Stabilized BFGS approximate Kalman filter

The MIT Faculty has made this article openly available. Please share how this access benefits you. Your story matters.

<table>
<thead>
<tr>
<th>Citation</th>
<th>Bibov, Alexander, Heikki Haario, and Antti Solonen. “Stabilized BFGS Approximate Kalman Filter.” IPI 9, no. 4 (October 2015): 1003–1024. © 2015 American Institute of Mathematical Sciences</th>
</tr>
</thead>
<tbody>
<tr>
<td>As Published</td>
<td><a href="http://dx.doi.org/10.3934/ippi.2015.9.1003">http://dx.doi.org/10.3934/ippi.2015.9.1003</a></td>
</tr>
<tr>
<td>Publisher</td>
<td>American Institute of Mathematical Sciences (AIMS)</td>
</tr>
<tr>
<td>Version</td>
<td>Final published version</td>
</tr>
<tr>
<td>Accessed</td>
<td>Mon Apr 25 03:38:19 EDT 2016</td>
</tr>
<tr>
<td>Citable Link</td>
<td><a href="http://hdl.handle.net/1721.1/100416">http://hdl.handle.net/1721.1/100416</a></td>
</tr>
<tr>
<td>Terms of Use</td>
<td>Article is made available in accordance with the publisher's policy and may be subject to US copyright law. Please refer to the publisher's site for terms of use.</td>
</tr>
<tr>
<td>Detailed Terms</td>
<td></td>
</tr>
</tbody>
</table>
STABILIZED BFGS APPROXIMATE KALMAN FILTER

ALEXANDER BIBOV
LUT Mafy - Department of Mathematics and Physics
Lappeenranta University Of Technology, P.O. Box 20 FI-53851, Finland

HEIKKI HAARIO AND ANTTI SOLONEN
Lappeenranta University of Technology, Department of Mathematics and Physics
Lappeenranta, P.O. Box 20 FI-53851, Finland
and
Massachusetts Institute of Technology, Department of Aeronautics and Astronautics
77 Massachusetts Ave, Cambridge, MA 02139, USA

(Communicated by Jari Kaipio)

ABSTRACT. The Kalman filter (KF) and Extended Kalman filter (EKF) are well-known tools for assimilating data and model predictions. The filters require storage and multiplication of $n \times n$ and $n \times m$ matrices and inversion of $m \times m$ matrices, where $n$ is the dimension of the state space and $m$ is dimension of the observation space. Therefore, implementation of KF or EKF becomes impractical when dimensions increase. The earlier works provide optimization-based approximative low-memory approaches that enable filtering in high dimensions. However, these versions ignore numerical issues that deteriorate performance of the approximations: accumulating errors may cause the covariance approximations to lose non-negative definiteness, and approximative inversion of large close-to-singular covariances gets tedious. Here we introduce a formulation that avoids these problems. We employ L-BFGS formula to get low-memory representations of the large matrices that appear in EKF, but inject a stabilizing correction to ensure that the resulting approximative representations remain non-negative definite. The correction applies to any symmetric covariance approximation, and can be seen as a generalization of the Joseph covariance update.

We prove that the stabilizing correction enhances convergence rate of the covariance approximations. Moreover, we generalize the idea by the means of Newton-Schultz matrix inversion formulae, which allows to employ them and their generalizations as stabilizing corrections.

1. Introduction. In recent years several approximations for KF and EKF have been proposed to reduce the computational complexity in large-scale cases, such as arise in robotics or data assimilation tasks. One of the simplest ways to reduce matrix storage requirements is to replace high-order covariances with their sampled estimates, which leads to the large family of ensemble Kalman filters (EnKF), first proposed in [18]. The advantageous property of EnKF formulations is their parallel nature as the ensemble members can be processed independently from each other.

Key words and phrases. Extended Kalman filter, approximate Kalman filter, low-memory storage, BFGS update, observation-deficient inversion, chaotic dynamics.

The present study was supported by Finnish Academy Centre of Excellence in Inverse Problems, project 134937.
The quality of EnKF estimates usually depends on the way the ensemble members are generated and on their ability to capture the essential part of the state dynamics.

An alternative approach is exploited by the Reduced Rank Kalman filter and the Reduced Order Extended Kalman filter that are based on projection of the full state space onto its subspace of a smaller dimension (see e.g. [13], [10], [34], [20]). The success of this approach is based on judicious choice of reduction strategy applied to the state space.

Another emerging solution to perform large-scale data assimilation similar to approximate Kalman filter is the Weak-Constraint 4D-VAR analysis, which was firstly proposed in [35] and later suggested as a replacement for the 4D-VAR, the dominating method for data assimilation in weather prediction [32], [22]. A very similar tool was also applied in oceanography, where the data assimilation was used in a study of the German bight [5].

One of the possible EKF approximations can be obtained by replacing the explicit algebraic formulas used to calculate posterior state vector and covariance matrix with a maximum posterior optimization problem. This allows to formulate Variational Kalman filter (VKF), which avoids undesired explicit storage of large matrices by employing Quasi-Newton optimization algorithms [3], [4]. The VKF performance depends on the smallest eigenvalue magnitude of the estimate covariance matrix. If the matrix degenerates to singularity, the VKF must be regularized by adjusting the model error covariance and the observation noise level. Another general problem shared by the variational formulations is necessity to invert the forecast estimate covariance matrix (i.e. prediction covariance), which sometimes is close to singular, so that the inversion operation should be replaced by pseudoinversion, whereas this may not be easily implemented in large-scale.

A variant of approximation for the EKF can be derived by directly replacing the high-order matrices in the EKF formulas with their low-memory approximative versions. This replacement is done by introducing auxiliary optimization problems, which thereafter are solved using a Quasi-Newton or conjugate-gradient optimization methods. In contrast to the variational formulations mentioned above the direct approximation does not act over approximate inversion of the full forecast estimate covariance. Instead, the inversion is applied to the forecast estimate covariance projected onto the observation space. This is advantageous when the number of observations is small compared to the dimension of dynamical system. Hence, these direct approximations possess potentially lucrative properties, not preserved by their variational counterparts.

Several algorithms based on the direct approximation idea have been discussed in [2] and [4]. However, the previously proposed methods provide no guarantee that the approximated covariance matrix remains non-negative definite. This results in accumulation of approximation errors. In addition, the optimization methods that produce low-memory covariance approximations often rely on non-negative definiteness of the cost function’s Hessian [25]. Therefore, when such methods are combined with direct EKF formulae approximations that do not retain non-negative definiteness property, the resulting low-memory representations may be inadequate.

The problem of poor numerical estimation of the analysis covariance matrix, which in practical implementations may become indefinite due to the round-off errors, was firstly addressed by Joseph and prompted him to reformulate the covariance update (see [30]). However, Joseph’s arrangement requires almost the order of magnitude more arithmetical operations than the original algorithm by Kalman.
An alternative approach, which does not impose extra arithmetical complexity, was suggested by Potter [12]. The key idea of this method is to factorize the analysis covariance matrix and to perform the updates on the square root, which eventually leads to more precision-tolerant numerics. An exhaustive overview of this idea together with its further extensions is given in [7]. However, both the Joseph update and the square root filters inspired by Potter have storage requirements similar to that of the classical sequential Kalman filter. Therefore, neither of them address the problems that arise in case when the state space dimension is large.

In this paper we propose stabilized approximation for the EKF, which corrects the previously suggested approximative formulas with no negative impact on their limited-memory design. We prove that with this correction applied, the approximative estimate covariance matrix remains non-negative definite regardless of the approximation quality. Furthermore, we show that the stabilizing correction ensures quadratic convergence rate, while the approximations previously suggested in [2] and [4] can only guarantee linear convergence rate. We also propose a possible generalization of the suggested stabilizing correction achieved by using Newton-Schultz inverse matrix recurrent formula. This approach can be even further improved by using Schultz-type recurrent formulas of the higher orders based on Chebyshev polynomials [27].

This paper contains three sections apart from the introduction. In the methods section we discuss the direct EKF approximation [2] based on L-BFGS unconstrained optimizer (see e.g. [25]). In this section we also introduce the family of stabilizing corrections and prove non-negative definiteness of the corresponding estimate covariance approximations.

The last two sections are devoted to examples. In this paper we consider application of the proposed method for a filtering task formulated on top of an observation-deficient non-linear large-scale chaotic dynamical system. As an example of such system we employ the two-layer Quasi-Geostrophic model (QG-model) [19], which is one of the common benchmarks employed to estimate performance of data assimilation algorithms [21].

In the last section approximative EKF with stabilized correction is compared with direct L-BFGS EKF approximation described in [2]. By running these filtering algorithms for the QG-model initialized at different resolutions it is demonstrated that direct approximation fails when assimilation model dimension increases. In all experiments the filters are used to deduce full state of the QG-model based on deficient observation data, which constitutes a challenging ill-posed inverse problem. In addition, the dimension of the model is kept low enough to allow use of the exact EKF formulas, so that the EKF estimates can be employed as a baseline for performance analysis of Stabilized Approximate EKF (SA-EKF) proposed in this paper. The criteria used to measure quality of the estimates are the root mean square error (RMS error) and the forecast skill.

Finally, the paper is wrapped out with a short conclusion part based on the experiment results. In this section we summarize the properties and point out the benefits of the stabilizing correction. The paper has an appendix section, which comprises self-contained derivation of matrix approximation formulas based on BFGS update.

2. Methods. We begin our discussion of approximative Kalman filters by considering the following coupled system of stochastic equations:

\[ x_{k+1} = M_k(x_k) + \epsilon_k, \]
In the first equation \( x_k \) denotes \( n \times 1 \) state vector at time instance \( k \), \( M_k \) is transition operator from time instance \( k \) to time instance \( k + 1 \), \( \epsilon_k \) is a random \( n \times 1 \) vector representing modelling error of the transition operator \( M_k \). In the second equation \( y_{k+1} \) denotes \( m \times 1 \) observed vector at time instance \( k + 1 \), \( H_{k+1} \) is observation operator, and \( \eta_{k+1} \) is an \( m \times 1 \) random vector representing the observation error.

From now on we assume that the error terms \( \epsilon_k \) and \( \eta_{k+1} \) are mutually independent, with zero mean and covariance matrices equal \( C_{\epsilon_k} \) and \( C_{\eta_k} \) respectively.

Following general formulation of a filtering problem the task is to derive an estimate of system state \( x_{k+1} \) at time instant \( k + 1 \) and compute the corresponding estimate covariance matrix \( C_{\text{est}}^{k+1} \) given the estimate \( x_{k} \) of \( x_k \) together with its analysis covariance \( C_{\text{est}}^k \), observed data \( y_{k+1} \), and covariance matrices \( C_{\epsilon_k} \) and \( C_{\eta_k} \) of the corresponding error terms. Putting it another way, the problem in consideration is to invert deficient observation data based on previously made analytical prediction and related statistical parameters. Since the problem can be significantly ill-posed, the quality of resulting estimates is expected to depend on particular properties of observation operator and on the noise levels.

2.1. Extended Kalman filter. Kalman filter is the standard approach taken for the formulated inverse problem. This is an optimal decision making tool which couples the information from model prediction and observed data and produces the estimate for the next state. In a case in which \( M_k \) and \( H_k \) are linear, the Kalman filter estimate of \( x_{k+1} \) is proved to be optimal in the sense it yields the minimal variance unbiased linear estimator [23]. Below we formulate the basic Kalman filter algorithm with linear transition operator \( M_k \) and linear observation operator \( H_k \) (the notation here is altered due to the assumed linearity).

**Algorithm I. Kalman Filter.**

**INPUT:** \( M_k, H_{k+1}, x_{k+1}^{\text{est}}, C_{\text{est}}^k, y_{k+1}, C_{\epsilon_k}, C_{\eta_k} \).

**OUTPUT:** \( x_{k+1}^{\text{est}}, C_{\text{est}}^{k+1} \).

(i) Prediction step

a) Compute state prediction \( x_{k+1}^p = M_k x_k^{\text{est}} \);

b) Compute covariance matrix for predicted state

\[
C_{k+1}^p = M_k C_k^{\text{est}} M_k^T + C_{\epsilon_k}.
\]

(ii) Correction step

a) Compute the Kalman gain

\[
G_{k+1} = C_{k+1}^p H_{k+1}^T \left( H_{k+1} C_{k+1}^p H_{k+1}^T + C_{\eta_k} \right)^{-1}.
\]

b) Compute the next state estimate

\[
x_{k+1}^{\text{est}} = x_{k+1}^p + G_{k+1} \left( y_{k+1} - H_{k+1} x_{k+1}^p \right);
\]

c) Compute covariance of the next state estimate

\[
C_{k+1}^{\text{est}} = C_{k+1}^p - G_{k+1} H_{k+1} C_{k+1}^p.
\]

The extension of the algorithm to nonlinear case is provided by the extended Kalman filter (EKF). It is obtained by using the nonlinear model in prediction step (i. a), and in simulated observations (step ii.b), i.e. in these steps \( M_k \) and \( H_{k+1} \) should
be replaced with $\mathcal{M}_k$ and $\mathcal{H}_{k+1}$ respectively. The rest of the algorithm employs Jacobians instead of nonlinear operators $\mathcal{M}_k$ and $\mathcal{H}_k$:

\[
M_k = \frac{\partial \mathcal{M}_k(x_{\text{est}}^k)}{\partial x}, \quad H_k = \frac{\partial \mathcal{H}_k(x_{\text{est}}^k)}{\partial x}.
\]

For large models the derivatives have to be implemented as special code-level subroutines to avoid explicit matrix storage. The routines used to compute Jacobian matrices $M_k$ and $H_k$ are called tangent-linear codes. Subroutines implementing the corresponding adjoint operators $M^T_k$ and $H^T_k$ are called adjoint codes. To separate notation used in algorithm I hereinafter we will use the superscript $(\cdot)^{TL}$ to denote a tangent-linear code and the superscript $(\cdot)^{AD}$ to emphasize an adjoint code.

2.2. L-BFGS low-memory matrix approximation. The EKF implementation complexity is estimated as $O(n^3 + m^3)$ multiplications. Namely, EKF requires to store and multiply $n \times n$, $m \times m$ and $n \times m$ matrices, and to invert an $m \times m$ matrix. This may become prohibitively expensive when the state space dimension $n$ or the observation space dimension $m$ increases. In case of large $n$ or $m$ EKF can be approximated by replacing the impracticable high-order matrices with their approximative low-memory representations. A number of different low-memory matrix approximations have been suggested (see e.g. [2], [3], [4]). In this paper the desired approximation is attained by using the BFGS update \[25\]. The BFGS update is a rank-two update formula addressed by a number of Quasi-Newton unconstraint optimizers. The present study employs Limited-Memory BFGS iterative optimization method (L-BFGS) that uses the BFGS update to produce low-memory approximation for Hessian of the cost function. More precisely, the following quadratic minimization problem is considered:

\[
(4) \quad \arg\min_x \frac{1}{2} (Ax, x) - (b, x),
\]

where $A$ is an $n$-by-$n$ positive-definite matrix, $b$ is an $n$-by-1 vector. When applied to the problem (4), the L-BFGS iterations generate series of vector pairs $(s_1, y_1), \ldots, (s_k, y_k)$ such that $As_i = y_i$, $i = 1, \ldots, k$. As discussed in [9] this iteration history can be then used to construct low-memory approximation for the matrix $A$ or $A^{-1}$. This idea allows us to reformulate the EKF by explicitly replacing the prohibitively high-order matrices with their L-BFGS low-memory approximations. In this paper we denote such approximations of the problematic matrices by adding symbol “˘” on top of the regular matrix notation introduced in algorithm I. For instance, an approximation of the estimate covariance matrix $C_{\text{est}}^k$ introduced in algorithm I is denoted by $\hat{C}_{\text{est}}^k$, whereas the nature of the approximation will be clarified by the context.

Hereby, the algorithm that we call L-BFGS Approximate EKF can be formulated as follows:

**Algorithm II. L-BFGS Approximate Extended Kalman Filter.**

**INPUT:** $\mathcal{M}_k$, $M_k^{TL}$, $M_k^{AD}$, $\mathcal{H}_{k+1}$, $H_{k+1}^{TL}$, $H_{k+1}^{AD}$, $x_{\text{est}}^k$, $y_{k+1}$, $C_{\epsilon k}$, $C_{\eta k+1}$.

**OUTPUT:** $x_{\text{est}}^{k+1}$, $\hat{C}_{\text{est}}^{k+1}$.

(i) Prediction step

a) Compute state prediction $x_{k+1}^p = \mathcal{M}_k(x_{\text{est}}^k)$;

b) On the code level define prior covariance operator

\[
C_{k+1}^p = M_k^{TL}\hat{C}_{\text{est}}^k M_k^{AD} + C_{\epsilon k}.
\]
(ii) Correction step

a) Define optimization problem of type (4) assuming
\[ A = H_{k+1}^{TL} C_{k+1}^p H_{k+1}^{AD} + C_{q_{k+1}}, \quad b = y_{k+1} - H_{k+1}^{TL} x_{k+1}^p. \]
Set \( x^* \) to be the minimizer of this problem and \( B^* \) to be the L-BFGS approximation of the matrix \( A^{-1} \).

b) Update the state estimate
\[ x_{k+1}^{est} = x_{k+1}^p + C_{k+1}^p H_{k+1}^{AD} x^*. \]

c) Define optimization problem of type (4) assuming
\[ A = C_{k+1}^p - C_{k+1}^p H_{k+1}^{AD} B^* H_{k+1}^{TL} C_{k+1}^p, \quad b = 0. \]
Update estimate covariance matrix by assigning \( \hat{C}_{k+1}^{est} \) to the L-BFGS approximation of \( A \).

The outlined algorithm defines one iteration of the L-BFGS approximate EKF. The step (ii. c) can be modified by assuming that vector \( b \) is a normally distributed random variable with zero mean and identity covariance [4].

The step (i. b) on the code level must be implemented as a subroutine. This means that the high order matrices from algorithm I either get implemented on the code level as memory-efficient subroutines (assuming that such implementation is available) or represented by the means of iteration history generated by L-BFGS. Since the number of L-BFGS iterations is usually much smaller than the order of target Hessian matrix [25], the storage requirements are notably reduced with respect to that of previously discussed algorithm I.

It should also be mentioned that in algorithm II high order matrix multiplications are never actually done and either performed implicitly in an efficient way by the subroutines containing the corresponding TL- and AD-codes or performed based on L-BFGS iteration history by the procedure called two-loop recursion (see [9]).

This wraps out the overview of the previously proposed methods that were used as the basis for the algorithm proposed in this paper. In the next section containing the main part of the study we outline the problematic parts of algorithm II and provide an arrangement to treat them.

2.3. Stabilized approximate extended Kalman filter. The main pitfall in algorithm II described in the last section is that the matrix \( A \) in step (ii. c) relies on approximative matrix \( B^* \) and thus does not guarantee non-negative definiteness of the result. This drawback violates the assumptions beyond the BFGS update formula, which may lead to an invalid low-memory BFGS approximation for the estimate covariance \( \hat{C}_{k+1}^{est} \). Furthermore, this error migrates to the next iteration of the filter and potentially affects the prior covariance in step (i. b). Hence, the L-BFGS approximation in step (ii. a) becomes incorrect and algorithm II can be potentially vulnerable to accumulated errors.

In this paper we present correction for the step (ii. c) which eliminates the discussed vulnerabilities. This correction is based on the following simple lemma.

**Lemma 2.1.** Let \( A = H_{k+1}^{TL} C_{k+1}^p H_{k+1}^{AD} + C_{q_{k+1}} \), where \( C_{k+1}^p \) is computed in accordance with step (i.) of algorithm II and \( \hat{C}_{k}^{est} \) is assumed to be nonnegative definite. Then for an arbitrary symmetric matrix \( B^* \) the matrix
\[ \hat{C}_{k+1}^{est} = C_{k+1}^p - C_{k+1}^p H_{k+1}^{AD} (2I - B^* A) B^* H_{k+1}^{TL} C_{k+1}^p, \]
is non-negative definite and \( \hat{C}_{k+1}^{est} \) is equal to \( C_{k+1}^{est} \) when \( B^* = A^{-1} \) (see algorithm I, step (ii. c)). Here \( I \) denotes the identity matrix of respective dimension.
Proof. For conciseness we denote \( C = C^p_{k+1}, H = H^T_{k+1}, R = C_{nk+1}, G = CH^TB^* \).

The non-negative definiteness of \( \hat{C}^\text{est}_{k+1} \) is proved by the following sequence of obvious matrix equalities:

\[
\hat{C}^\text{est}_{k+1} = C - CH^T (2I - B^*A) B^*HC = \\
= C - CH^T (B^* - (B^*A - I)B^*) HC = \\
= C - CH^T (B^*HC - (B^*A - I)B^*HC) = \\
= C - CH^TB^*HC + CH^T(B^*A - I)B^*HC = \\
= C - GHC + CH^T(B^*A - I)G^T = \\
= C - GHC + (CH^TB^* (HCH^T + R) - CH^T) G^T = \\
= C - GHC + (GHC^T + GR - CH^T) G^T = \\
= C - GHC - CH^T G^T + GHC^T G^T + GRG^T = \\
= (I - GH)C(I - GH)^T + GRG^T.
\]

Obviously the matrix \((I - GH)C(I - GH)^T + GRG^T\) is non-negative definite.

The identity \( \hat{C}^\text{est}_{k+1} = C^\text{est}_{k+1} \) when \( B^* = A^{-1} \) can be easily verified by plugging the matrix \( A^{-1} \) in the place of \( B^* \).

When \( B^* = A^{-1} \) the above sequence of matrix equalities reduces to derivation of the Joseph stabilized covariance update [8], [12] extensively employed in the past to circumvent the problem of accumulated errors in Kalman filter. The main advantage over the original Joseph’s approach is that \( B^* \) is allowed to be any symmetric approximation of \( A^{-1} \), which enables the use of a wide variety of methods approximating \( A^{-1} \). For instance, stabilizing correction allows usage of SR-1 approximation [26] in place of \( B^* \), while L-BFGS EKF suggested in [2] assumed that \( B^* \) was at least non-negative definite (even in this case giving no guarantee that corresponding estimate covariance \( \hat{C}^\text{est}_{k+1} \) was proper covariance matrix with no negative eigenvalues).

In accordance with the claim of lemma 2.1 we can correct the step (ii. c) of algorithm II. We call the updated algorithm Stabilized Approximate EKF (SA-EKF).

Our next goal is to show that the suggested stabilizing correction (5) actually converges to \( C^\text{est}_{k+1} \) when \( B^* \) approaches \( A^{-1} \). Furthermore, we will show that the rate of convergence for the stabilized approximation (5) is quadratic, whereas the standard approximation employed by algorithm II guarantees only linear convergence rate.

Lemma 2.2. Suppose that the assumptions made in lemma 2.1 hold. Then the stabilizing approximation (5) converges to \( C^\text{est}_{k+1} \) as \( B^* \) approaches \( A^{-1} \). Furthermore, the following inequalities hold:

\[
(6) \quad \|\hat{C}^\text{est}_{k+1} - C^\text{est}_{k+1}\|_{F^T} \leq \|A\|\|H^T_{k+1}C^p_{k+1}\|^2_{F^T}\|B^* - A^{-1}\|^2.
\]

\[
(7) \quad \|\hat{C}^\text{est}_{k+1} - C^\text{est}_{k+1}\| \leq \|A\|\|H^T_{k+1}C^p_{k+1}\|^2\|B^* - A^{-1}\|^2.
\]

Here \( \cdot \) \( F^T \) denotes Frobenius norm and \( \cdot \) \( \| \| \) denotes matrix 2-norm.

Proof. We start by considering an auxiliary statement: for any matrices \( W \) and \( P \) of consistent dimensions \( \|W^TP\|_{F^T} \leq \|P\|\|W\|^2_{F^T} \). Indeed, it is easy to verify
that \( \|WT PW\|_F^2 = \sum_{i,j} |(PW^i, W^j)|^2 \), where \( W^i \) denotes the \( i \)-th column of matrix \( W \) and operator \((\cdot, \cdot)\) is convenience notation for the inner product. Hence, recalling the Cauchy-Schwarz inequality we see that
\[
\|WT PW\|_F^2 \leq \sum_{i,j} \|PW^i\|_F^2 \|W^j\|_F^2 \\
\leq \sum_{i,j} \|P\|^2 \|W^i\|_F^2 \|W^j\|_F^2 \\
= \|P\|^2 \|W\|_F^4.
\]
Therefore, we proved that \( \|WT PW\|_F^2 \leq \|P\|^2 \|W\|_F^4 \) and thus taking the square root of both sides of this inequality we arrive to the desired statement.

Our next goal is to estimate discrepancy between \( \hat{C}_{k+1}^{\text{est}} \) and \( C_{k+1} \):
\[
\|\hat{C}_{k+1}^{\text{est}} - C_{k+1}\|_F = \|C_{k+1}H\hat{A}_k^P (2B^* - B^*AB^* - A^{-1})H_{k+1}^{TL}C_k^p\|
\]
Recalling the auxiliary inequality \( \|WT PW\|_F \leq \|P\|\|W\|_F^2 \) and assigning \( P = 2B^* - B^*AB^* - A^{-1}, W = H_{k+1}^{TL}C_{k+1} \) we arrive to the following estimate:
\[
\|\hat{C}_{k+1}^{\text{est}} - C_{k+1}\|_F \leq \|H_{k+1}^{TL}C_{k+1}^p\|_F^2 \|2B^* - B^*AB^* - A^{-1}\|.
\]
We now need to derive an estimate for \( \|2B^* - B^*AB^* - A^{-1}\| \). Let us denote \( B^* = A^{-1} + \Delta \). Hence, plugging \( B^* \) back in (8) we see that
\[
\|2B^* - B^*AB^* - A^{-1}\| = \|A\|\|\Delta\|\|^2 = \|A\|\|B^* - A^{-1}\|^2.
\]
Putting together (8) and (9) we arrive to the desired inequality (6). Inequality (7) is proved analogically following the properties of matrix norm.

**Lemma 2.3.** Approximation \( \hat{C}_{k+1}^{\text{est}} \) in the step (ii. c) of algorithm II converges to \( C_{k+1}^{\text{est}} \) when \( B^* \) approaches \( A^{-1} \). Furthermore, the following inequalities hold:
\[
\|\hat{C}_{k+1}^{\text{est}} - C_{k+1}\|_F \leq \|H_{k+1}^{TL}C_{k+1}^p\|_F^2 \|B^* - A^{-1}\|.
\]
\[
\|\hat{C}_{k+1}^{\text{est}} - C_{k+1}\|_F \leq \|H_{k+1}^{TL}C_{k+1}^p\|_F^2 \|B^* - A^{-1}\|.
\]
**Proof.** The proof is similar to the proof of lemma 2.2.

The claims of lemmas 2.1 and 2.2 can be generalized. Moreover, if some additional assumptions are made it is possible to show that stabilizing correction (5) improves initial approximation \( B^* \) of \( A^{-1} \).

Earlier in this section we have shown that approximate analysis covariance \( C_{k+1}^{\text{est}} \) from algorithm II has better numerical properties if approximation \( B^* \) of the matrix \( A^{-1} \) from this algorithm is replaced by \( (2I - B^*A)B^* \). Hence, the question arises if this replacement term as well as the nice numerical features it brings to the covariance approximation can be described more generally. We begin by considering the following sequence of matrix valued functions \( \mathcal{F}_n(X) \) defined by recurrent relation:
\[
\mathcal{F}_n(X) = 2\mathcal{F}_{n-1}(X) - \mathcal{F}_{n-1}(X) A\mathcal{F}_{n-1}(X),
\]
where \( A \) is defined in the same way as in lemma 2.1, \( X \) is an arbitrary symmetric matrix of the same dimension as \( A \), and the recursion is initialized by \( \mathcal{F}_1(X) = X \). It is easy to verify by induction that \( A^{-1} \) is the fixed point of function \( \mathcal{F}_n(X) \), i.e. \( \mathcal{F}_n(A^{-1}) = A^{-1} \), \( n \geq 1 \) and that when \( n = 2 \) and \( X = B^* \), the expression for \( \mathcal{F}_n \) yields exactly the stabilizing replacement \( 2B^* - B^*AB^* \) employed in (5). The next
Lemma 2.4. Suppose that assumptions made in lemma 2.1 are met. Let $F_n(X)$ be a matrix-valued function sequence generated by the recurrent relation (12). Then for any symmetric $X$ the following matrix is nonnegative definite:

\[
\hat{C}_{k+1}^\text{est} (n) = C_{k+1}^p - C_{k+1}^p H_{k+1}^{AD} F_n(X) H_{k+1}^{TL} C_{k+1}^p, \quad n \geq 2.
\]

Moreover, the series of inequalities hold:

\[
\|F_n(X) - A^{-1}\| \leq \|A^{-1}\| \|AX - I\|^{2n-1},
\]

\[
\|C_{k+1}^\text{est} - \hat{C}_{k+1}^\text{est} (n)\|_F \leq \|A\| \|A^{-1}\| \|H_{k+1}^{TL} C_{k+1}^p\|_F \|AX - I\|^2 - \|AX - I\|^2^{n-1},
\]

\[
\|C_{k+1}^\text{est} - \hat{C}_{k+1}^\text{est} (n)\| \leq \|A\| \|A^{-1}\| \|H_{k+1}^{TL} C_{k+1}^p\|^2 \|AX - I\|^{2n-1}.
\]

Proof. Before delving into the details of the proof we note that $\hat{C}_{k+1}^\text{est} (n)$ defined in (13) includes the approximation from algorithm II and the stabilized correction (5) as special cases when $n = 1$ and $n = 2$ respectively.

We start by proving that matrix $\hat{C}_{k+1}^\text{est} (n)$ is nonnegative definite. Foremost, we note that as far as $X$ is symmetric, the matrix $F_n(X)$ is also symmetric for all $n \geq 1$. Then, accounting for the claim of lemma 2.1 we conclude that $\hat{C}_{k+1}^\text{est} (n)$ is nonnegative definite when $n \geq 2$.

To prove (14) we firstly derive the following auxiliary equality:

\[
F_n(X) = A^{-1} - \Delta (A\Delta)^{2n-1-1},
\]

where $\Delta = A^{-1} - X$ and $n \geq 1$. We proceed by induction: if $n = 1$, then $F_n = X$ and formula (17) holds. Next, we suppose that (17) is true for $n = n_0$ and prove that it remains true when $n = n_0 + 1$:

\[
F_{n_0 + 1}(X) = 2F_{n_0}(X) - F_{n_0}(X) AF_{n_0}(X) =
\]

\[
= 2A^{-1} - 2\Delta (A\Delta)^{2n_0-1-1} -
\]

\[
- \left( A^{-1} - \Delta (A\Delta)^{2n_0-1-1} \right) A \left( A^{-1} - \Delta (A\Delta)^{2n_0-1-1} \right) =
\]

\[
= A^{-1} - \Delta (A\Delta)^{2n_0-1-1} A\Delta (A\Delta)^{2n_0-1-1} = A^{-1} - \Delta (A\Delta)^{2n_0-1}.
\]

Thereby, we proved that (17) holds for any $n \geq 1$. We can now finalize the proof of (14):

\[
\|F_n(X) - A^{-1}\| = \|\Delta (A\Delta)^{2n-1-1}\| = \|A^{-1} - X\| (I - AX)^{2n-1-1} =
\]

\[
\|A^{-1} (I - AX)^{2n-1-1}\| \leq \|A^{-1}\| \|AX - I\|^{2n-1}.
\]

Let us assume that $\|AX - I\| < 1$. This implies $F_n(X) \rightarrow A^{-1}$, where the rate of convergence is estimated by an infinitesimal $O(\|AX - I\|^{2n})$.

Taking into account (14) the validity of inequalities (15) and (16) can be shown by following the sketch of the proof of lemma 2.2. \hfill \Box

In essence, sequence (12) corresponds to a matrix inversion formula of Newton-Schultz type. As follows from Lemma 2.4, $F_n$ converges to $A^{-1}$ when initial approximation $B^*$ complies with requirement $\|AB^* - I\| < 1$. This convergence result
is not new, however the inequalities provided by lemma 2.4 are of interest as they make use of (12) in the framework of the Kalman filter, which was not analysed before. Under certain requirements assumed for \( B^* \), this result holds even for non-symmetric matrices, and moreover, it can be proved that when assumption of the matrix \( A \) being non-singular is relaxed, the sequence \( F_n \) converges to Moore pseudoinverse \( A^+ \) as shown in [6]. In addition, properties of Chebyshev polynomials allow to generalize quadratic sequence \( F_n \) by cubic recurrence formula, which provides accelerated convergence rate (see [27] for details). For the case of symmetric \( A \) it was proved in [27] that the initializer \( B^* = (1/\|A\|) I \) guarantees that \( \|AB^* - I\| \leq 1 - 1/(m^{1/2}k) \), where \( m \) is dimension of the matrix and \( k \) is its condition number. Therefore, such choice may not provide satisfactory speed of convergence when \( m \) is large and/or matrix \( A \) is bad-conditioned. Hence, generalized stabilizing correction (12) would not improve filtering properties until provided with a good initial guess \( B^* \). Deriving such initial guess or using Newton-Schultz series in approximative Kalman filter formulations other than discussed in this paper leaves room for further research. However, it should be mentioned that in our numerical experiments initializers computed by BFGS update formula failed to satisfy condition \( \|AB^* - I\| < 1 \) when dimension of observation space increased over 100.

3. Two-layer Quasi-Geostrophic model. This section describes the two-layer Quasi-Geostrophic model [28]. The model is handy to use for data assimilation testing purposes as it is chaotic and its dimension can be adjusted by increasing the density of spatial discretization. At the same time the QG-model is relatively cheap to run and requires no special hardware to perform the simulations. However, the model is sufficiently realistic and leaves room for formulation of challenging inverse problems, so it can be used for testing purposes in performance analysis of data assimilation algorithms [21], [33].

In this section we discuss the governing equations of the QG-model and very briefly highlight the main stages of the numerical solver that we use to integrate the model. In our study we implemented and used the tangent-linear and adjoint codes of this solver. However, we skip the details here as the derivation of the TL- and AD- codes is rather straightforward but contains exhaustive amount of technical details.

The two-layer Quasi-Geostrophic model considered in this paper simulates the quasi-geostrophic (slow) wind motion. The model is defined on a cylindrical surface vertically divided into two interacting layers, where the bottom layer is influenced by the earth topography. The model geometry implies periodic boundary conditions along the latitude. The boundary conditions on the northern and southern walls (the cap and the bottom of the cylinder) are user-defined constant values. Aside from the boundary conditions at the “poles” and the earth topography the model parameters include the depths of the top and the bottom undisturbed atmospheric layers, which we hereafter denote by \( D_1 \) and \( D_2 \) respectively. In addition, the QG-model accounts for the earth rotation via the value of Coriolis parameter \( f_0 \). A possible geometrical topology of the QG-model is presented in figure 1. In the figure the terms \( U_1 \) and \( U_2 \) denote mean zonal flows in the top and the bottom layer respectively.

The QG-model acts on the components of potential vorticity and stream function related to each other in accordance with the equations:

\[
q_1 = \nabla^2 \psi_1 - F_1 (\psi_1 - \psi_2) + \beta y, \tag{18}
\]
Figure I. Geometric topology of the two-layer quasi-geostrophic model

(19) \[ q_2 = \nabla^2 \psi_2 - F_2 (\psi_2 - \psi_1) + \beta y + R_s. \]

Here \( q_i \) and \( \psi_i \) denote potential vorticity and stream function at the top layer for \( i = 1 \) and at the bottom layer for \( i = 2 \). The stream function \( \psi_i \) operated by the model is analogous to pressure. More precisely, \( \nabla \psi_i = (v_i, -u_i) \), where \( u_i \) and \( v_i \) are zonal and meridional wind speed components defined at the corresponding layer. In addition, it is assumed that potential vorticity respects substantial conservation law, which reads as follows:

(20) \[ \frac{D_1 q_1}{Dt} = 0, \quad \frac{D_2 q_2}{Dt} = 0. \]

The terms \( \frac{D_i}{Dt} \) denote substantial derivatives defined for the speed field \( (u_i, v_i) \), i.e. \( \frac{D_i}{Dt} = \frac{\partial}{\partial t} + u_i \frac{\partial}{\partial x} + v_i \frac{\partial}{\partial y} \).

Equations (18), (19) and (20) fully qualify the two-layer QG-model. However, these PDEs are nondimensional. Nondimensionalization procedure for the QG-model is defined by the following relations:

(21) \[ F_1 = \frac{f_0^2 L^2}{g D_1}, \quad F_2 = \frac{f_0^2 L^2}{g D_2}, \]

(22) \[ \dot{\hat{g}} = \frac{\Delta \theta}{\hat{\theta}}; \]

(23) \[ R_s = \frac{S(x, y)}{\eta D_2}; \]

(24) \[ \beta = \beta_0 \frac{L}{U}, \]

where \( L \) and \( U \) are the length and speed main scales, \( \beta_0 \) is the northward gradient of the Coriolis parameter, \( \hat{g} \) is the gravity acceleration, \( \Delta \theta \) is temperature change across the layer interface, \( \hat{\theta} \) is mean temperature, \( S(x, y) \) is a surface specifying the earth topography, \( \eta = \frac{U}{f_0 L} \) is the Rossby number of the given system.
In the rest of this section we sketch out the integration method for the PDE system (18), (19), (20). The key idea is to assume that potential vorticity at the current time instant denoted as $q_i(t_0)$ is already known. Then in order to obtain the corresponding stream function we have to invert equations (18) and (19). By applying $\nabla^2$ to (18) and subtracting the sum of $F_1$ times (19) and $F_2$ times (18) from the result we arrive at:

$$\nabla^2 [\nabla^2 \psi_1] - (F_1 + F_2) [\nabla^2 \psi_1] =$$

$$= \nabla^2 q_1(t_0) - F_2 (q_1(t_0) - \beta y) - F_1 (q_2(t_0) - \beta y - R_s).$$

Equation (25) is nonhomogeneous Helmholtz equation with negative constant parameter $-(F_1 + F_2)$, where $\nabla^2 \psi_1$ is unknown [29]. The boundary conditions for this auxiliary problem are inherited from the boundary conditions of the QG-model. A general method for solving (25) is discussed in [11]. Alternatively, it is possible to apply finite-difference scheme, which turns out to be stable when the parameter of Helmholtz equation is a negative constant. In our implementation we use the second approach.

Assuming that (25) is solved, we can compute the value for $\psi_1$ by solving the corresponding Poisson problem [17]. The solution is obtained by finite-difference method [24]. The value for $\psi_2$ can be deduced from (18) and (19) by substituting the obtained value for $\psi_1$. Therefore, given potential vorticity for the current time instant we can compute the corresponding stream function in both layers. Finally, in order to account for the conservation law (20) we define propagation of potential vorticity over the time by a semi-Lagrangian method [31].

Thereby, the discrete numerical solver for the PDE system (18), (19) and (20) can be composed of three stages, where the first one employs semi-Lagrangian advection to propagate the system in time, and the rest include double application of finite-differences to define the result of propagation over the spatial discretization grid.

4. Numerical experiments. In this section we present data assimilation benchmarks made to assess competitive performance of SA-EKF algorithm introduced in the previous sections.

The numerical experiments were carried out using the two-layer Quasi-Geostrophic model with several benchmarks. In our benchmarks we consider all values simulated by the QG-model on its two-level spatial grid stacked into a long vector, which we hereinafter refer as the state vector of the model. The natural vector space corresponding to the state vector is referred as the state space of the model.

Our benchmarks were run with increasing dimension of the state space. The intention was to assess performance of L-BFGS-EKF and of its stabilized version SA-EKF with respect to dimension of the problem. In addition, each benchmark itself included two parallel executions of the QG-model made at different resolutions. The simulation executed with the higher resolution, which hereafter is referred to as the truth run, was used to generate observations of the “nature”. The lower-resolution simulation, referred to as the biased run, was employed as the transition operator $M_k$ in the Kalman filters.

In order to introduce an additional bias the truth and the biased runs were executed with different values for the layer depths $D_1$ and $D_2$. In addition, the state of the truth run was not fully observed having only a few of its components continuously monitored. The locations of the observations were selected randomly.
Due to the difference in resolutions between the truth and the biased runs, the spatial locations of the observation sources and the nodes of the discrete grid of the values simulated by the biased run did not necessarily line up. This was circumvented by interpolating the state space of the biased run onto the state space of the truth run using bilinear interpolation. Hence, the observation operator $\mathcal{H}_k$ was composed of two steps with the first stage implementing interpolation of the state space and the second stage performing projection of the interpolated space onto the subspace of the observations. Since the observed locations were selected once and remained unchanged during each assimilation run, the operator $\mathcal{H}_k$ was independent of $k$. Consequently, in the experiments $\mathcal{H}_k$ was linear and its own implementation was thus used as the TL-code. The corresponding AD-code was programmed by applying the chain rule to the AD-codes of the bilinear interpolator and of the orthogonal projector.

The TL and AD codes of the evolution were derived by a careful examination of the numerical solver implementing transition operator $\mathcal{M}_k$. The time step for the solver was set to 1 hour (this is the discretization time step used in integration of the PDEs (18), (19), (20)). The step of data assimilation was 6 hours. This means that a single state transition by $\mathcal{M}_k$ comprised 6 repetitive evaluations of the underlying numerical solver. Therefore, the corresponding TL- and AD-codes utilized information from a nonlinear model trajectory tracing it forwards and backwards respectively and applying the chain rule. It should be noted that the prediction operator $\mathcal{M}_k$ was fixed, i.e. independent of $k$ in the simulations. The further details regarding configuration of the data assimilation and settings used by the model are described in the following subsections.

4.1. The covariance matrices. The prediction and observation uncertainties were modelled by normally distributed random variables $\epsilon_k$, $\eta_{k+1}$ (see equations (1) and (2)) with covariance matrices defined as follows:

$$C_{\epsilon_k} = \begin{bmatrix} \sigma^2 I & \alpha \sigma^2 I \\ \alpha \sigma^2 I & \sigma^2 I \end{bmatrix}, \quad C_{\eta_{k+1}} = \xi^2 I,$$

where $\sigma^2$ is the variance of model uncertainty, $\alpha$ is correlation coefficient between the layers, $\xi^2$ is the variance of observation measurement error, and $I$ is identity matrix of dimension dictated by prediction and observation models. We assigned the following values for these parameters: $\sigma = 0.1$, $\alpha = 0.8$, $\xi = 0.5$. The values used for $\alpha$ and $\sigma$ were estimated from model simulations. The value of $\xi^2$ was selected in accordance with the perturbation noise applied to the observation data.

4.2. The settings of the QG-model. The configuration of the QG-model used in our benchmarks was defined in accordance with the experiments performed by ECMWF [21]. The settings described here are aimed to create baroclinic instability in the model dynamics, which is a quasi-periodic regime qualified by seemingly “stormy” behaviour. This model regime can be used to efficiently analyse performance of data assimilation methods as its trajectories reside within a bounded region but do not converge towards equilibria. The model was simulated in a 12000km-by-6000km rectangular domain. The bottom surface orography was modelled by a Gaussian hill of 2000m height and 1000km wide located at coordinates (1200km, 9000km) counting from the lower-left corner of the simulation domain. The nondimensionalization of the model was defined by the main length scale $L = 1000km$ and the main velocity scale $U = 10m/sec$. The rest
of the model parameters introduced in (21) were assigned to the following values: 
\[ f_0 = 10^{-4}, \Delta \theta = 0.1, \beta_0 = 1.5 \cdot 10^{-11}. \]
The values for model layer depths \( D_1 \) and \( D_2 \) were different for the truth and the biased runs to introduce an extra bias into prediction. These settings are described in detail in the next subsection.

4.3. The assimilation runs. As pointed earlier, we run two instances of the QG-model in parallel, where the first instance has the higher resolution and simulates “the nature”, and the second instance with lower resolution models prediction operator employed by the filters. The layer depths in the truth run were defined by \( D_1 = 6000m \) for the top layer and \( D_2 = 4000m \) for the bottom layer. The biased runs were executed using the layer depths of 5500m and 4500m, respectively. Prior to start of the data assimilation the truth and the biased runs were initialized with the same initial state (interpolated for the corresponding resolution of the grid) and were both executed for 10 days of the model time. As illustrated by figure III this caused notable divergence between the model states. The data assimilation began after the end of this 10 day spin-up period. The state of the biased run at the end of this period was used as the first initial estimate \( x^{est}_0 \). The uncertainty of this estimate was modeled “naively” by identity covariance, i.e. \( C^{est}_0 = I \).

The goal of assessment was to test the ability of the assimilation methods to recover from the error in the initial estimate, account for the bias in transition operator, and accurately predict the state at the next time instant. As criteria for quality of assimilation we used two concepts: the root mean square error of the estimates, and the forecast skill (precise definitions are given below). Each data assimilation experiment comprised 500 assimilation rounds, which is equivalent to 125 days of the model time. The forecast skills were calculated skipping the time period where assimilation did not recover from the initial gross error, so that first steps of assimilation were not counted in the skill. For all filters a total number of 150 assimilation rounds was assumed to be a “satisfactory” recovery time. Figure II contains example root mean square errors of the estimates computed by the filters during the first 30 assimilation rounds. These curves demonstrate elimination of the error present in the initial estimate. Hereinafter, this process will be sometimes referred to as filter convergence.

In our experiments the model state vector was defined by discretized stream functions \( \psi_1 \) and \( \psi_2 \) (see equations (18), (19)). The Kalman filters were employed to correct predictions made by the biased run using the observation data obtained from the truth run. The quality of resulting SA-EKF estimates was assessed by root mean square error (RMS error) defined as follows:

\[
\mathcal{R} \left( x^{est} \right) = \frac{\| x^{truth} - x^{est} \|}{\sqrt{N}}.
\]

Here \( x^{est} \) is the estimate of the true state \( x^{truth} \) at a given time instant, \( N \) is dimension of these vectors, and \( \| \cdot \| \) is Euclidean norm. Because in our benchmarks the truth and the biased runs are executed at different resolutions, the vectors \( x^{truth} \) and \( x^{est} \) have different dimensions. Hence, in order to give meaning to the numerator in formula (26) the estimate \( x^{est} \) is interpolated onto the spatial grid of the corresponding truth run.

An additional study of performance was done by the forecast skills. At each time instant, the estimates were propagated by applying transition operator \( \mathcal{M}_k \). The resulting forecast was compared with the corresponding “real” state of the truth
run using the RMS error metrics. Hence, the resulting average over all assimilation rounds describes accuracy of the forecast for a given prediction range. The forecast skills were compared against standard baseline called “tourist brochure” (climatology), which is defined as the RMS error of the mean model state.

In all discussed benchmarks, the biased run had resolution that allowed explicit application of the exact EKF formulas. Therefore, the EKF estimates were used as a reference line for performance assessment of the approximative L-BFGS-EKF and SA-EKF. Our numerical experiment included three benchmarks with increasing resolution of the biased run (and therefore, even higher resolution of the corresponding truth run). Hereinafter in the text we will often refer to these benchmarks by numbers (i.e. benchmark 1, benchmark 2, and benchmark 3 in the order of resolution increase).

Table I describes settings used in the benchmarks. Each row of this table contains settings of the benchmark in the first column. The second and the third columns define resolutions of the truth and the biased runs respectively. The values in the fourth and the fifth columns are the number of observation sources and dimension of the state vector.

Table II summarizes results obtained from the benchmarks. Numerical values in the table are the mean values of the RMS error of the filter estimates obtained given different maximal number of stored BFGS iterations. The cells without numbers

**Figure II.** Filter recovery from the errors in the first initial estimate.

**Table I.** Benchmark options depending on model resolution.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>10-by-10</td>
<td>9-by-9</td>
<td>50</td>
<td>162</td>
</tr>
<tr>
<td>II</td>
<td>15-by-15</td>
<td>12-by-12</td>
<td>85</td>
<td>288</td>
</tr>
<tr>
<td>III</td>
<td>80-by-40</td>
<td>40-by-20</td>
<td>500</td>
<td>1600</td>
</tr>
</tbody>
</table>
Figure III. Truth and biased run states after 10 days of model time simulation. X-, and Y-axes correspond to longitude and latitude respectively. The isolines describe values of the stream function $\psi_i$ with color mapping from blue to red.

Table II. Mean values of RMS error obtained from the benchmarks using varied capacity of the BFGS storage.

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>I</th>
<th>II</th>
<th>III</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BFGS-EKF</td>
<td>SA-EKF</td>
<td>BFGS-EKF</td>
</tr>
<tr>
<td>BFGS vectors</td>
<td>5</td>
<td>0.5860</td>
<td>0.5508</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>0.5573</td>
<td>0.5445</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>0.4959</td>
<td>0.4788</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>0.4686</td>
<td>0.4695</td>
</tr>
<tr>
<td>EKF ref.</td>
<td>0.4247</td>
<td>0.4346</td>
<td>0.2094</td>
</tr>
</tbody>
</table>
Table III. Expected forecast length with respect to varied capacity of the BFGS storage.

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>I</th>
<th>II</th>
<th>III</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BFGS-EKF</td>
<td>SA-EKF</td>
<td>BFGS-EKF</td>
</tr>
<tr>
<td>BFGS vectors</td>
<td>5</td>
<td>78</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>84</td>
<td>84</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>84</td>
<td>84</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>84</td>
<td>84</td>
</tr>
<tr>
<td>EKF ref.</td>
<td>90</td>
<td>66</td>
<td>48</td>
</tr>
</tbody>
</table>

marked by “–” covariance inflation with suggested inflation factor did not prevent the divergence of the filter.

Table III contains results obtained from the same experiments in form of the forecast length. Here the forecast length is the time period between beginning of the forecast and intersection of the forecast skill curve with “tourist brochure”. Time periods in Table III are rounded to hours of model time.

5. Conclusion. In this paper we propose a stabilizing correction for L-BFGS approximation of the extended Kalman filter previously introduced in the literature. We call our reformulation of the algorithm SA-EKF to emphasize the difference. The suggested correction ensures that the covariance matrix of a filter estimate remains “proper” in the sense that all of its eigenvalues are non-negative.

Earlier results in [2] demonstrated that even the non-stabilized L-BFGS approximation converges for ‘easy’ non-chaotic large scale models, such as the heat equation, or smaller size chaotic systems, such as Lorenz95 with 40 state dimensions. However, when the state space dimension increases in a chaotic system, the L-BFGS EKF tends to require larger BFGS storage in order to converge, or may fail due to accumulating errors. It was shown that SA-EKF performs in a robust way, and is able to achieve satisfactory approximation results with considerably smaller number of BFGS iterations.

6. Appendix. The appendix section contains details about the BFGS method. Here we prove the BFGS update formula and discuss its use in Quasi-Newton unconstraint optimization algorithms.

The L-BFGS unconstraint optimizer is a Quasi-Newton method for quadratic minimization (nevertheless, it can be extended to non-quadratic cost functions). Therefore, it is based on approximation of inverse Hessian of the cost function. The approximation addressed by the L-BFGS method is based on BFGS update, a rank-two update formula for matrix approximation. Below we present a sketch of derivation of the BFGS formula based on good Broyden 1-rank update.

6.1. Step 1. Good Broyden update. We begin by considering the following task: given a pair of vectors \( s, y \), where \( s \) is a non-zero, and a square matrix \( A \) find matrix \( D \) such that \( (A + D)s = y \) and \( D \) has a minimal Frobenius matrix norm.

In other words, we need to find a solution for the optimization task that obeys the following:

\[
(27) \quad \text{argmin}_D \|D\|_{F} : \ (A + D)s = y.
\]
Lemma 6.1. The task (27) is solved by $D = y - As, s^T$.

Proof. The solution for the task (27) can be constructed explicitly. Consider an arbitrary matrix $D$. The Frobenius norm of this matrix can be defined as follows

$$
\|D\|_{Fr} = \text{tr} (D^T D),
$$

where $\text{tr}(\cdot)$ denotes the trace defined as the sum of matrix diagonal elements. Hence, in accordance with the matrix spectral theorem $\|D\|_{Fr} = \text{tr} (PLP^T)$, where $P$ is the matrix composed of the column eigenvectors of $D^T D$ and $L$ is diagonal matrix of the corresponding eigenvalues. Since trace is invariant to the choice of basis, we can re-express Frobenius norm of the matrix $D$:

$$
\|D\|_{Fr} = \sum_i \lambda_i,
$$

where $\lambda_i \geq 0$ is the $i$-th eigenvalue of $D^T D$. At the same time, we require $Ds = y - As = r$. Hence, $s^T D^T Ds = r^T r$. Applying again the spectral theorem, we obtain $\tilde{s}^T L \tilde{s} = r^T r$, where $\tilde{s} = P^T s$ is representation of vector $s$ in orthonormal basis of the eigenvectors in $P$. Therefore, the task (27) can be reformulated.

(28)

$$
\|D\|_{Fr} = \sum_i \lambda_i \rightarrow \min,
$$

(29)

$$
\sum_i \lambda_i \tilde{s}_i^2 = r^T r.
$$

Next, we assume that $\tilde{s}_1^2 \geq \tilde{s}_i^2, \forall i$. We thus suppose that $\tilde{s}_1^2$ is the biggest among the terms $\tilde{s}_i$ (it is a legal assumption since $\tilde{s}_1^2$ can always be made the biggest by changing the numeration of the eigenvalues $\lambda_i$). Hence, from the constraint equation (29) we obtain

$$
\lambda_1 = \frac{r^T r - \sum_{i \neq 1} \lambda_i \tilde{s}_i^2}{\tilde{s}_1^2}.
$$

Substituting the expression for $\lambda_1$ into minimization problem (28) we eliminate the constraint (29).

(30)

$$
\frac{r^T r}{\tilde{s}_1^2} + \sum_{i \neq 1} \lambda_i \left(1 - \frac{\tilde{s}_i^2}{\tilde{s}_1^2}\right) \rightarrow \min.
$$

Obviously, the sum in (30) is non-negative and as far as we want to selected $D$ so that is minimizes the task (27), we set $\lambda_i = 0$ when $i \neq 1$. Hence $\lambda_1 = \frac{r^T r}{\tilde{s}_1^2}$. However, $\tilde{s}_1^2$ is determined by magnitude of orthogonal projection of vector $s$ onto the first eigenvector of matrix $D^T D$. Therefore, $\lambda_1$ is minimized by assigning the first eigenvector of $D^T D$ to $\frac{s}{\|s\|}$. Then $\tilde{s}_1^2 = \|s\|^2$ and therefore $D^T D = s \frac{s^T s}{\|s\|^2} s^T$, which leads to the obvious solution $D = y - As, s^T$. This 1-rank update is known as good Broyden update.

6.2. Step 2. BFGS update formula. Our next goal is to strengthen the constraints of the problem (27) by requiring the updated matrix $A + D$ to be symmetric and positive definite assuming that the matrix $A$ meets this requirement. However, there may be no solution satisfying the constraints. The following lemma gives a simple existence criterion.
Lemma 6.2. A symmetric positive definite matrix \( A + D \) satisfying the constraint \((A + D)s = y\) exists if and only if there is a non-zero vector \( z \) and non-singular matrix \( L \) such that \( y = Lz \) and \( z = L^T s \).

Proof. Indeed, assume that \( A + D \) is a symmetric positive definite update, which satisfies the constraint \((A + D)s = y\). Then we can apply Cholesky decomposition which yields \((A + D) = \Sigma \Sigma^T\). Recalling the constraint, we obtain \(\Sigma \Sigma^T s = y\). Hence, setting \( z = \Sigma^T s \) and \( L = \Sigma \) finalizes the first part of the proof.

Suppose now, that there is a non-zero vector \( z \) and a non-singular matrix \( L \) such that the lemma requirements are met. Then we can set \( A + D = LL^T \). Indeed, \((A + D)s = LL^T s = Lz = y\), therefore the desired constraint is satisfied.

Next we derive the BFGS formula. In this paper we only provide a sketch of the proof, which was together with lemma 6.2 suggested in [16]. We need to emphasize that the derivation of the BFGS update formula is done for the case when the matrix being updated by the formula is positive-definite. However, earlier in this paper we employed a more weak assumption of non-negative definiteness. This weakening is possible due to the fact that BFGS and L-BFGS optimization algorithms can be implemented even when Hessian matrix of the cost function is singular [26], although the derivation of the BFGS update formula does not hold in this case.

Lemma 6.3. Assume that \( B = JJ^T \) is positive definite. Consider a non-zero vector \( s \) and a vector \( y \). Hereby, there is a symmetric positive-definite matrix \( B = \tilde{J} \tilde{J}^T \) such that \( Bs = y \) if and only if \( y^Ts > 0 \). If the latter requirement is satisfied, then

\[
\tilde{J} = J + \frac{y - y^Ts \tilde{J}s}{z^T z} \tilde{J}s^T s
\]

Proof. We start by considering a non-zero vector \( s \) and a positive definite matrix \( \tilde{B} \) such that \( \tilde{B}s = y \). Then lemma 6.2 implies that there is a non-zero vector \( z \) and a non-singular matrix \( L \) such that \( y = Lz \) and \( z = L^Ts \). Hence, \( z^T z = (s^T L) (L^{-1} y) = s^T y > 0 \).

Hereafter we assume \( s^T y > 0 \). Our goal is to construct \( \hat{J} \) such that \( \hat{B} = \hat{J} \hat{J}^T s = y \). Let us denote \( z = \hat{J}^Ts \) and assume that \( z \) is already known. Then we can employ lemma 6.1 to update the given matrix \( J \).

\[
\hat{J} = J + \frac{y - Jz}{z^T z} z^T.
\]

The equality \( z = \hat{J}^Ts \) and expression (31) for \( \hat{J} \) allow us to write an equation with respect to the vector \( z \).

\[
\hat{J}^Ts = J^Ts + \left(\frac{y^Ts - z^T J^T s}{z^T z} s\right) z = z.
\]

This implies that \( z = \alpha J^Ts \). Plugging the expression for \( z \) back into the latter equation, we obtain

\[
\alpha = 1 + \frac{y^Ts - s^T Bs}{\alpha s^T Bs}.
\]

Hence, \( \alpha^2 = (y^Ts)/(s^T Bs) \) and as far as \( y^Ts > 0 \) and \( B \) is positive definite we can get a real value for \( \alpha \) (it is not important whether the chosen value for \( \alpha \) is positive or negative). Therefore \( z = \sqrt{(y^Ts)/(s^T Bs)} J^Ts \). Substitution of the expression for \( z \) into (31) yields the desired formula for \( \hat{J} \). □
The proven lemma is fundamental for derivation of the classical form of the BFGS update. This can be done by plugging the derived formula for \( \tilde{J} \) into \( \tilde{B} = \tilde{J}_t \). Skipping the onerous calculations the substitution yields:

\[
(32) \quad \tilde{B} = B + \frac{yy^T}{y^Ts} - \frac{Bss^TB}{s^TBs}.
\]

In addition, the Sherman-Morrison-Woodbury matrix inversion lemma applied to (32) gives the BFGS update formula for inverse matrix:

\[
(33) \quad \tilde{B}^{-1} = \frac{ss^T}{y^Ts} + \left( I - \frac{sy^T}{y^Ts} \right) B^{-1} \left( I - \frac{ys^T}{y^Ts} \right).
\]

It is possible to prove that (33) defines a least change update for the positive definite \( B^{-1} \) given \( \tilde{B}^{-1}y = s \) [14], [15]. More precisely, (33) provides solution for the following optimization problem:

\[
\begin{align*}
\arg\min_{\tilde{B}: \tilde{B} > 0, \tilde{B}^{-1}y = s} \| B^{-1} - \tilde{B}^{-1} \|_F, \\
\end{align*}
\]

where \( B \) is assumed to be positive-definite.

When applied to approximation of the Newton method the update formula (33) leads to the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm. For the purposes of the paper, the BFGS method is limited to quadratic minimization. However, the results given here can be extended to non-quadratic case [26].

Below we present a sketch of the BFGS algorithm. We adhere to the following notation: \( A \) and \( b \) are as defined in optimization problem (4), \( x_0 \) is an \( n \)-by-1 vector providing initial approximation for the cost function minimizer, \( h_0 \) is a scale factor of inverse Hessian matrix approximation initially assigned to \( h_0I \), \( N \) is the requested number of iterations, and \( x_{min} \) is the obtained minimizer of the cost function \( f(x) \).

**Algorithm III. BFGS unconstraint optimizer.**

**INPUT:** \( f(x) = \frac{1}{2}(Ax, x) - (b, x) \), \( x_0 \), \( h_0 \), \( N \).

**OUTPUT:** \( x_{min} \).

1. Assign iteration number \( i = 0 \). Initialize the initial inverse hessian approximation \( H_0 = h_0I \).
2. If \( i \geq N \) assign \( x_{min} = x_i \) and exit, proceed to the next step otherwise.
3. Compute gradient \( g_i = \nabla f(x_i) = Ax_i - b \) and set the decent direction \( d_i = -H_i g_i \).
4. Compute the step length \( \alpha_i = -\frac{d_i^T g_i}{d_i^T A d_i} \).
5. Define the next approximative minimizing point \( x_{i+1} = x_i + \alpha_i d_i \) and the corresponding gradient \( g_{i+1} = \nabla f(x_{i+1}) = Ax_{i+1} - b \).
6. Specify the mapped pair \( s_i = x_{i+1} - x_i \), \( y_i = g_{i+1} - g_i \). Obviously, \( A s_i = y_i \).
   
   Update inverse hessian \( H_i \) using (33) and the mapped pair \((s_i, y_i)\). Assign the updated approximation to \( H_{i+1} \), update iteration number \( i = i + 1 \) and repeat all over starting from step 2.

Assume that the given algorithm has generated the mapped pairs \( (s_i, y_i) \), where \( i = 0, \ldots, N - 1 \). Making use of these data and the BFGS inverse update (33) it is possible to approximate the matrix-vector product \( A^{-1}x \) for an arbitrary \( x \).

By limiting the maximal number of the stored mapped pairs to a given threshold \( M \leq N \), the algorithm III can be reformulated in a reduced memory fashion called L-BFGS. The mapped pairs generated by the L-BFGS thus provide a low-memory
representation of the operator $A^{-1}$. Leveraging formula (32) the same mapped pairs can be employed to define a low-memory representation for the forward matrix $A$.

Wrapping up this section we should note, that the sketch given by algorithm III can not be considered the way to follow in order to get an efficient implementation of BFGS or L-BFGS. In addition, we do not discuss the question of choosing the initial Hessian scale $h_0$ and the cases of singular Hessian matrix. A more detailed analysis of these algorithms is available in [25].

7. **Acknowledgements.** The contribution discussed in the present paper has been financially supported by the Finnish Academy Project 134937 and the Centre of Excellence in Inverse Problems Research (project number 250 215).

**REFERENCES**


[33] Y. Tremolet and A. Hofstadler, OOPS as a common framework for Research and Operations, Presentation 14th Workshop on meteorological operational systems, ECMWF, 2013.


Received August 2014; revised May 2015.

*E-mail address*: aleksandr.bibov@lut.fi

*E-mail address*: heikki.haario@lut.fi

*E-mail address*: antti.solonen@gmail.com