AN ENSEMBLE KALMAN FILTER USING THE CONJUGATE GRADIENT SAMPLER

Johnathan M. Bardsley,[1,*] *Antti Solonen,*[2,3] *Albert Parker,*[4] *Heikki Haario,*[2] *& Marylesa Howard*[1]

[1] *Department of Mathematical Sciences, University of Montana, Missoula, Montana, 59812, USA*

[2] *Department of Mathematics and Physics, Lappeenranta University of Technology, Lappeenranta, Finland*

[3] *Finnish Meteorological Institute, Helsinki, Finland*

[4] *Center for Biofilm Engineering, Montana State University, Bozeman, Montana, 59717, USA*

Original Manuscript Submitted: 08/23/2011; Final Draft Received: 11/30/2011

The ensemble Kalman filter (EnKF) is a technique for dynamic state estimation. EnKF approximates the standard extended Kalman filter (EKF) by creating an ensemble of model states whose mean and empirical covariance are then used within the EKF formulas. The technique has a number of advantages for large-scale, nonlinear problems. First, large-scale covariance matrices required within EKF are replaced by low-rank and low-storage approximations, making implementation of EnKF more efficient. Moreover, for a nonlinear state space model, implementation of EKF requires the associated tangent linear and adjoint codes, while implementation of EnKF does not. However, for EnKF to be effective, the choice of the ensemble members is extremely important. In this paper, we show how to use the conjugate gradient (CG) method, and the recently introduced CG sampler, to create the ensemble members at each filtering step. This requires the use of a variational formulation of EKF. The effectiveness of the method is demonstrated on both a large-scale linear, and a small-scale, nonlinear, chaotic problem. In our examples, the CG-EnKF performs better than the standard EnKF, especially when the ensemble size is small.

KEY WORDS: *ensemble Kalman filter, data assimilation, conjugate gradient iteration, conjugate gradient sampler*

1. INTRODUCTION

The Kalman filter (KF), first introduced in [1], is the extension of Bayesian minimum variance estimation to problems in which the unknown to be estimated, which we call the state, varies according to some (approximately) known model, and is indirectly and sequentially observed in time.

KF assumes linear state and observation models, and its extension to nonlinear cases is known as the extended Kalman filter (EKF) [2]. The outputs from KF and EKF at a particular time step are estimates of the state and its covariance matrix. At the next time step, as new data come in, these estimates are then updated using the Kalman, or some equivalent, formulas.

Both KF and EKF have been successfully used in a number of settings, e.g., autonomous and assisted navigation. However, for problems in which the dimension of the state space is prohibitively large, such as arising in weather and ocean forecasting, storing and operating by the state covariance matrix, which is dense, is computationally prohibitive.

[*] Correspond to Johnathan M. Bardsley, E-mail: bardsleyj@mso.umt.edu

DOI: 10.1615/Int.J.UncertaintyQuantification.2012003889

To overcome this issue, a number of approximations have been developed that replace the state-space covariance matrices appearing within the filter by low-rank or low-storage approximations. These include approaches in which the state space is projected onto a smaller subspace [3–6]. However, since the chosen subspace is typically fixed in time, the dynamics of the system, which changes in time, is often not correctly captured [7].

A related approach makes use of iterative methods both for state estimation and covariance approximation. For example, in [8–10], the limited memory BroydenFletcherGoldfarbShanno (BFGS) (LBFGS) method is used for state estimation, as well as to build low-storage (full rank) covariance approximations, in both standard and variational [Bayesian maximum *a posteriori* (MAP)] formulations of KF and EKF. In [11], the same approach is taken, with the conjugate gradient iteration used instead of LBFGS, yielding low-rank covariance approximations. Similar ideas are explored in [12]; however, the covariance approximations are derived in a more complicated fashion. These approaches have the benefit that the associated covariance approximations change with each step in the filter based on local information. However, for nonlinear problems, the methods require tangent linear and adjoint codes for computing the Jacobian of the model and its adjoint at an arbitrary state. Such software is model dependent and can be quite tedious to write for complex models.

An approach that does not require such linearization software is the so-called ensemble Kalman filter (EnKF). In EnKF, a random sample of states, called an ensemble, is computed at each time step, and the state and covariance estimates are taken to be the sample mean and covariance of the ensemble. EnKF was first proposed in [13], but several variants now exist [14]. Not surprisingly, the technique has its own set of problems, e.g., sampling errors due to random perturbations of model state and observations, and from ensemble in-breeding; see, e.g., [15–17]. Moreover, ensemble filters yield low-rank approximate covariance matrices, with column space spanned by the mean-subtracted ensemble members. To make the approximations full-rank, an additional matrix must be added, which is known in the data assimilation literature as "covariance inflation" [18, 19].

In a recent paper [20], an ensemble version of the variational approach set forth in [9] is presented. As in [9], LBFGS is used to minimize a quadratic function—the negative log of the Bayesian posterior density function—whose minimizer and inverse Hessian are the Kalman filter state estimate and covariance, respectively. A new ensemble is generated directly from the covariance approximation without having to store dense matrices and perform matrix decompositions. The resulting forecast ensemble is then used to build an approximate forecast covariance without the use of tangent linear and adjoint codes. In contrast with standard EnKF approaches, random perturbations of the model state and observations are not used in this method, and hence some of the problems related to existing ensemble methods do not arise.

In this paper, we follow the general approach of [20], but instead use the conjugate gradient (CG) iteration for quadratic minimization and the CG sampler of [21] for ensemble calculation. The resulting method has several advantages over the LBFGS version. Perhaps most importantly, for the examples we consider, it results in a more accurate and faster converging filter. Also, it is more intuitive and much simpler to implement, requiring the addition of only a single inexpensive line of code within CG. And finally, a rigorous supporting theory for the accuracy of the CG ensembles was developed in the recent work of [21]. We call the proposed method the conjugate gradient ensemble Kalman filter (CG-EnKF).

The paper is organized as follows. In Section 2, we recall the basics of Kalman filtering and ensemble methods. We introduce the CG-EnKF algorithm and the relevant CG theory in Section 3 and demonstrate the method with numerical examples in Section 4. We end with conclusions in Section 5.

2. KALMAN FILTERING METHODS

In this paper, we consider the following coupled system of discrete, nonlinear, stochastic difference equations

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

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

In the first equation, $\mathbf{x}_k$ denotes the $n \times 1$ state vector of the system at time k, $\mathcal{M}$ is the (possibly) nonlinear evolution operator, and $\boldsymbol{\varepsilon}_k^p$ is a $n \times 1$ random vector representing the model 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 matrices $\mathbf{C}_{\boldsymbol{\varepsilon}_k^p}$ and $\mathbf{C}_{\boldsymbol{\varepsilon}_k^o}$, respectively. In this paper, we do not consider the (often cumbersome) estimation of these matrices, and assume that they are given.

The task is to estimate the state $\mathbf{x}_k$ and its error covariance $\mathbf{C}_k$ at time point k given $\mathbf{y}_k$, $\mathbf{K}_k$, $\mathbf{C}_{\boldsymbol{\varepsilon}_k^o}$, the function $\mathcal{M}(\mathbf{x})$, $\mathbf{C}_{\boldsymbol{\varepsilon}_k^p}$ and estimates $\mathbf{x}_{k-1}^{est}$ and $\mathbf{C}_{k-1}^{est}$ of the state and covariance at time point $k-1$.

The extended Kalman filter (EKF) is the standard method for solving such problems [2]. The formulation of EKF requires that we linearize the nonlinear function $\mathcal{M}$ at $\mathbf{x}_k$ for each k. In particular, we define

$$\mathbf{M}_k = \partial \mathcal{M}(\mathbf{x}_{k-1}^{est})/\partial \mathbf{x}, \tag{3}$$

where $\partial/\partial\mathbf{x}$ denotes the Jacobian computation with respect to $\mathbf{x}$. EKF then has the following form.

Algorithm 1 (EKF). *Select initial guess $\mathbf{x}_0^{est}$ and covariance $\mathbf{C}_0^{est}$, and set $k = 1$.*

1. *Compute the evolution model estimate and covariance:*
   - a. *Compute $\mathbf{x}_k^p = \mathcal{M}(\mathbf{x}_{k-1}^{est})$;*
   - b. *Define $\mathbf{M}_k = \partial\mathcal{M}(\mathbf{x}_{k-1}^{est})/\partial\mathbf{x}$ and $\mathbf{C}_k^p = \mathbf{M}_k\mathbf{C}_{k-1}^{est}\mathbf{M}_k^{\mathrm{T}} + \mathbf{C}_{\boldsymbol{\varepsilon}_k^p}$.*
2. *Compute Kalman filter estimate and covariance:*
   - a. *Define the Kalman gain matrix $\mathbf{G}_k = \mathbf{C}_k^p\mathbf{K}_k^{\mathrm{T}}(\mathbf{K}_k\mathbf{C}_k^p\mathbf{K}_k^{\mathrm{T}} + \mathbf{C}_{\boldsymbol{\varepsilon}_k^o})^{-1}$;*
   - b. *Compute the Kalman filter estimate $\mathbf{x}_k^{est} = \mathbf{x}_k^p + \mathbf{G}_k(\mathbf{y}_k - \mathbf{K}_k\mathbf{x}_k^p)$;*
   - c. *Define the estimate covariance $\mathbf{C}_k^{est} = \mathbf{C}_k^p - \mathbf{G}_k\mathbf{K}_k\mathbf{C}_k^p$.*
3. *Update $k := k + 1$ and return to step* i.

Note that in the linear case $\mathcal{M}(\mathbf{x}_{k-1}) = \mathbf{M}_k\mathbf{x}_{k-1}$, and EKF reduces to the classical linear Kalman filter [1].

For large-scale, nonlinear problems, such as arising in weather and sea forecasting, the linearization of $\mathcal{M}$ required within EKF can be problematic. For example, numerical approximations may need to be used, yielding inaccuracies, or the linearization might be either too computationally burdensome or complicated. Storage of dense covariance matrices of size $n \times n$, where n is the size of the state space, can also be problematic.

An approximation of EKF that has reduced storage requirements and does not involve linearization of $\mathcal{M}$ is the ensemble Kalman filter. In EnKF, a representative ensemble is sampled at each step and is integrated forward by the model $\mathcal{M}$. The resulting ensemble is then used to create a low-rank (and hence low-storage) approximation of the model covariance $\mathbf{C}_k^p$.

Algorithm 2 (EnKF). *Sample initial ensemble $\mathbf{x}_{0,i}^{est}$ for $i = 1, \ldots, N$ from $N(\mathbf{x}_0^{est}, \mathbf{C}_0^{est})$, and set $k = 1$.*

1. *Integrate the ensemble forward in time and then compute the evolution model estimate and covariance:*
   - a. *Sample $\boldsymbol{\varepsilon}_{k,i}^p \sim N(\mathbf{0}, \mathbf{C}_{\boldsymbol{\varepsilon}_k^p})$, $i = 1, \ldots, N$, then define*
     $\mathbf{x}_{k,i}^p = \mathcal{M}(\mathbf{x}_{k-1,i}^{est}) + \boldsymbol{\varepsilon}_{k,i}^p$;
   - b. *Set $\mathbf{x}_k^p = \mathcal{M}(\mathbf{x}_{k-1}^{est})$, and estimate the model covariance using*
     $\mathbf{C}_k^p = 1/N\sum_{i=1}^N(\mathbf{x}_{k,i}^p - \mathbf{x}_k^p)(\mathbf{x}_{k,i}^p - \mathbf{x}_k^p)^T$.
2. *Compute a new ensemble using the Kalman filter formulas:*
   - a. *Define the Kalman gain matrix $\mathbf{G}_k = \mathbf{C}_k^p\mathbf{K}_k^{\mathrm{T}}(\mathbf{K}_k\mathbf{C}_k^p\mathbf{K}_k^{\mathrm{T}} + \mathbf{C}_{\boldsymbol{\varepsilon}_k^o})^{-1}$;*

b. *Sample* $\boldsymbol{\varepsilon}^o_{k,i} \sim N(\mathbf{0}, \mathbf{C}_{\boldsymbol{\varepsilon}^o_k})$ *for* $i = 1, \ldots, N$, *then define new ensemble members* $\mathbf{x}^{est}_{k,i} = \mathbf{x}^p_{k,i} + \mathbf{G}_k(\mathbf{y}_k - \mathbf{K}_k\mathbf{x}^p_{k,i} + \boldsymbol{\varepsilon}^o_{k,i})$*;*

c. *Compute the state and covariance estimates as the ensemble mean* $\mathbf{x}^{est}_k = 1/N\sum_{i=1}^N \mathbf{x}^{est}_{k,i}$ *and empirical covariance* $\mathbf{C}^{est}_k = 1/(N-1)\sum_{i=1}^N(\mathbf{x}^{est}_{k,i} - \mathbf{x}^{est}_k)(\mathbf{x}^{est}_{k,i} - \mathbf{x}^{est}_k)^T$.

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

EnKF computations can be carried out efficiently so that the covariances are kept in the low-rank "ensemble form", without explicitly computing the covariance $\mathbf{C}^p_k$ or the Kalman gain matrix $\mathbf{G}_k$ [14]. However, these low-rank approximations typically require the use of a form of regularization known as "covariance inflation" [18, 19]. Another issue with EnKF is that unless the ensemble size N is sufficiently large, the estimator defined by the ensemble mean in step *ii*.3 can be inaccurate and the method may perform poorly. Finally, EnKF also suffers from inaccuracies brought on by the random perturbation of model states and observations; see steps *i*.1 and *ii*.2 of the algorithm.

3. THE CG ENSEMBLE KALMAN FILTER

The Kalman filter estimate $\mathbf{x}^{est}_k$ and its covariance $\mathbf{C}^{est}_k$, described in step *ii* of Algorithm 1, can be written in Bayesian MAP form, as the minimizer and inverse Hessian of the quadratic function

$$\ell(\mathbf{x}|\mathbf{y}_k) = \frac{1}{2}(\mathbf{y}_k - \mathbf{K}_k\mathbf{x})^{\mathrm{T}}\mathbf{C}^{-1}_{\boldsymbol{\varepsilon}^o_k}(\mathbf{y}_k - \mathbf{K}_k\mathbf{x}) + \frac{1}{2}(\mathbf{x} - \mathbf{x}^p_k)^{\mathrm{T}}(\mathbf{C}^p_k)^{-1}(\mathbf{x} - \mathbf{x}^p_k), \tag{4}$$

resulting in an equivalent variational (i.e., optimization-based) formulation of the Kalman filter. Note that (4) is the negative log of the posterior density function with likelihood defined by the observation model (2) and defined prior by step *i* of Algorithm 1, i.e., by $N(\mathbf{x}^p_k, \mathbf{C}^p_k)$. Also, using (4) requires that multiplication by $\mathbf{C}^{-1}_{\boldsymbol{\varepsilon}^o_k}$ is efficient.

An alternative approach to EnKF is to reformulate step *ii* in Algorithm 2 using (4), so that instead an iterative method is used to estimate both the minimizer and inverse Hessian, $\nabla^2\ell(\mathbf{x}|\mathbf{y}_k)^{-1}$, of (4). The N new ensemble members are then sampled from $N(\mathbf{x}^{est}_k, \mathbf{C}^{est}_k)$, where $\mathbf{x}^{est}_k$ and $\mathbf{C}^{est}_k$ are the estimates of the minimizer and inverse Hessian computed by the method.

To efficiently implement the variational approach, one must overcome two computational bottlenecks. First, one needs to make sure that the evaluation of the quadratic expression is efficient and that no large dense matrices are stored in memory. Especially, an efficient way to multiply by $(\mathbf{C}^p_k)^{-1}$ is needed. Second, an efficient way to produce samples from $N(\mathbf{x}^{est}_k, \mathbf{C}^{est}_k)$ is required. Usually, sampling from a Gaussian is done via the Cholesky decomposition of the dense covariance matrix, which is prohibitively expensive in high-dimensional cases, where one cannot even store dense covariance matrices.

In [20], low-storage covariance approximations are used to overcome the above issues, computed using the limited memory BFGS (LBFGS) [22] optimization algorithm. As is well-known, LBFGS yields an approximate minimizer and uses iteration history to construct an approximate inverse Hessian of (4). As shown in [20], random samples can also be produced efficiently from the LBFGS covariance (inverse Hessian) approximations.

CG iteration history can also be used to create a low-rank approximation of the inverse Hessian. Moreover, recent results in [21] show how samples from the Gaussian with this approximate covariance can be easily generated from within CG with almost no additional computational cost, and in such a way that the only additional requirement is to store the N samples (ensemble members), each of size $n \times 1$.

Computationally, the CG implementation is slightly more efficient than the LBFGS implementation. When LBFGS is used, some additional storage besides N samples is required for the LBFGS covariance approximation, and the generation of the ensemble members requires an additional computation after the LBFGS iteration has stopped; see [20] for details.

Finally, the theory for the accuracy of the CG samples is well-developed (Section 3.1.1), whereas, to our knowledge, such an analysis does not yet exist for LBFGS inverse-Hessian approximation.

The CG ensemble Kalman filter has the following form.

Algorithm 3 (CG-EnKF). *Sample initial ensemble* $\mathbf{x}_{0,i}^{est}$ *for* $i = 1, \ldots, N$ *from* $N(\mathbf{x}_0^{est}, \mathbf{C}_0^{est})$*, and set* $k = 1$.

1. *Integrate the ensemble forward in time and estimate its covariance:*
   a. *Set* $\mathbf{x}_k^p = \mathcal{M}(\mathbf{x}_{k-1}^{est})$ *and* $\mathbf{x}_{k,i}^p = \mathcal{M}(\mathbf{x}_{k-1,i}^{est})$ *for* $i = 1, \ldots, N$*;*
   b. *Estimate the model covariance using* $\mathbf{C}_k^p = 1/N \sum_{i=1}^N (\mathbf{x}_{k,i}^p - \mathbf{x}_k^p)(\mathbf{x}_{k,i}^p - \mathbf{x}_k^p)^T + \mathbf{C}_{\boldsymbol{\varepsilon}_k^p}$.
2. *Compute a new ensemble using the CG sampler:*
   a. *Use CG to estimate the minimizer* $\mathbf{x}_k^{est}$ *of (4), as well as to compute the new ensemble members* $\mathbf{x}_{k,i}^{est}$ *for* $i = 1, \ldots, N$ *from* $N(\mathbf{x}_k^{est}, \mathbf{C}_k^{est})$*, where* $\mathbf{C}_k^{est}$ *is the approximation of* $\nabla^2 \ell(\mathbf{x}|\mathbf{y}_k)^{-1}$ *generated by CG.*
3. *Update* $k := k + 1$ *and return to step* i.

Applying CG in step *ii*.1 requires that efficient multiplication by $(\mathbf{C}_k^p)^{-1}$ is possible. For this, define

$$\mathbf{X}_k = [(\mathbf{x}_{k,1}^p - \mathbf{x}_k^p), (\mathbf{x}_{k,2}^p - \mathbf{x}_k^p), \ldots, (\mathbf{x}_{k,N}^p - \mathbf{x}_k^p)]/\sqrt{N}.$$

Then $\mathbf{C}_k^p = \mathbf{X}_k \mathbf{X}_k^T + \mathbf{C}_{\boldsymbol{\varepsilon}_k^p}$, and the matrix inversion lemma can be used as in [20]:

$$(\mathbf{C}_k^p)^{-1} = (\mathbf{X}_k \mathbf{X}_k^T + \mathbf{C}_{\boldsymbol{\varepsilon}_k^p})^{-1} = \mathbf{C}_{\boldsymbol{\varepsilon}_k^p}^{-1} - \mathbf{C}_{\boldsymbol{\varepsilon}_k^p}^{-1} \mathbf{X}_k (\mathbf{I} + \mathbf{X}_k^T \mathbf{C}_{\boldsymbol{\varepsilon}_k^p}^{-1} \mathbf{X}_k)^{-1} \mathbf{X}_k^T \mathbf{C}_{\boldsymbol{\varepsilon}_k^p}^{-1}. \tag{5}$$

Note that $\mathbf{I} + \mathbf{X}_k^T \mathbf{C}_{\boldsymbol{\varepsilon}_k^p}^{-1} \mathbf{X}_k$ is an $N \times N$ matrix. Thus, assuming that N is not too large, and that multiplication by $\mathbf{C}_{\boldsymbol{\varepsilon}_k^p}^{-1}$ is efficient, multiplication by (5) will also be efficient. In our examples, we assume that $\mathbf{C}_{\boldsymbol{\varepsilon}_k^p}$ is a diagonal matrix, which makes the inversion and multiplication with a vector easy.

We observe that in contrast to EnKF and many of its variants, the model error term is included in the cost function $\ell(\mathbf{x}|\mathbf{y}_k)$ directly, and the optimization is carried out in the full state space. Since a full-rank model error covariance is used, the technique of covariance inflation [18, 19] is not needed. Naturally, the trouble of tuning the model error covariance still remains, but this quantity has a clear statistical interpretation, unlike the rather *ad hoc* covariance inflation parameters. Many existing ensemble methods cannot incorporate model error directly, and this is one of the strengths of the variational approach.

We will now introduce the CG sampler, used in step *ii*.1, for computing both the estimator and ensemble members.

3.1 The CG Sampler

First, we rewrite $\ell(\mathbf{x}|\mathbf{y}_k)$ defined in step (4) as

$$\ell(\mathbf{x}|\mathbf{y}_k) \simeq \frac{1}{2}\mathbf{x}^T \mathbf{A} \mathbf{x} - \mathbf{x}^T \mathbf{b}, \tag{6}$$

where $\mathbf{A} = \mathbf{K}_k^T (\mathbf{C}_{\boldsymbol{\varepsilon}_k^o})^{-1} \mathbf{K}_k + (\mathbf{C}_k^p)^{-1}$ and $\mathbf{b} = \mathbf{K}_k^T (\mathbf{C}_{\boldsymbol{\varepsilon}_k^o})^{-1} \mathbf{y}_k + (\mathbf{C}_k^p)^{-1} \mathbf{x}_k^p$.

Recent work of [21] shows that while finding the minimizer of the quadratic ℓ, the CG algorithm can also be used to inexpensively generate approximate samples $\mathbf{w}_i$ from $N(\mathbf{0}, \mathbf{A}^{-1})$. The new ensemble (in step *ii*.1 of Algorithm 3) is then defined

$$\mathbf{x}_{k,i}^{est} = \mathbf{x}_k^{est} + \mathbf{w}_i, \tag{7}$$

where $\mathbf{x}_k^{est}$ is the approximate minimizer of $\ell(\mathbf{x}|\mathbf{y}_k)$ computed by CG.

We call this modified CG algorithm the CG sampler, as in [21]. It is given as follows.

Algorithm 4 (CG sampler). *Given* $\mathbf{A}$, $\mathbf{b}$, *and* $\mathbf{x}_0$, *let* $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$, $\mathbf{p}_0 = \mathbf{r}_0$, $d_0 = \mathbf{p}_0^T \mathbf{A} \mathbf{p}_0$, $j = 1$, *and* $\mathbf{w}_{i,0} = \mathbf{0}$ *for* $i = 1, \ldots, N$. *Specify some stopping tolerance* ϵ *and iterate:*

1. $\gamma_{j-1} = (\mathbf{r}_{j-1}^T \mathbf{r}_{j-1})/(d_{j-1})$;
2. $\mathbf{x}_j = \mathbf{x}_{j-1} + \gamma_{j-1}\mathbf{p}_{j-1}$;
3. $\mathbf{w}_{i,j} = \mathbf{w}_{i,j-1} + (z_i/\sqrt{d_{j-1}})\mathbf{p}_{j-1}$, $z_i \sim N(0,1)$, *for* $i = 1, \ldots, N$;
4. $\mathbf{r}_j = \mathbf{b} - \mathbf{A}\mathbf{x}_j = \mathbf{r}_{j-1} - \gamma_{j-1}\mathbf{A}\mathbf{p}_{j-1}$;
5. $\beta_j = -(\mathbf{r}_j^T \mathbf{r}_j)/(\mathbf{r}_{j-1}^T \mathbf{r}_{j-1})$;
6. $\mathbf{p}_j = \mathbf{r}_j - \beta_j \mathbf{p}_{j-1}$ *and* $d_j = \mathbf{p}_j^T \mathbf{A}\mathbf{p}_j$;
7. *If* $||\mathbf{r}_j|| < \epsilon$, *set* $\mathbf{w}_i = \mathbf{w}_{i,j}$ *for* $i = 1, \ldots, N$. *Else set* $j := j + 1$ *and go to step* i.

The cost of implementing CG is dominated by the matrix-vector multiply at a cost of about $2n^2$ flops in each CG iteration [23]. Thus, as long as the number of CG iterations j is small relative to n, then CG-EnKF will be cheaper to implement than EnKF.

Remark 5. *At iteration k of CG-EnKF, after the CG sampler iterations have stopped, we define $\mathbf{x}_k^{est}$ to be the most recent CG iterate $\mathbf{x}_j$, and $\mathbf{w}_i$ to be the most recent CG sample $\mathbf{w}_{i,j}$ as in step* vii. *The new ensemble is then given by Eq. (7).*

3.1.1 Analysis of the CG Sampler Approximations

In this section, we provide the relevant theory from [21] regarding how well the mean and covariance matrix of the CG samples $\{\mathbf{w}_i\}$ approximate the mean and covariance of $N(\mathbf{0}, \mathbf{A}^{-1})$, the target Gaussian of interest.

In exact arithmetic, CG is guaranteed to find a solution to the $n \times n$ linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$ [or, equivalently, to find the minimizer of the quadratic in Eq. (6)] after a finite number of iterations, and the CG samples will be distributed as $N(\mathbf{0}, \mathbf{A}^{-1})$ when the eigenvalues of $\mathbf{A}$ are distinct and n CG iterations are computed. The reason for this efficiency is that the CG search directions $\{\mathbf{p}_l\}$ are $\mathbf{A}$-conjugate, $\mathbf{p}_l^T \mathbf{A}\mathbf{p}_m = 0$ (see, e.g., [24]).

In finite precision, however, the CG search directions lose conjugacy at some iteration less than n. Nevertheless, CG still finds a solution to $\mathbf{A}\mathbf{x} = \mathbf{b}$ as long as "local conjugacy" of the search directions is maintained [25]. In fact, when the eigenvalues of $\mathbf{A}$ are clustered into j groups, CG finds the approximate solution after j iterations in a j-dimensional *Krylov space*,

$$\mathcal{K}^j(\mathbf{A}, \mathbf{r}_0) := \mathrm{span}(\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \mathbf{A}^2\mathbf{r}_0, ..., \mathbf{A}^{j-1}\mathbf{r}_0).$$

The loss of conjugacy of the search directions is detrimental to the CG sampler (just as it is for iterative eigensolvers), and, without corrective measures, prohibits sampling from the full Gaussian of interest, $N(\mathbf{0}, \mathbf{A}^{-1})$. Nonetheless, the resulting samples $\mathbf{w}_i$ have a realized covariance which is the best j-rank approximation to $\mathbf{A}^{-1}$ (with respect to the 2-norm) in the same j-dimensional Krylov space searched by the CG linear solver.

In order to make the previous discussion more explicit, we establish some notation. If we let $\mathbf{P}_j$ be the $n \times j$ matrix with $\{\mathbf{p}_l\}_{l=0}^{j-1}$ as columns, and $\mathbf{P}_B$ be the $n \times (n-j)$ matrix with $\{\mathbf{p}_l\}_{l=j}^{n-1}$ as columns, then, by conjugacy,

$$\mathbf{D}_n = \begin{pmatrix} \mathbf{D}_j & 0 \\ 0 & \mathbf{D}_B \end{pmatrix} = \begin{pmatrix} \mathbf{P}_j^T \mathbf{A}\mathbf{P}_j & 0 \\ 0 & \mathbf{P}_B^T \mathbf{A}\mathbf{P}_B \end{pmatrix} = \mathbf{P}_n^T \mathbf{A}\mathbf{P}_n$$

is an invertible diagonal matrix with entries $[\mathbf{D}_n]_{ll} = \mathbf{p}_l^T \mathbf{A}\mathbf{p}_l$. Thus

$$\mathbf{A}^{-1} = \mathbf{P}_n \mathbf{D}_n^{-1} \mathbf{P}_n^T = \mathbf{P}_j \mathbf{D}_j^{-1} \mathbf{P}_j^T + \mathbf{P}_B \mathbf{D}_B^{-1} \mathbf{P}_B^T. \tag{8}$$

Now, a CG sample can be written as $\mathbf{w}_i = \mathbf{w}_{i,j} = \mathbf{P}_j \mathbf{D}_j^{-1/2} \mathbf{z}$, where $\mathbf{z} \sim N(\mathbf{0}, \mathbf{I}_j)$ (since we initialize the sampler with $\mathbf{w}_{i,j=0} = \mathbf{0}$). Thus, when the CG sampler terminates after $j < n$ iterations,

$$\mathbf{w}_i \sim N(\mathbf{0}, \mathbf{P}_j \mathbf{D}_j^{-1} \mathbf{P}_j^T). \tag{9}$$

Since the covariance matrix $\mathbf{P}_j\mathbf{D}_j^{-1}\mathbf{P}_j^T$ is singular, the distribution of $\mathbf{w}_i$ is called an *intrinsic Gaussian* in [26].

In exact arithmetic, Eqs. (8) and (9) show that at iteration $j = n$, the CG sampler produces the sample

$$\mathbf{w}_i \ \sim \ N(\mathbf{0}, \mathbf{A}^{-1})$$

as long as $\mathbf{A}$ has n distinct eigenvalues. When the eigenvalues are not distinct, then CG terminates at iteration $j < n$ [25]. In finite precision, the CG search directions $\{\mathbf{p}_j\}$ lose conjugacy at some iteration $j < n$. The rest of this section is devoted to answering the following question: How good is the distribution approximation $N(\mathbf{0}, \mathbf{P}_j\mathbf{D}_j^{-1}\mathbf{P}_j^T)$ to $N(\mathbf{0}, \mathbf{A}^{-1})$ after conjugacy in the CG search directions is lost?

It is well known that at the jth iteration, CG can be used to inexpensively estimate j of the eigenvalues of $\mathbf{A}$, the extreme ones and the well-separated ones [25, 27–29]. The eigenvalue estimates are known as *Ritz values*, and we will denote them by $\theta_1^j < ... < \theta_j^j$. The corresponding eigenvectors can also be estimated by CG, although at a non-negligible cost, using *Ritz vectors*. In exact arithmetic, by the time that CG converges to a solution of $\mathbf{Ax} = \mathbf{b}$ with residual $\mathbf{r}_j = \mathbf{0}$, the Ritz values have already converged to the j extreme and well-separated eigenvalues of $\mathbf{A}$, and the Ritz vectors have converged to an invariant subspace of $\mathbf{A}$ spanned by the corresponding eigenspaces [25].

It can be shown that [21, Theorem 3.1] the nonzero eigenvalues of $\mathrm{Var}(\mathbf{w}_i|\mathbf{b}) = \mathbf{P}_j\mathbf{D}_j^{-1}\mathbf{P}_j^T$ are the reciprocals of the Ritz values $1/\theta_i^j$ (called the *harmonic Ritz values* of $\mathbf{A}^{-1}$ on $\mathcal{K}^j(\mathbf{A}, \mathbf{r}_0)$ [30–32]). The eigenvectors of $\mathrm{Var}(\mathbf{w}_i|\mathbf{b})$ are the Ritz vectors which estimate the corresponding j eigenvectors of $\mathbf{A}$. Thus, when the CG sampler converges with residual $\mathbf{r}_j = 0$, then [21, Remark 4]

$$[\mathbf{A}^{-1} - \mathrm{Var}(\mathbf{w}_i|\mathbf{b})]\mathbf{v} = \mathbf{0} \tag{10}$$

for any $\mathbf{v} \in \mathcal{K}^j(\mathbf{A}, \mathbf{r}_0)$. This shows that $\mathrm{Var}(\mathbf{w}_i|\mathbf{b})$ is the best j-rank approximation to $\mathbf{A}^{-1}$ in the eigenspaces corresponding to the extreme and well-separated eigenvalues of $\mathbf{A}$. In other words, the CG sampler has successfully sampled from these eigenspaces.

Moreover, from Weyl's Theorem [27] and the triangle inequality, it can be shown that $||\mathbf{A}^{-1} - \mathrm{Var}(\mathbf{w}_i|\mathbf{b})||_2$ is at least as large as the largest eigenvalue of $\mathbf{A}^{-1}$ not being estimated, and it can get as large as this eigenvalue plus the error in the Ritz estimates (see [21] for more detail). Like the difficulty faced by iterative eigenproblem solvers, the accuracy of the j-rank approximation $\mathrm{Var}(\mathbf{w}_i|\mathbf{b})$ to $\mathbf{A}^{-1}$ depends on the distribution of the eigenvalues of $\mathbf{A}$.

Loss of conjugacy of the CG search directions occurs at the same iteration j when at least one of the Ritz pairs converge [25], but this can happen before the CG sampler terminates (due to a small residual). As described by Eq. (10), at the iteration when loss of conjugacy occurs, $\mathrm{Var}(\mathbf{w}_i|\mathbf{b})$ is the best approximation to $\mathbf{A}^{-1}$ in the Krylov subspace $\mathcal{K}^j(\mathbf{A}, \mathbf{r}_0)$ which contains the converged Ritz vector(s). Numerical simulations presented in [21] suggest that after loss of conjugacy, continuing to run the CG sampler until the residual is small does not have a deleterious effect on the samples. On the contrary, in the examples considered in [21], the CG sampler continued to sample from new eigenspaces, providing samples with realized covariances which better approximated $\mathbf{A}^{-1}$.

4. NUMERICAL EXPERIMENTS

In this section, we perform tests and comparisons with CG-EnKF. We consider two synthetic examples: the Lorenz 95 system, which is a first-order nonlinear, chaotic ordinary differential equation system that shares some characteristics with weather models; and a two-dimensional heat equation with a forcing term that can be made large-scale.

4.1 Lorenz 95

We begin with the Lorenz 95 test case, introduced in [33], and analyzed in [34]. The model shares many characteristics with realistic atmospheric models and it is often used as a low-order test case for weather forecasting schemes. We use a 40-dimensional version of the model given as follows:

$$\frac{dx_i}{dt} = (x_{i+1} - x_{i-2})x_{i-1} - x_i + 8, \ \ i = 1, 2, ..., 40. \tag{11}$$

The state variables are periodic: $x_{-1} = x_{39}$, $x_0 = x_{40}$, and $x_{41} = x_1$. Out of the 40 model states, measurements are obtained from 24 states via the observation operator $\mathbf{K}_k = \mathbf{K}$, where

$$[\mathbf{K}]_{rp} = \begin{cases} 1, & (r,p) \in \{(3j+i, 5j+i+2)\} \\ 0 & \text{otherwise}, \end{cases} \tag{12}$$

with $i = 1, 2, 3$ and $j = 0, 1, ..., 7$. Thus, we observe the last three states in every set of five.

To generate data, we add Gaussian noise to the model solution with zero mean and covariance $(0.15\sigma_{\text{clim}})^2\mathbf{I}$, where $\sigma_{\text{clim}} = 3.641$ (standard deviation used in climatological simulations). In the filtering methods, we use $\mathbf{C}_{\varepsilon_k^p} = (0.05\sigma_{\text{clim}})^2\mathbf{I}$ as the model error covariance and $\mathbf{C}_{\varepsilon_k^o} = (0.15\sigma_{\text{clim}})^2\mathbf{I}$ as the observation error covariance. As initial guesses in the filtering, we use $\mathbf{x}_0^{\text{est}} = \mathbf{1}$ and $\mathbf{C}_0^{\text{est}} = \mathbf{I}$. For more details about the example, see [8, 9].

We run experiments with varying ensemble size N, and we note that in the linear case, as $N \to \infty$, EnKF converges to EKF (which in the linear case is just KF). When the model $\mathcal{M}$ is nonlinear, however, the approximations of the model covariance $\mathbf{C}_k^p$ computed in step i.2 of Algorithms 1 and 2 (EKF and EnKF, respectively) will be different; specifically, in EKF $\mathbf{C}_k^p$ is obtained from a linearization of $\mathcal{M}$ about $\mathbf{x}_{k-1}^{est}$, whereas in EnKF $\mathbf{C}_k^p$ is computed after the ensemble members have been pushed forward by the model.

CG-EnKF is the ensemble version of the CG variational Kalman filter (CG-VKF), which was introduced in [11]. In implementation, the two algorithms are very similar. Specifically, in CG-VKF the covariance approximation $\mathbf{P}_j\mathbf{D}_j^{-1}\mathbf{P}_j$ defined in (8) is used directly, whereas in CG-EnKF, the ensembles are sampled from $N(\mathbf{x}_k^{est}, \mathbf{P}_j\mathbf{D}_j^{-1}\mathbf{P}_j)$. Moreover, as for EnKF and EKF, in the linear case CG-EnKF converges to the CG-VKF as $N \to \infty$. For completeness, we present CG-VKF now.

Algorithm 6 (CG-VKF). *Select initial guess $\mathbf{x}_0^{est}$ and low-rank covariance approximation $\mathbf{B}_0^{\#} = \mathbf{X}_0\mathbf{X}_0^T$ of $\mathbf{C}_0^{est}$, and set $k = 1$.*

```
1. Compute the evolution model estimate and covariance:
    a. Compute $\mathbf{x}_k^p = \mathcal{M}(\mathbf{x}_{k-1}^{est})$ and the linearization $\mathbf{M}_k$ of $\mathcal{M}$ about $\mathbf{x}_{k-1}^{est}$ defined in (3);
    b. Define $(\mathbf{C}_k^p)^{-1}$ using the matrix inversion lemma (5) with $\mathbf{X}_k = \mathbf{M}_k\mathbf{P}_j\mathbf{D}_j^{-1/2}$;
2. Compute variational Kalman filter and covariance estimates:
    a. Use CG to estimate the minimizer $\mathbf{x}_k^{est}$ of (4) and to compute the low-rank approximation $\mathbf{P}_j\mathbf{D}_j^{-1}\mathbf{P}_j$ of $\nabla^2\ell(\mathbf{x}|\mathbf{y}_k)^{-1} = \mathbf{C}_k^{est}$, where $j$ is the number of CG iterations;
3. Update $k := k + 1$ and return to step i.
```

For comparisons, we plot the relative error

$$[\textbf{relative\_error}]_k = \frac{||\mathbf{x}_k^{\text{est}} - \mathbf{x}_k^{\text{true}}||}{||\mathbf{x}_k^{\text{true}}||}, \tag{13}$$

where, at iteration k, $\mathbf{x}_k^{\text{est}}$ is the filter estimate and $\mathbf{x}_k^{\text{true}}$ is the truth used in data generation. For the ensemble methods, we test ensemble sizes $N = 10$ and $N = 20$. Since ensemble filters are stochastic methods, we show relative errors averaged over 20 repetitions. In CG-based filters, the residual norm stopping tolerance was set to 10^{-6}, and the maximum number of iterations was set to 50. From the results in the top plot in Fig. 1, we see that CG-EnKF outperforms EnKF for small ensemble sizes. When the ensemble size gets larger, CG-EnKF, CG-VKF, and EnKF performances approach each other, as expected. Finally, CG-VKF and EKF perform equally well, although EKF reduces the error faster in early iterations. Nonmonotonicity in the reduction of the error plots is a product of the chaotic nature of the Lorenz 95 model.

In the bottom plot in Fig. 1, we also compare the forecast skills given by different methods, using ensemble sizes $N = (15, 20, 40)$. For this comparison, we compute the following statistics for forecasts launched at every fourth

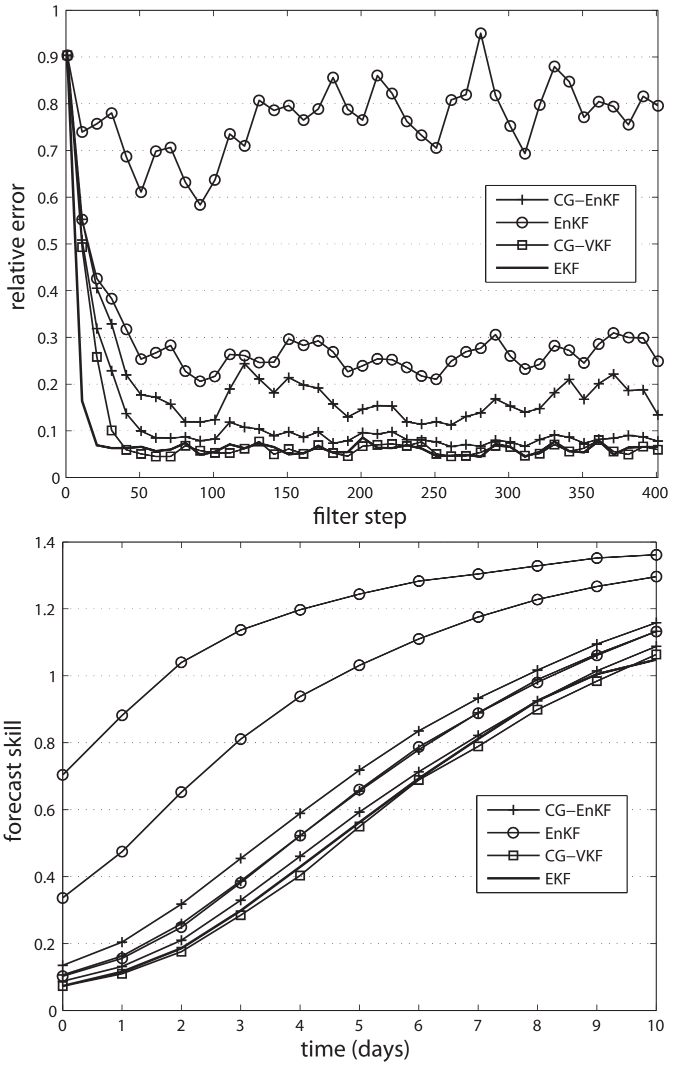

FIG. 1: In the top plot is a comparison of relative errors. For EnKF and CG-EnKF, the upper and lower curves correspond to ensemble sizes of $N = 10$ and 20, respectively. In the bottom plot is a 'forecast skill' comparison. For EnKF and CG-EnKF, the upper, middle, and lower curves correspond to ensemble sizes $N = 15$, 20, and 40, respectively.

filter step, starting from the 64th step (when all filters have converged). Take $j \in \mathcal{I} := \{4i \mid i = 16, 17, \ldots, 100\}$ and define

$$[\mathbf{forecast\_error}_j]_i \quad = \quad \frac{1}{40}\|\mathcal{M}_{4i}(\mathbf{x}_j^{est}) - \mathbf{x}_{j+4i}^{true}\|^2, \quad i = 1, \ldots, 20, \tag{14}$$

where $\mathcal{M}_n$ denotes a forward integration of the model by n time steps. 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 average the forecast accuracy over the 85 forecasts, and define the forecast skill vector as

$$[\mathbf{forecast\_skill}]_i = \frac{1}{\sigma_{\text{clim}}}\sqrt{\frac{1}{85}\sum_{j\in\mathcal{I}}[\mathbf{forecast\_error}_j]_i}, \quad i = 1, \ldots, 20, \tag{15}$$

Again, CG-EnKF outperforms EnKF, especially when N is small. For instance, CG-EnKF with $N = 20$ performs as well as EnKF with $N = 40$. CG-VKF and EKF perform in a similar way.

One might ask which of EKF, CG-EnKF, and CG-VKF is the most desirable in a given situation. In this example, the numerical results suggest that EKF performs best, followed by CG-VKF. However, both EKF and CG-VKF require the Jacobian matrix $\mathbf{M}_k$ defined in (3) and its transpose $\mathbf{M}_k^T$, which cannot be efficiently computed in some large-scale nonlinear examples. In such cases, CG-EnKF is the best approach. In terms of computational cost, for sufficiently large-scale problems (such as the one considered next), the storage and inversion of $n \times n$ covariance matrices required in EKF is prohibitive: n^2 elements must be stored and inversion requires $\mathcal{O}(n^3)$ operations. For CG-EnKF and CG-VKF, nj elements are stored—where j is the number of ensemble members for CG-EnKF and the number of stored CG search directions for CG-VKF—and the matrix inverse computed in (5) requires $\mathcal{O}(j^3)$ operations, which is a significant savings if $j \ll n$.

Finally, we compare CG-EnKF with the limited memory BFGS (LBFGS)-based ensemble Kalman filter of [20]. This method has the same form as Algorithm 3, except that in step *ii*, LBFGS is used in place of CG to compute both the approximate minimizer $\mathbf{x}_k^{est}$ of $\ell(\mathbf{x}|\mathbf{y}_k)$ defined in (4), as well as of the covariance $\mathbf{C}_k^{est} = \nabla^2 \ell(\mathbf{x}|\mathbf{y}_k)^{-1}$. Sampling from $N(\mathbf{0}, \mathbf{B}_k)$, where $\mathbf{B}_k$ is the LBFGS approximation of $\mathbf{C}_k^{est}$, requires some ingenuity, and the interested reader should see [20] for details. For completeness, we present the LBFGS ensemble filter now.

Algorithm 7 (LBFGS-EnKF)**.** *Sample initial ensemble* $\mathbf{x}_{0,i}^{est}$ *for* $i = 1, \ldots, N$ *from* $N(\mathbf{x}_0^{est}, \mathbf{C}_0^{est})$*, and set* $k = 1$.

1. *Same as Algorithm 3.*
2. *Compute a new ensemble using the LBFGS sampler:*
   a. *Use LBFGS to estimate the minimizer* $\mathbf{x}_k^{est}$ *of (4), as well as to compute the new ensemble members* $\mathbf{x}_{k,i}^{est}$ *for* $i = 1, \ldots, N$ *from* $N(\mathbf{x}_k^{est}, \mathbf{B}_k)$*, where* $\mathbf{B}_k$ *is the LBFGS approximation of* $\nabla^2 \ell(\mathbf{x}|\mathbf{y}_k)^{-1} = \mathbf{C}_k^{est}$.
3. *Update* $k := k + 1$ *and return to step* i.

For a fair comparison between the CG and LBFGS ensemble filters, the stopping criteria for LBFGS are the same as for CG. In LBFGS, there are two tuning parameters: the initial inverse Hessian approximation and the number of BFGS vectors that are stored in the algorithm (see [22]). We use a heuristic method for choosing the initial inverse Hessian, as in [20, 22], and to avoid an unfair comparison we store all vectors in the LBFGS iterations. In Fig. 2, we

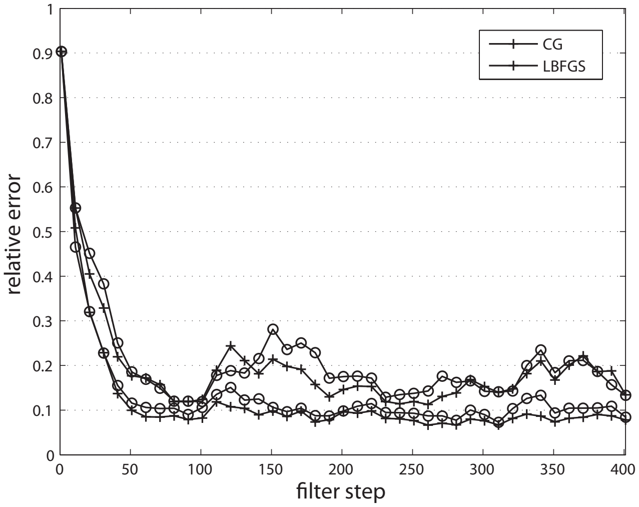

FIG. 2: Comparison of variational ensemble Kalman filters implemented by CG and LBFGS with ensemble sizes $N = (10, 20)$. For LBFGS-EnKF and CG-EnKF, the upper and lower curves correspond to ensemble sizes of $N = 10$ and 20, respectively.

compare the relative errors for ensemble sizes $N = (10, 20)$, averaged over 20 repetitions. In these cases, the CG implementation performs better than the LBFGS implementation. When the ensemble size gets larger, the methods perform equally well.

4.2 Heat Equation

The purpose of this example is to demonstrate CG-EnKF behavior when the dimension is large. The example is linear, so we can directly compare to KF. However, as the dimension of the problem is increased, KF cannot be run due to memory issues. Note that while the example does illustrate computational aspects related to the methods, this system is well-behaved and we cannot conclude much about how the methods work in a high-dimensional chaotic case such as numerical weather prediction.

The model describes heat propagation in a two-dimensional grid and it is written as a partial differential equation:

$$\frac{\partial x}{\partial t} = -\frac{\partial^2 x}{\partial u^2} - \frac{\partial^2 x}{\partial v} + \alpha \exp\left(-\frac{(u-2/9)^2 + (v-2/9)^2}{\sigma^2}\right), \tag{16}$$

where x is the temperature at coordinates u and v over the domain $\Omega = \{(u, v)|u, v \in [0, 1]\}$. The last term in the equation is an external heat source, whose magnitude can be controlled with the parameter $\alpha \geq 0$.

We discretize the model using a uniform $S \times S$ grid. This leads to a linear forward model $\mathbf{x}_{k+1} = \mathbf{M}\mathbf{x}_k + \mathbf{f}$, where $\mathbf{M} = \mathbf{I} - \Delta t\mathbf{L}$, where Δt is the time step, $\mathbf{L}$ is the discrete negative Laplacian, and $\mathbf{f}$ is an external forcing; see [8, 9] for more details. The dimension of the problem can be controlled by changing S. The observation operator $\mathbf{K}$ is defined as in [8, 9], with the measured temperature a weighted average of the temperatures at neighboring points at $S^2/64$ evenly spaced locations.

Data are generated by adding normally distributed random noise to the model state and the corresponding response:

$$\mathbf{x}_{k+1} = \mathbf{M}\mathbf{x}_k + \mathbf{f} + \mathrm{N}[\mathbf{0}, (0.5\sigma_{ev})^2\mathbf{I}] \tag{17}$$

$$\mathbf{y}_{k+1} = \mathbf{K}\mathbf{x}_{k+1} + \mathrm{N}[\mathbf{0}, (0.8\sigma_{obs})^2\mathbf{I}]. \tag{18}$$

In data generation, we use $\alpha = 0.75$ and choose σ_{ev} and σ_{obs} so that the signal to noise ratios at the initial condition, defined by $||\mathbf{x}_0||^2/S^2\sigma_{ev}^2$ and $||\mathbf{K}\mathbf{x}_0||^2/m^2\sigma_{obs}^2$, are both 50. The initial condition for data generation is

$$[\mathbf{x}_0]_{ij} = \exp\left(-(u_i - 1/2)^2 - (v_j - 1/2)^2\right). \tag{19}$$

For the filtering we use a biased model, where the forcing term is dropped by setting $\alpha = 0$. The error covariances used for model and observations are $\sigma_{ev}^2\mathbf{I}$ and $\sigma_{obs}^2\mathbf{I}$, respectively. We start all filters from initial guess $\mathbf{x}_0 = \mathbf{0}$. For ensemble filters, all members are initialized to the same value and for KF we set initial covariance estimate to $\mathbf{C}_0^{est} = \mathbf{0}$.

As our first test, we take $S = 2^j$ and choose $j = 5$, which is the largest integer so that KF can still be computed on a standard desktop computer. Thus, the dimension of the first test was $d = S^2 = 1024$. Then, we compared CG-EnKF and CG-VKF in a case where the dimension is much higher: ($j = 7, d = S^2 = 16, 384$). The stopping tolerance for CG was once again set to 10^{-6} and the number of maximum iterations was set to 20. Also, as above, the CG-EnKF results are averages over 20 repetitions. EnKF results are not included in these experiments because it performs poorly with such small ensemble sizes. In the top plot in Fig. 3, we compare KF, CG-VKF and CG-EnKF with ensemble sizes $N = (10, 20, 50)$, noting again that as N increases, CG-EnKF approaches CG-VKF. In the higher dimensional case, as noted above, KF cannot be used anymore due to memory issues.

5. CONCLUSIONS

The ensemble Kalman filter is a state-space estimation technique approximating the standard extended Kalman filter. In EnKF, an ensemble of approximate states is created at each time point and each member is propagated forward by

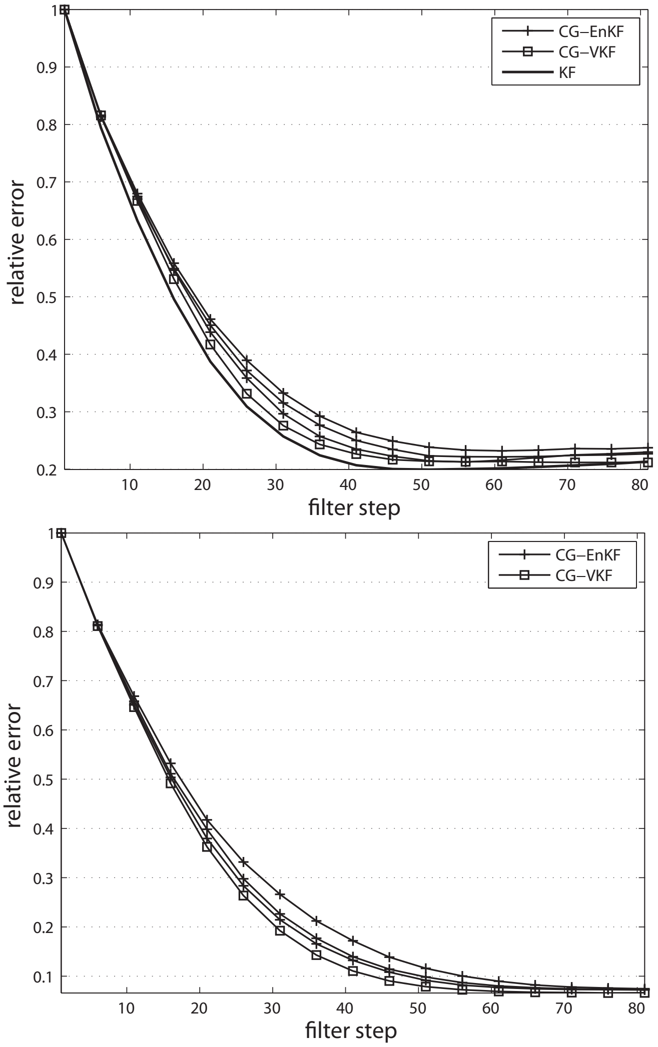

FIG. 3: Performance comparison of KF, CG-VKF, and CG-EnKF with ensemble sizes $N = (10, 20, 50)$ in the case where $d = 1024$ (top) and $d = 16,384$ (bottom). For EnKF and CG-EnKF, the upper, middle, and lower curves correspond to ensemble sizes $N = 10$, 20, and 50, respectively.

the state-space model $\mathcal{M}$ in (1). The covariance of the resulting ensemble is then used within the EKF formulas to yield an efficient approximate filter.

In this paper, we show how the CG sampler of [21] can be applied to a variational formulation of EKF for the creation of a new ensemble at each time point. The use of CG yields a point estimate of the state, and the computed ensemble members are optimal within a certain Krylov subspace. Implementation of the CG sampler requires the addition of only a single inexpensive line of code within CG. We call the resulting algorithm CG-EnKF, and we present an analysis of the accuracy of the CG samples.

We apply CG-EnKF to two examples, one of which is large-scale and linear, and the other small-scale, nonlinear, and chaotic. In both cases, it outperforms standard EnKF, and as the ensemble size increases, the relative error curve for CG-EnKF approaches that of CG-VKF, as the theory suggests. Finally, CG-EnKF compares favorably to the LBFGS-EnKF method of [20].

REFERENCES

1. Kalman, R. E., A new approach to linear filtering and prediction problems, *Trans. ASME–J. Basic Eng., Ser. D*, 82:35–45, 1960.
2. Welch, G. and Bishop, G., An introduction to the Kalman filter, *UNC Chapel Hill, Dept. of Computer Science Tech. Report*, 95-041, 1995.
3. Cane, M. A., Miller, R. N., Tang, B., Hackert, E. C., and Busalacchi, A. J., Mapping tropical Pacific sea level: Data assimilation via reduced state Kalman filter, *J. Geophys. Res.*, 101:599–617, 1996.
4. Dee, D. P., Simplification of the Kalman filter for meteorological data assimilation, *Q. J. R. Meteor. Soc.*, 117:365–384, 1990.
5. Fisher, M., Development of a simplified Kalman filter, *Technical Memorandum 260, European Center for Medium Range Weather Forecasting*, 1998.
6. Gejadze, I. Y., Le Dimet, F.-X., and Shutyaev, V., On analysis error covariances in variational data assimilation, *SIAM J. Sci. Comput.*, 30:1847–1874, 2008.
7. Fisher, M. and Andersson, E., Developments in 4D-var and Kalman filtering, *Technical Memorandum 347, European Center for Medium Range Weather Forecasting*, 2001.
8. Auvinen, H., Bardsley, J. M., Haario, H., and Kauranne, T., Large-scale Kalman filtering using the limited memory BFGS method, *Electr. Trans. Numer. Anal.*, 35:217–233, 2009.
9. Auvinen, H., Bardsley, J. M., Haario, H., and Kauranne, T., The variational Kalman filter and an efficient implementation using limited memory BFGS, *Int. J. Numer. Methods Fluids*, 64(3):314–335, 2010.
10. Veersé, F., Variable-storage quasi-Newton operators for modeling error covariances, *Proc. of the Third WMO International Symp. on Assimilation of Observations in Meteorology and Oceanography*, 7–11, Quebec City, Canada, 1999.
11. Bardsley, J. M., Parker, A., Solonen, A., and Howard, M., Krylov space approximate Kalman filtering, *Numer. Linear Algebra Appl.*, DOI: 10.1002/nla.805, 2011.
12. Zupanski, M., Maximum likelihood ensemble filter: Theoretical aspects, *Mont. Weather Rev.*, 133:1710–1726, 2005.
13. Evensen, G., Sequential data assimilation with a non-linear quasi-geostrophic model using Monte Carlo methods to forecast error statistics, *J. Geophys. Res.*, 99:10143–10162, 1994.
14. Evensen, G., *Data Assimilation: The Ensemble Kalman Filter*, Springer, Berlin, 2007.
15. Sacher, W. and Bartello, P., Sampling errors in ensemble Kalman filtering. Part I: Theory, *Mon. Weather Rev.*, 136:3035–3049, 2008.
16. Meng, Z. and Zhang, F., Tests of an ensemble Kalman filter for mesoscale and regional-scale data assimilation. Part II: Imperfect model experiments, *Mon. Weather Rev.*, 135:1403–1423, 2007.
17. Li, H., Kalnay, E. and Takemasa, M., Simultaneous estimation of covariance inflation and observation errors within an ensemble Kalman filter. *Q. J. R. Meteorol. Soc.*, 135:523–533, 2009.
18. Whitaker, J. S. and Hamill, T. M., Ensemble data assimilation without perturbed observations, *Mon. Weather Rev.*, 130:1913–1924, 2002.
19. Ott, E., Hunt, B. R., Szunyogh, I., Zimin, A. V., Kostelich, E. J., Corazza, M., Kalnay, E., Pati, O., and Yorke, J. A., A local ensemble Kalman filter for atmospheric data assimilation, *Tellus A*, 56:415–428, 2004.
20. Solonen, A., Haario, H., Hakkarainen, J., Auvinen, H., Amour, I., and Kauranne, T., Variational ensemble Kalman filtering using limited memory BFGS, submitted, 2011.
21. Parker, A. and Fox, C., Sampling gaussian distributions in Krylov spaces with conjugate gradients, *SIAM J. Sci. Comput.*, 34(3):B312-B334, 2012.
22. Nocedal, J. and Wright, S., *Numerical Optimization*, Springer, New York, 2000.
23. Watkins, D., *Fundamentals of Matrix Computations*, 2nd Edition Wiley, New York, 2002.
24. Golub, G. H. and Van Loan, C. F., *Matrix Computations*, The Johns Hopkins University Press, Baltimore, 3rd ed., 1996.
25. Meurant, G., *The Lanzcos and Conjugate Gradient Algorithms*, SIAM, Philadelphia, 2006.
26. Rue, H. and Held, L., *Gaussian Markov Random Fields: Theory and Applications*, Chapman & Hall/CRC, New York, 2005.
27. Parlett, B., *Symmetric Eigenvalue Problem*, Prentice Hall, Englewood Cliffs, NJ, 1980.

28. Saad, Y., *Numerical Methods for Large Eigenvalue Problems,* SIAM, Philadelphia, 2011.

29. Sleijpen, G. L. C. and Van Der Sluis, A., Further results on the convergence behavior of conjugate-gradients and Ritz values, *Linear Alg. Appl.*, 246:233–278, 1996.

30. Morgan, R. B., Computing interior eigenvalues of large matrices, *Linear Alg. Appl.*, 154-156:289–309, 1991.

31. Paige, C., Parlett, B. N., and van der Vosrst, H. A., Approximate solutions and eigenvalue bounds from Krylov subspaces, *Num. Linear Alg. Appl.*, 2(2):115–133, 1995.

32. Sleijpen, G. and van der Eshof, J., Accurate approximations to eigenpairs using the harmonic Rayleigh-Ritz method, *Preprint 1184, Department of Mathematics, University of Utrecht*, 2001.

33. Lorenz, E. N., Predictability: A problem partly solved, *Proc. of the Seminar on Predicability, European Center on Medium Range Weather Forecasting*, 1:1–18, 1996.

34. Lorenz, E. N. and Emanuel, K. A., Optimal sites for supplementary weather observations: Simulation with a small model, *J. Atmosp. Sci.*, 55:399–414, 1998.