C}_k^{est}$ and $\mathbf{b} = \mathbf{0}$. The LBFGS matrix $\mathbf{B}_\nu^{-1}$ is then a low storage approximation of $(\mathbf{C}_k^{est})^{-1}$ and formulas for $\mathbf{C}_k^{est}$ from [3]—and also found in Appendix A.2—can be used.

Additionally, when m is sufficiently large the computation of $(\mathbf{K}_k\mathbf{C}_k^p\mathbf{K}_k^{\mathrm{T}}+\mathbf{C}_{\varepsilon_k^o})^{-1}(\mathbf{y}_k-\mathbf{K}_k\mathbf{x}_k^p)$ that is required in Step 2, (ii) of the Kalman filter iteration will be prohibitively expensive.

For the approximation of $(\mathbf{K}_k\mathbf{C}_k^p\mathbf{K}_k^{\mathrm{T}} + \mathbf{C}_{\varepsilon_k^o})^{-1}(\mathbf{y}_k - \mathbf{K}_k\mathbf{x}_k^p)$, we set $\mathbf{A} = \mathbf{K}_k\mathbf{C}_k^p\mathbf{K}_k^{\mathrm{T}} + \mathbf{C}_{\varepsilon_k^o}$ and $\mathbf{b} = \mathbf{y}_k - \mathbf{K}_k\mathbf{x}_k^p$ in (3.1) and apply LBFGS to the problem of minimizing (3.1).

The LBFGS Kalman filter method can now be presented.

The LBFGS Kalman Filter (LBFGS-KF)

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

Step 1: Compute the evolution model estimate and covariance:

(i) Compute $\mathbf{x}_k^p = \mathbf{M}_k\mathbf{x}_k^{est}$;

(ii) Define $\mathbf{C}_k^p = \mathbf{M}_k\mathbf{B}_\#\mathbf{M}_k^{\mathrm{T}} + \mathbf{C}_{\varepsilon_k^p}$.

Step 2: Compute the Kalman filter estimate and covariance:

(i) Define $\mathbf{A} = (\mathbf{K}_k\mathbf{C}_k^p\mathbf{K}_k^{\mathrm{T}}+\mathbf{C}_{\varepsilon_k^o})$ and $\mathbf{b} = \mathbf{y}_k-\mathbf{K}_k\mathbf{x}_k^p$ in (3.1) and compute the LBFGS approximations $\mathbf{B}_*$ of $\mathbf{A}^{-1}$ and $\mathbf{u}_*$ of $\mathbf{A}^{-1}\mathbf{b}$.

(ii) Compute the LBFGS-KF estimate $\mathbf{x}_{k+1}^{est} = \mathbf{x}_k^p + \mathbf{C}_k^p\mathbf{K}_k^{\mathrm{T}}\mathbf{u}_*$;

(iii) Define $\mathbf{A} = \mathbf{C}_k^p - \mathbf{C}_k^p\mathbf{K}_k^{\mathrm{T}}\mathbf{B}_*\mathbf{K}_k\mathbf{C}_k^p (\approx \mathbf{C}_{k+1}^{est})$ and $\mathbf{b} = \mathbf{0}$ in (3.1) and compute the LBFGS approximation $\mathbf{B}_\#$ of $\mathbf{C}_{k+1}^{est}$ using (A.2).

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

All operations with the $\mathbf{C}_k^{est}$ and $\mathbf{A}^{-1}$ are done using the LBFGS formulas; see Appendix A. As a result, LBFGS-KF is much less memory and computationally intensive than KF making its use on large-scale problems more feasible. Specifically, the storage requirements for the LBFGS estimate of $\mathbf{C}_k^{est}$ are on the order of $2n\ell + 4n$, where ℓ is the number of stored vectors in LBFGS (typically 10-20), rather than $n^2 + 4n$ [22, Section 9.1], and the computational cost for both obtaining and using this estimate is order n. Furthermore, the inversion of the $m \times m$ matrix $\mathbf{K}_k\mathbf{C}_k^p\mathbf{K}_k^{\mathrm{T}}+\mathbf{C}_{\varepsilon_k^o}$ is carried out in order m operations and its storage requirements are on the order of $2m\ell + 4m$ rather than $m^2 + 4m$ [22].

The accuracy of the LBFGS covariance approximations is an important question. An analysis addressing this question in the similar variational setting is performed in [2]. We believe that the results of that analysis should be similar for LBFGS-KF. Thus, we choose not to repeat it here.

In the first example considered in the numerical experiments, LBFGS-KF and KF are compared and it is noted that LBFGS-KF is roughly 10 times faster, in terms of CPU time, than KF when applied to the same problem. Moreover, using our MATLAB implementation, LBFGS-KF can be used on significantly larger-scale problems.

As we have mentioned, in our implementations of KF and LBFGS-KF, the covariance matrices $\mathbf{C}_{\varepsilon_k^p}$ and $\mathbf{C}_{\varepsilon_k^o}$ are taken to be diagonal. This is not a necessary requirement. More structured covariances can be used, containing important prior information [17], however in order to maintain the computational efficiency and low storage requirements of LBFGS-KF, $\mathbf{C}_{\varepsilon_k^p}$ and $\mathbf{C}_{\varepsilon_k^o}$ must be comparable to $\mathbf{M}_k$, $\mathbf{B}_\#$ and $\mathbf{K}_k$, $\mathbf{B}_*$, respectively, in terms storage requirements and the computational cost required for their multiplication.

In the next section, we test the algorithm on two examples. The first is large-scale and linear, while the second is small-scale and nonlinear. The purpose of these experiments is to demonstrate that LBFGS-KF and LBFGS-EKF are effective algorithms. We leave their comparison with other state of the art methods for approximate Kalman filtering [5, 8, 9, 11, 12, 14, 17, 23, 26] for a later paper.

4. Numerical experiments. In this section, we present numerical results that justify the use of LBFGS-KF. In particular, we apply the method to two examples. The first is sufficiently large-scale that the use of LBFGS-KF is justified. In particular, we assume the following forced heat equation evolution model

$$\frac{\partial x}{\partial t} = -\frac{\partial^2 x}{\partial u^2} - \frac{\partial^2 x}{\partial v^2} + \alpha \exp\left[-\frac{(u-2/9)^2 + (v-2/9)^2}{\sigma^2} \right], \tag{4.1}$$

where x is a function of u and v over the domain $\Omega = \{(u,v) \mid 0 \le u, v \le 1\}$ and $\alpha \ge 0$. In our experiment, we will generate synthetic data using (4.1) with $\alpha > 0$ and assume that the evolution model is given by (4.1) with $\alpha = 0$, which gives a model bias. The problem can be made as large-scale as one wants via the choice of a sufficiently fine discretization of the domain Ω.

However, the well-behaved nature of solutions of (4.1)—in particular the fact that its solutions tend to a steady state—makes further experiments with a different test case a necessity. For this reason, we also test our method on a second example, which contains chaotic solutions, and hence has unpredictable behavior. In particular, we consider the simple non-linear model introduced and analyzed in [18, 19] and which is given by

$$\frac{\partial x^i}{\partial t} = (x^{i+1} - x^{i-2})x^{i-1} - x^i + 8, \qquad i = 1, 2, \ldots, 40, \tag{4.2}$$

with periodic state space variables, i.e., $x^{-1} = x^{n-1}$, $x^0 = x^n$ and $x^{n+1} = x^1$, $n = 40$. Then (4.2) is a chaotic dynamical system (cf. [19]), which is desirable for testing purposes. As the model is computationally light and shares many characteristics with realistic atmospheric models (cf. [19]), it is commonly used for testing different data analysis schemes for weather forecasting.

4.1. An example with a large-scale linear evolution model. We perform our first experiments using model (4.1) using a uniform $N \times N$ computational grid and the standard finite difference discretization of both the time and spatial derivatives, which yields the following 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, Δt is chosen to guarantee stability, and $\mathbf{f}$ is the constant vector determined by the evaluation of the forcing term in (4.1) at each of the points of the computational grid.

We define $\mathbf{K}_k = \mathbf{K}$ for all k in (2.2), where $\mathbf{K}$ is a matrix modeling an array of square sensors on the computational grid. Assuming that each sensor collects a weighted average of the state values in a 3×3 pixel region centered at every 8^{th} pixel in both the x and y directions, $\mathbf{K}$ will have dimension $(n/64) \times n$. We assume, further, that the weighted average in the 3×3 region is defined by

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

In our first test, we generate synthetic data using the linear stochastic equations

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{M}\mathbf{x}_k + \mathbf{f} + N(\mathbf{0}, (0.5\sigma_{\text{ev}})^2 \mathbf{I}), \\ \mathbf{y}_{k+1} &= \mathbf{K}\mathbf{x}_{k+1} + N(\mathbf{0}, (0.8\sigma_{\text{obs}})^2 \mathbf{I}), \end{aligned}$$

with $\alpha = 3/4$ in (4.1) and where σ^2_{ev} and σ^2_{obs} are chosen so that the signal to noise ratios, defined by $\|\mathbf{x}_0\|^2/n^2\sigma^2_{\text{ev}}$ and $\|\mathbf{K}\mathbf{x}_0\|^2/n^2\sigma^2_{\text{obs}}$ respectively, are both 50. The initial condition used for the data generation was

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

where (u_i, v_j) is the ijth grid point.

In our implementation of KF, we used the biased models

$$\mathbf{x}_{k+1} = \mathbf{M}\mathbf{x}_k + N(\mathbf{0}, \sigma^2_{\text{ev}}\mathbf{I}), \tag{4.3}$$

$$\mathbf{y}_{k+1} = \mathbf{K}\mathbf{x}_{k+1} + N(\mathbf{0}, \sigma^2_{\text{obs}}\mathbf{I}), \tag{4.4}$$

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 the LBFGS-KF 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 1.8 GHz Core 2 Duo processor) the largest such j was 5, making $N = 32$ and $n = 1024$. We note that in our implementation of the LBFGS method within LBFGS-KF, we have chosen to take only 10 LBFGS iterations with 9 saved vectors. These choices may seem crude at first, however, more stringent stopping tolerances and/or a larger number of stored vectors did not appreciably affect the results for the examples that we considered.

The purpose of this test is to show that the results obtained with LBFGS-KF are comparable results with those obtained with KF. To do this, we present a plot in Figure 4.1 of the relative error vector, which has kth component

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

for both the LBFGS Kalman Filter and for the standard Kalman Filter. We see that results obtained using the two approaches yield similar, though not identical, relative error curves. 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 that we did using this large-scale model, our implementation of the LBFGS-KF was on average about 10 times faster than was the standard KF.

Additionally, in Figure 4.2, we present the filter estimates obtained from both KF and LBFGS-KF together with the true state values at time points 35 and 70. Note that in the early iterations of the filter, represented by time point 35, the filter does not detect the source because it is overwhelmed by the initial temperature and is not contained in the model (4.3), (4.4). However, the source is detected once the initial temperature has sufficiently dissipated.

For a thorough comparison, we perform the same test using values for σ^2_{ev} and σ^2_{obs} that yield signal-to-noise ratios of 10. The relative error curves in Figure 4.3 result. Interestingly, LBFGS-KF provides better results at the beginning of the filtering period than does KF. This can be explained, we believe, by the fact that a regularization of sorts is implicitly implemented via the use of a truncated LBFGS algorithm.

Finally, we choose σ^2_{ev} and σ^2_{obs} as in the original experiment, but take $\alpha = 2$, which has the effect of making the state space model that is used within LBFGS-KF and KF less accurate. When this is done, we obtain the solution curves appearing in Figure 4.4. Thus it seems that as the underlying evolution model becomes less accurate and the noise level remains moderately low KF provides better results

ETNA
Kent State University
http://etna.math.kent.edu

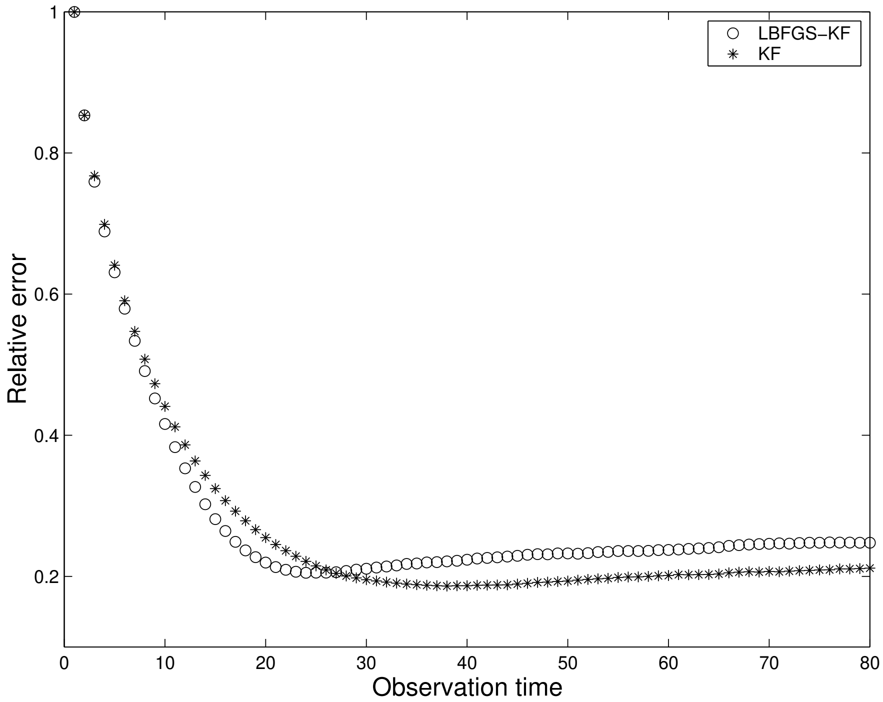

FIG. 4.1. *Relative error curves for KF (∗) and LBFGS-KF (o). The horizontal axis represents the observation time.*

In order to show that satisfactory results can also be obtained for much larger scale problems, we take $j = 8$ which gives $N = 256$ and $n = 65536$. We take all other parameter values to be those of the original experiment. Note, however, that the stability condition of the time stepping scheme requires a much smaller time step for this problem. A relative error plot similar to those in the previous example is given in Figure 4.5. We do not include an error curve for the Kalman filter because memory issues prevent its implementation on our computer for either $N = 128$ ($n = 16384$) or $N = 256$ ($n = 65536$).

The previous large-scale example remains orders of magnitude smaller than the typical size of systems considered in practical weather models. We stopped at $N = 256$ because our experiments were performed on a laptop that could not handle a larger-scale problem. However, the discussion of computational cost and storage in the paragraph following the description of the LBFGS-KF algorithm suggests that it scales well with problem size. Thus the use of LBFGS-KF on much larger-scale problems should be feasible. Efficiency can be further improved if several time steps are allowed in the forward model for each Kalman filter iteration, much as is done in 4D-Var implementations. In addition, to the degree that LBFGS is parallelizable, LBFGS-KF will also be parallelizable.

4.2. An example with a small-scale, nonlinear evolution model. In our next example, we apply EKF and LBFGS-EKF to the problem of estimating the state variables from data generated using the nonlinear, chaotic evolution model (4.2). To generate the data, a time integration of the model was first performed using a fourth order Runge-Kutta (RK4) method with time-step $\Delta t = 0.025$. Analysis in [19] suggests that when (4.2) is used as a test example for weather forecasting data assimilation algorithms, the characteristic time scale is such that the above Δt corresponds to 3 hours, which we will use in what follows. It is also noted in

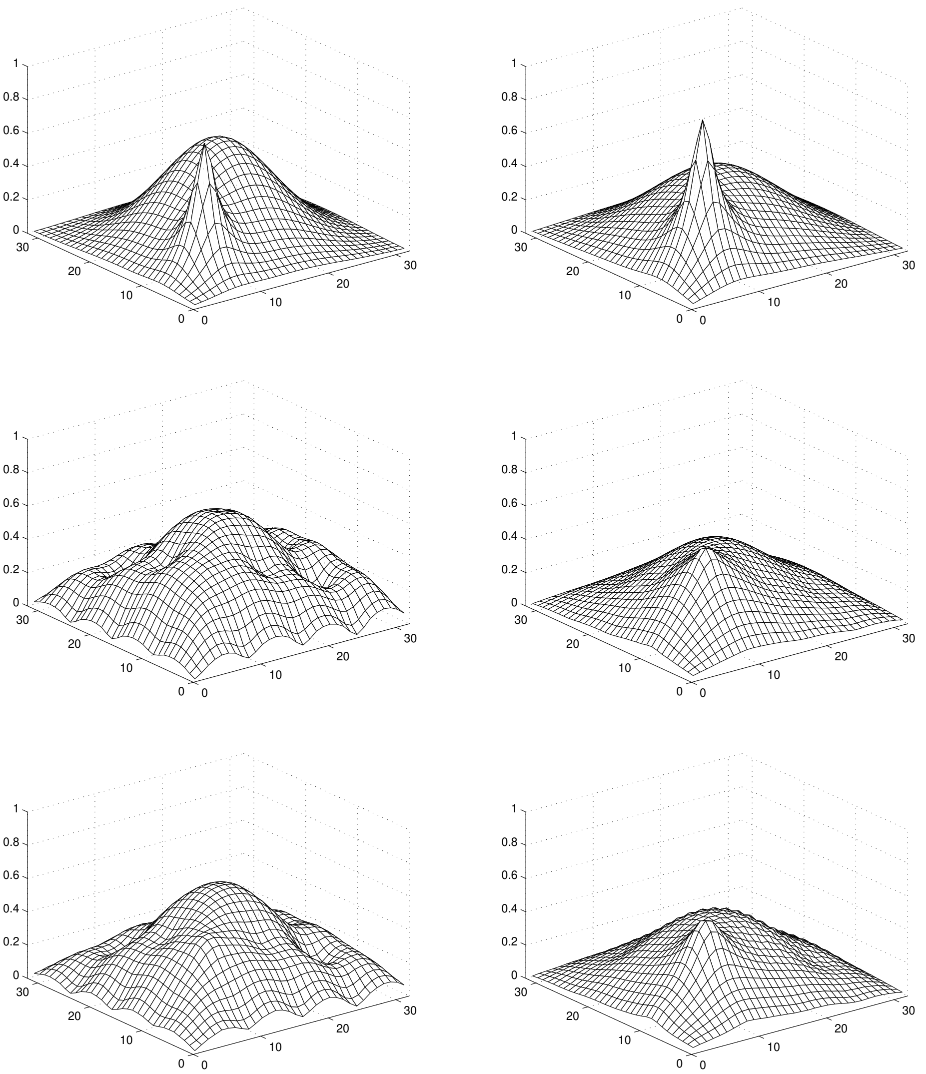

FIG. 4.2. *The top plots are of the true state at time points 35 and 70. The middle plots are of the Kalman filter estimates at time points 35 and 70. The bottom plots are of the LBFGS Kalman filter estimates at time points 35 and 70.*

[19] that for $\Delta t \leq 0.5$, the RK4 method is stable. The "true data" was generated by taking 42920 time steps of the RK4 method, which corresponds to 5365 days. The initial state at the beginning of 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

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

The observation error is simulated using the Gaussian random vector $N(\mathbf{0}, (0.15\,\sigma_{\text{clim}})^2\mathbf{I})$

ETNA
Kent State University
http://etna.math.kent.edu

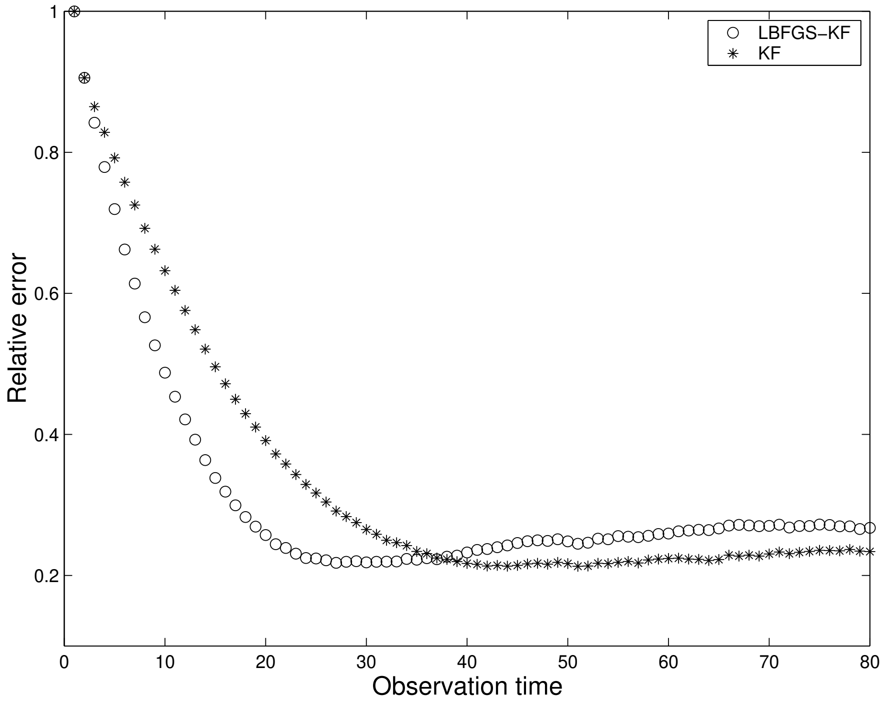

FIG. 4.3. *Relative error curves for KF (∗) and LBFGS-KF (o). The horizontal axis represents the observation time.*

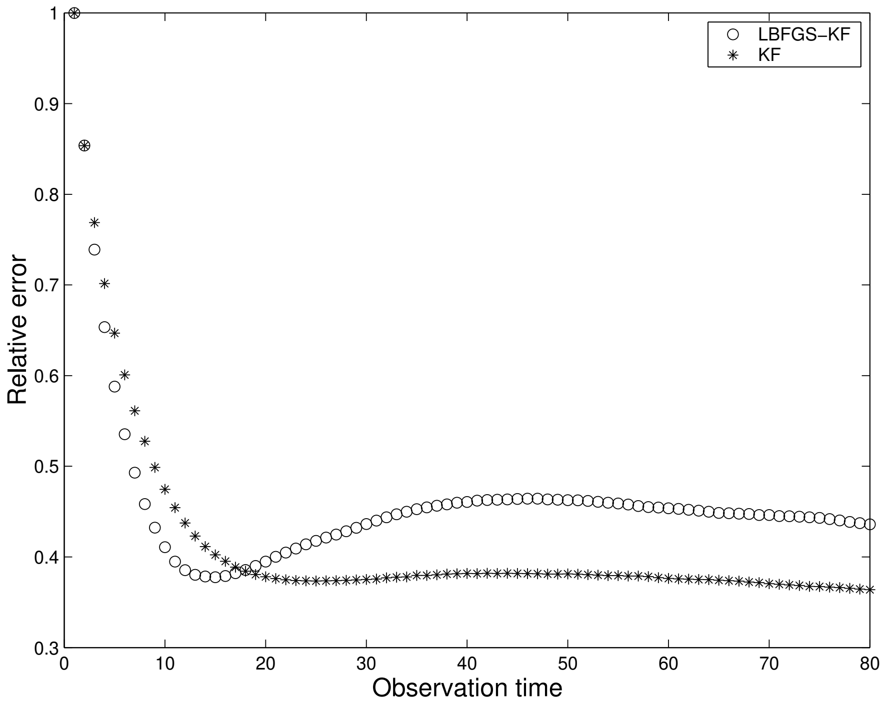

FIG. 4.4. *Relative error curves for KF (∗) and LBFGS-KF (o). The horizontal axis represents the observation time.*

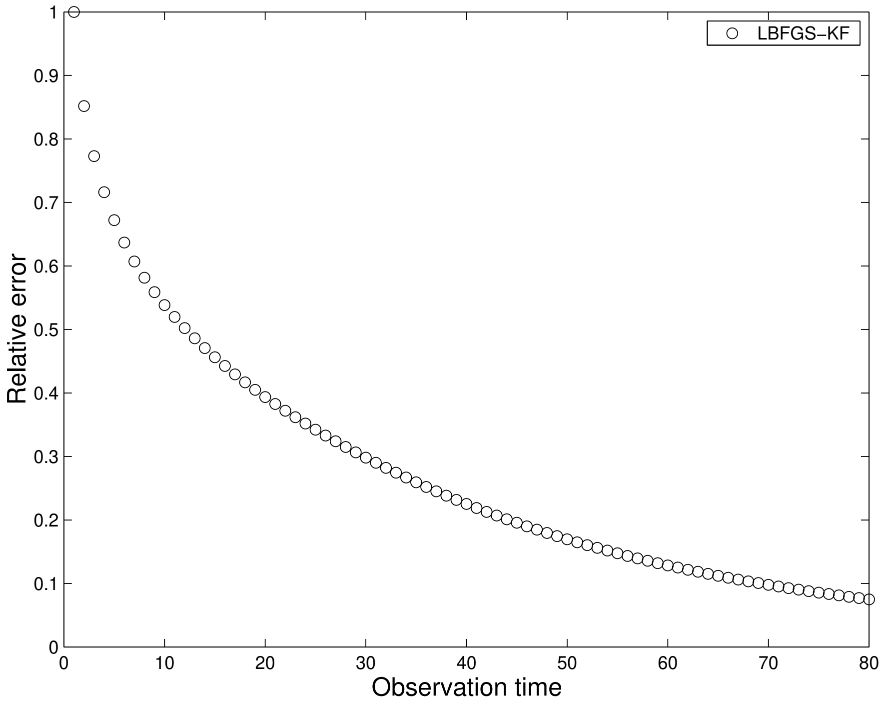

FIG. 4.5. *Relative error curves for LBFGS-KF. The horizontal axis represents the observation time.*

where $\sigma_{\rm clim}$ is a standard deviation of the model state used in climatological simulations, $\sigma_{\rm clim} := 3.6414723$. The data generation codes were written in MATLAB and were transcribed by us from the `scilab` codes written by the author of [15].

In our application of EKF and LFBGS-EKF, we assume the coupled stochastic system

$$\mathbf{x}_{k+1} = \mathcal{M}(\mathbf{x}_k) + N(\mathbf{0}, (0.05\,\sigma_{\rm clim})^2\mathbf{I}), \tag{4.5}$$

$$\mathbf{y}_{k+1} = \mathbf{K}\mathbf{x}_{k+1} + N(\mathbf{0}, (0.15\,\sigma_{\rm clim})^2\mathbf{I}), \tag{4.6}$$

where $\mathcal{M}(\mathbf{x}_k)$ is obtained by taking two steps of the RK4 method applied to (4.2) with initial condition $\mathbf{x}_k$ with time-step 0.025. We note that if the noise term is removed from (4.5) and the above initial condition is used, our data generation scheme results.

Due to the fact that $\mathcal{M}$ is a nonlinear function, EKF must be used; see (2.5) and (2.6). Since $\mathcal{K} := \mathbf{K}$ in (2.6) is linear, $\mathbf{K}_k = \mathbf{K}$ for all k in (2.7). However, a linearization of the nonlinear evolution function $\mathcal{M}$ is required. Fortunately, the computation of $\mathbf{M}_k$ in (2.7) is performed by a routine in one of the `scilab` codes mentioned above and that we have adapted for our use in MATLAB.

The initial condition used in implementation of both the EKF and LBFGS-EKF is defined by $[\mathbf{x}_{t0}]_i = [\mathbf{x}_{t0}^{\rm true}]_i + N(0, (0.3\,\sigma_{\rm clim})^2)$ for all i, and the initial covariance was taken to be $\mathbf{C}_0^{est} = (0.13\,\sigma_{\rm clim})^2\mathbf{I}$. In our implementation of the LBFGS method within LBFGS-EKF, we computed 10 iterations with 9 saved vectors.

In order to analyze the accuracy of the state estimates $\mathbf{x}_k^{est}$ obtained by both EKF and LBFGS-EKF we plot the vector with components

$$[\mathbf{rms}]_k = \sqrt{\frac{1}{40}\|\mathbf{x}_k^{est} - \mathbf{x}_k^{true}\|^2} \tag{4.7}$$

ETNA
Kent State University
http://etna.math.kent.edu

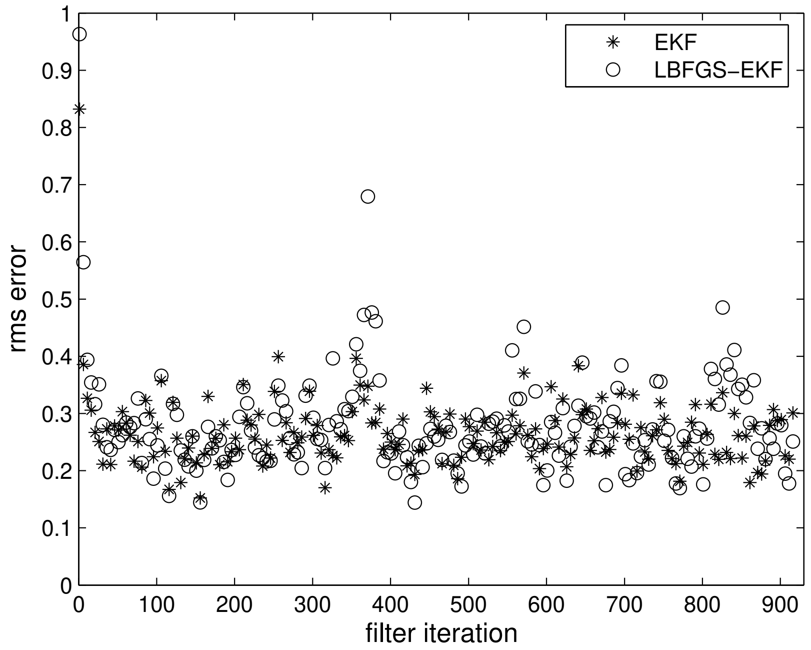

FIG. 4.6. *Plot of residual mean square error for LBFGS-EKF (o) and EKF (∗) applied to (4.2).*

in Figure 4.6. We can see that the two 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 $j \in \mathcal{I} := \{8i \mid i = 1, 2, \ldots, 100\}$ and define

$$[\textbf{forcast\_error}_j]_i = \frac{1}{40}\|\mathcal{M}_{4i}(\mathbf{x}_j^{est}) - \mathbf{x}_{j+4i}^{true}\|^2, \quad i = 1, \ldots, 20, \tag{4.8}$$

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. This allows us to define the forecast skill vector

$$[\textbf{forecast\_skill}]_i = \frac{1}{\sigma_{\text{clim}}}\sqrt{\frac{1}{100}\sum_{j\in\mathcal{I}}[\textbf{forecast\_error}_j]_i}, \quad i = 1, \ldots, 20, \tag{4.9}$$

which is plotted in Figure 4.7. The results show that the forecasting skill of the two methods is very similar, which suggests that on the whole, the quality of the LBFGS-EKF estimates is as high as those obtained using EKF. Figure 4.7 also illustrates the fact that the Lorenz 95 model (4.2) is truly chaotic.

In the test cases considered here, a linear or linearized model matrix $\mathbf{M}_k$ has been available. This is not true in important examples such as in numerical weather forecasting, where, on the other hand, a tangent linear code [14] is available that provides a means of computing the matrix vector product $\mathbf{M}_k\mathbf{x}$.

5. Conclusions. The standard implementations of KF and EKF become exceedingly time and memory intensive as the dimension of the underlying state space increases. Several

ETNA
Kent State University
http://etna.math.kent.edu

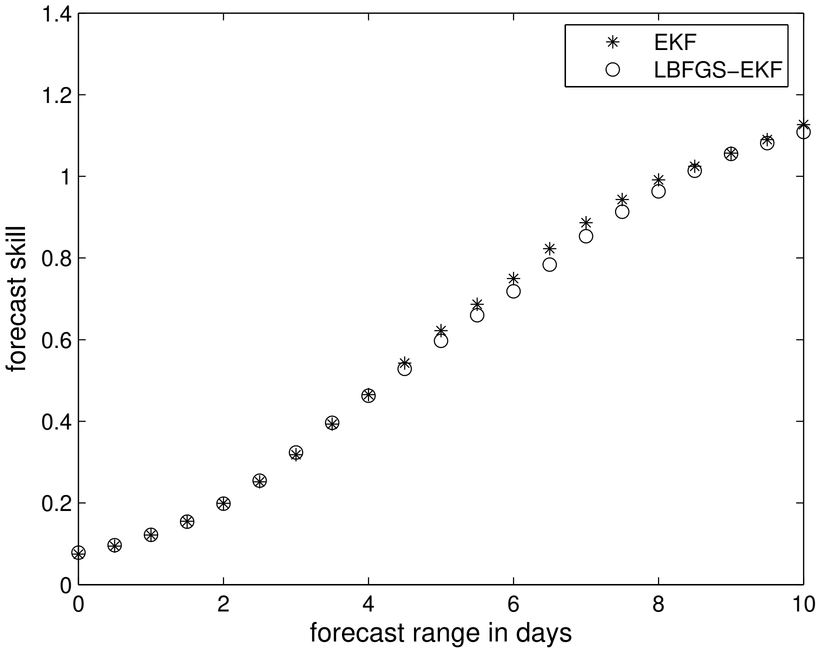

FIG. 4.7. *Plot of forecast skill vector for LBFGS-EKF (o) and EKF (∗) applied to (4.2).*

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., [4, 7, 30]) 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 of this is that nonlinear systems do 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 obtain using LBFGS. The large-scale matrix inversions required in KF and EKF implementations are also approximated using LBFGS. The resulting methods are denoted LBFGS-KF and LBFGS-EKF, respectively.

In order to test these methods, we consider two test cases: large-scale linear and small scale nonlinear. LBFGS-KF 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, LBFGS-EKF is implemented and is 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.

The symmetric rank one (SR1) quasi-Newton method for minimizing (3.1) could be another attractive method for use within KF and EKF, since it also yields estimates of both the minimizer and inverse Hessian. The main drawback of using SR1, however, is that the

ETNA
Kent State University
http://etna.math.kent.edu

inverse Hessian approximations are not guaranteed to be positive definite.

Appendix A. In this appendix, we give a general description of the LBFGS method for minimizing

$$q(\mathbf{u}) = \frac{1}{2}\langle \mathbf{A}\mathbf{u}, \mathbf{u}\rangle - \langle \mathbf{b}, \mathbf{u}\rangle,$$

where $\mathbf{A}$ is an $n \times n$ symmetric positive definite matrix and $\mathbf{b}$ is an $n \times 1$ vector. It is given by.

The LBFGS method for quadratic minimization
$\nu := 0$;
$\mathbf{u}_0 :=$ initial guess;
$\mathbf{B}_0^{-1} :=$ initial inverse Hessian approximation;
begin quasi-Newton iterations
  $\mathbf{g}_\nu := \nabla q(\mathbf{u}_\nu) = \mathbf{A}\mathbf{u}_\nu - \mathbf{b}$;
  $\mathbf{v}_\nu = \mathbf{B}_\nu^{-1}\mathbf{g}_\nu$;
  $\tau_\nu = \langle \mathbf{g}_\nu, \mathbf{v}_\nu\rangle / \langle \mathbf{v}_\nu, \mathbf{A}\mathbf{v}_\nu\rangle$;
  $\mathbf{u}_{\nu+1} := \mathbf{u}_\nu - \tau_\nu \mathbf{v}_\nu$;
  $\mathbf{B}_\nu^{-1} :=$ LBFGS approximation to $\mathbf{A}^{-1}$;
end quasi-Newton iterations

A.1. The limited memory approximation for $\mathbf{A}^{-1}$. The BFGS matrix $\mathbf{B}_\nu^{-1}$ is computed using recursion

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

where

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

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 ν, suppose that the j vector pairs $\{\mathbf{s}_i, \mathbf{d}_i\}_{i=\nu-j}^{\nu-1}$ are stored. Then we the LBFGS approximation of the inverse Hessian is given by

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

Assuming exact arithmetic and that j, we have that $\mathbf{u}_\nu$ converges to the unique minimizer of q in at most n iterations, and if n iterations are performed $\mathbf{B}_{n+1}^{-1} = \mathbf{A}^{-1}$ [22]. In the implementation in this paper, however, $j << n$ and LBFGS iterations are truncated before convergence is obtained.

It is proven in [21] 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 [22]

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

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

A.2. A low storage approximation of $\mathbf{A}$. The required formulas are given in [3] and take the following form. Let

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

then

$$\mathbf{B}_\nu = \xi_\nu \mathbf{I} - [\xi_\nu \mathbf{S}_\nu \quad \mathbf{D}_\nu] \begin{bmatrix} \xi_\nu \mathbf{S}_\nu^{\mathrm{T}}\mathbf{S}_\nu & \mathbf{L}_\nu \\ \mathbf{L}_\nu^{\mathrm{T}} & -\mathbf{D}_\nu \end{bmatrix}^{-1} \begin{bmatrix} \xi_\nu \mathbf{S}_\nu^{\mathrm{T}} \\ \mathbf{D}^{\mathrm{T}} \end{bmatrix}, \tag{A.2}$$

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

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

and

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

We note that when $\xi_\nu = 1$ for all ν in (A.2), we have an exact equality between $\mathbf{B}_\nu$ in (A.2) and (A.1). However, we have found that a more accurate Hessian approximation is obtained if, following [22], we use the scaling $\xi_\nu = \mathbf{d}_{\nu-1}^{\mathrm{T}}\mathbf{d}_{\nu-1}/\mathbf{s}_{\nu-1}^{\mathrm{T}}\mathbf{d}_{\nu-1}$ instead.

We note that the middle matrix in (A.2) 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; see [3] for details.

Acknowledgments. The authors would like to thank the referees and ETNA editors; the paper is better because of their efforts. This work was supported by the Academy of Finland (application number 213476, Finnish programme for Centres of Excellence in research 2006–2011). 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. 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 during the 2006-07 academic year, where this work was undertaken.

REFERENCES

[1] H. Auvinen, H. Haario, and T. Kauranne, *Optimal approximation of Kalman filtering with a temporally local 4D-Var in operational weather forecasting*, in Proceedings of the 11th ECMWF Workshop on Use of High-Performance Computing in Meteorology, W. Zwiefelhofer and G. Mozdzynski, eds., World Scientific, London, 2005.

[2] H. Auvinen, J. M. Bardsley, H. Haario, and T. Kauranne, *The variational Kalman filter and an efficient implementation using limited memory BFGS*, Internat. J. Numer. Methods Fluids, to appear.

[3] R. H. BYRD, J. NOCEDAL, AND R. B. SCHNABEL, *Representations of quasi-Newton matrices and their use in limited memory methods*, Math. Program., 63 (1994), pp. 129–156.
[4] M. A. CANE, 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., 101 (1996), pp. 599–617.
[5] Y. CAO, J. ZHU, I. M. NAVON, AND Z. LUO, *A reduced order approach to four-dimensional variational data assimilation using proper orthogonal decomposition*, Internat. J. Numer. Methods Fluids, 53 (2007), pp. 1571–1583.
[6] R. DALEY, *Atmospheric Data Analysis*, Cambridge University Press, Cambridge, UK, 1991.
[7] D. P. DEE, *Simplification of the Kalman filter for meteorological data assimilation* Q. J. R. Meteorol. Soc., 117 (1990), pp. 365–384.
[8] M. FISHER, *Minimization algorithms for variational data assimilation*, in Recent Developments in Numerical Methods for Atmospheric Modelling, Reading, UK, September 7–11, 2008, European Centre for Medium-Range Weather Forecasts, Reading, UK, 1998, pp. 364–385.
[9] M. FISHER AND E. ANDERSSON, *Developments in 4D-var and Kalman filtering*, European Centre for Medium-Range Weather Forecasts Technical Memorandum 347, Reading, UK, 2001.
[10] M. FISHER AND P. COURTIER, *Estimating the covariance matrices of analysis and forecast error in variational data assimilation*, European Centre for Medium-Range Weather Forecasts Technical Memorandum 220, Reading, UK, 1995.
[11] M. FISHER, M. LEUTBECHER, AND G. KELLEY, *On the equivalence between Kalman smoothing and weak-constraint four-dimensional variational data assimilation*, Q. J. R. Meteorol. Soc., 131 (2005), pp. 3235–3246.
[12] M. FISHER, J. NOCEDAL, Y. TRÉMOLET, AND S. J. WRIGHT, *Data assimilation in weather forecasting: a case study in PDE-constrained optimization*, Optim. Eng., 10 (2009), pp. 1389–4420.
[13] I. Y. GEJADZE, F.-X. LE DIMET, AND V. SHUTYAEV, *On analysis error covariances in variational data assimilation*, SIAM J. Sci. Comput., 30 (2008), pp. 1847–1874.
[14] F.-X. LEDIMET AND O. TALAGRAND, *Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects*, Tellus Ser. A, 38 (1986), pp. 97–110.
[15] M. LEUTBECHER, *A data assimilation tutorial based on the Lorenz-95 system*, European Centre for Medium-Range Weather Forecasts Web Tutorial, Reading, UK. Available at www.ecmwf.int/newsevents/training/meteorological_presentations/pdf/DA/ToyModel_notes.pdf.
[16] Z. LI AND M. NAVON, *Optimality of variational data assimilation and its relationship with the Kalman filter and smoother*, Q. J. R. Meteorol. Soc., 127 (2008), pp. 661–683.
[17] A. C. LORENC, *Modelling of error covariances by 4D-Var data assimilation*, Q. J. R. Meteorol. Soc., 129 (2003), pp. 3167–3182.
[18] E. N. LORENZ, *Predictability: A problem partly solved*, in Predictability of Weather and Climate, Tim Palmer and Renate Hagedorn, eds., Cambridge University Press, Cambridge, UK, 2006, pp. 40–58. Originally presented in a 1996 European Centre for Medium-Range Weather Forecasts workshop.
[19] E. N. LORENZ AND K. A. EMANUEL, *Optimal sites for supplementary weather observations: simulation with a small model*, J. Atmospheric Sci., (1998), pp. 399–414.
[20] L. NAZARETH, *A relationship between the BFGS and conjugate gradient algorithms and it implications for new algorithms*, SIAM J. Numer. Anal., 16 (1979), pp. 794–800.
[21] J. NOCEDAL, *Updating quasi-Newton matrices with limited storage*, Math. Comp., 35 (1980), pp. 773–782.
[22] J. NOCEDAL AND S. WRIGHT, *Numerical Optimization*, Springer, New York, 1999.
[23] F. RABIER, H. JÄRVINEN, E. KLINKER, J. F. MAHFOUF, AND A. SIMMONS, *The ECMWF operational implementation of four-dimensional variational assimilation. Part I: experimental results with simplified physics*, Q. J. R. Meteorol. Soc., 126 (2000), pp. 1143–1170.
[24] C. D. RODGERS, *Inverse Methods for Atmospheric Sounding: Theory and Practice*, World Scientific, London, 2000.
[25] X. TIAN, Z. XIE, AND A. DAI, *An ensemble-based explicit four-dimensional variational assimilation method*, J. Geophys. Res., 113 (2008), pp. D21124 (1–13).
[26] J. TSHIMANGA, S. GRATTON, A. T. WEAVER, AND A. SARTENAER, *Limited-memory preconditioners, with application to incremental four-dimensional variational data assimilation*, Q. J. R. Meteorol. Soc., 134 (2008), pp. 751–769.
[27] F. VEERSÉ, *Variable-storage quasi-Newton operators as inverse forecast/analysis error covariance matrices in variational data assimilation*, INRIA Report 3685, April 26, 1999.
[28] F. VEERSÉ, *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, 1999, Quebec City, Canada, World Meteorological Organization, Geneva.
[29] F. VEERSÉ, D. AUROUX, AND M. FISHER, *Limited-memory BFGS diagonal predonditioners for a data assimilation problem in meteorology*, Optim. Eng., 1 (2000), pp. 323–339.

ETNA
Kent State University
http://etna.math.kent.edu

[30] A. VOUTILAINEN, T. PYHÄLAHTI, K. KALLIO, J. PULLIAINEN, H. HAARIO, AND J. KAIPIO, *A filtering approach for estimating lake water quality from remote sensing data*, Int. J. Appl. Earth Observation & Geoinformation, 9 (2007), pp. 50–64.

[31] W. YANG, I. M. NAVON, AND P. COURTIER, *A new Hessian preconditioning method applied to variational data assimilation experiments using NASA general circulation models*, Monthly Weather Rev., 124 (1996), pp. 1000–1017.