LUT
Lappeenranta
University of Technology

Alexander Bibov

LOW-MEMORY FILTERING FOR LARGE-SCALE DATA ASSIMILATION

Thesis for the degree of Doctor of Science (Technology) to be presented with due permission for public examination and criticism in the Auditorium 2310 at Lappeenranta University of Technology, Lappeenranta, Finland on the 22nd of May, 2017, at noon.

Acta Universitatis
Lappeenrantaensis 744

Supervisor Professor Heikki Haario
LUT School of Engineering Science
Lappeenranta University of Technology
Finland

Reviewers PhD Kody Law
Oak Ridge National Laboratory
United States of America

Professor Youssef Marzouk
Department of Aeronautics and Astronautics
Massachusetts Institute of Technology
United States of America

Opponent Lassi Roininen
Imperial College London
Department of Mathematics
United Kingdom

ISBN 978-952-335-076-2
ISBN 978-952-335-077-9 (PDF)
ISSN-L 1456-4491
ISSN 1456-4491
Lappeenrannan teknillinen yliopisto
Yliopistopaino 2017

Abstract

Alexander Bibov
Low-Memory Filtering For Large-Scale Data Assimilation
Lappeenranta 2017
82 pages
Acta Universitatis Lappeenrantaensis 744
Diss. Lappeenranta University of Technology
ISBN 978-952-335-076-2, ISBN 978-952-335-077-9 (PDF), ISSN-L 1456-4491, ISSN 1456-4491

Data assimilation is process of combining information acquired from mathematical model with observed data in attempt to increase the accuracy of both. The real world phenomena are hard to model in exact way. In addition, even when certain processes allow very accurate mathematical description the model often cannot be represented in a closed form. These facts lead to necessity to deal with modelling errors when it comes to numerical simulations of real-life phenomena. One of the usual ways to improve the quality of simulated data is to use information from (possibly indirect) observations. The observations in turn are prone to the measurement errors that should also be taken into consideration. Therefore, the commonly assumed task, which is the subject for data assimilation could be roughly formulated as follows: given prediction computed by certain numerical simulation and the corresponding (possibly indirect) observation provide an optimal estimate for the state of the system in question. Here the optimality can have different meanings, but the commonly assumed case is that the estimate must be unbiased to gain correct grasp of reality and that it should have the minimal variance, which corresponds to noise reduction. The algorithms that solve the data assimilation task are called data assimilation methods. From the aforementioned descriptions it is visible that data assimilation is in the essence similar to fitting model to the data. The special term of "data assimilation" was borrowed from meteorological community and is mostly used when the system leveraged to compute predictions is having high dimension and is chaotic, i.e. sensitive to the initial state. This dissertation mainly focuses on such models, which is the reason to use this special term here.

In this dissertation we consider data assimilation methods that deal with the case where dimension of the state space of the system being simulated is too "large-scale" for the classical algorithms to be practicable. By "large-scale" here we presume that if n is dimension of the state space, then n is considered "large" if an n-by-n matrix could not fit into available computer memory using desired storage format (e.g. single- or double-precision). Common example of a large-scale model is an operational weather prediction simulation system. By the moment of writing this text for such systems $n \approx 10^9$, which means that a 10^9-by-10^9 matrix stored in double-precision format (this is often required in scientific simulations) would occupy approximately 7275958TB of memory, which is far beyond the capabilities of all modern supercomputers (when this text was written the fastest supercomputer was Sunway TaihuLight located in national supercomputer center in Wuxi, China and it was having only 1310TB or RAM).

The motivation for having special emphasis for the large-scale models is that the classical optimal data assimilation approaches employ covariance matrices of the state vectors (the vectors containing all parameters that fully represent a state of the model at given time instance). This implies necessity to store n-by-n matrices, which makes implementation of all such methods inefficient. In this dissertation we attempt to address this problem by considering low-memory approximations of the classical approaches. The approximations are by nature sub-optimal, but the way they are formulated allows to alleviate the memory issues that arise in the classical algorithms.

In the present work we concentrate on low-memory approximations of the Extended Kalman filter based on limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) unconstrained optimization scheme and present family of stabilizing corrections that allow to circumvent certain stability issues that are present in some previously known approaches based on this scheme. We also demonstrate that our stabilizing corrections imply better convergence properties and re-use this fact to formulate and solve the parallel filtering task, which is essentially a low-memory approximation of the fixed-lag Kalman smoother.

We study performance of our methods using a synthetic model, the two-layer Quasi-Geostrophic model, which describes conservative wind motion over cylindrical surface vertically divided into two layers. The model is a well-known test case and has been extensively used in ongoing research conducted in e.g., European Centre For Medium-Ranged Weather Forecasts, Reading, UK. We analyse performance of our methods by comparing them against a set of competing low-memory data assimilation techniques such as Variational Kalman Filter, BFGS Low-Memory Kalman Filter, Weak-Constraint 4D-VAR, and a selection of ensemble-based algorithms.

Finally, we analyse applicability of our approaches by considering the problem of estimating intensity of blooming in the coastal regions of the Finnish gulf during the Spring-Summer months. For this case we use high-resolution satellite images that provide concentrations of the chlorophyll in the gulf. However, the data taken at certain time instances is not complete due to cloudiness and therefore, the task of estimating the concentrations of chlorophyll turns out to be a perfect candidate for data assimilation. In addition, the problem is naturally large-scale due to the resolution of the original data.

Keywords: Dynamical systems, Data assimilation, Kalman filter, Low-memory matrix approximation, Weather prediction, Satellite Imaging

Acknowledgements

This work has originated in 2011 at the Department of Computational Engineering and Physics, formerly called the Department of Mathematics and Physics of Lappeenranta University of Technology (LUT). The project has been greatly supported by numerous discussions held in LUT, in the Dynamicum building of Finnish Meteorological Institute (FMI), in the European Centre For Medium-Ranged Weather Forecasts located in Reading, UK, and of course in the snowy huts of Lapland in informal but yet demanding atmosphere of the annual seminars held by my scientific supervisor Heikki Haario in Luosto, Lapland. With this in mind, I would like to particularly thank Professor Haario for his especially dedicated attitude and passion towards scientific success. His contribution to this work cannot be overestimated. I would like to particularly thank Heikki for the freedom he allows for his students in organization of their research approaches: this is one of the key points in creation of brilliant environment, which provides extraordinary opportunity for learning and lays the foundation for preparing outstandingly qualified professionals having the capability to challenge the trickiest problems of today's engineering science.

In addition to Heikki Haario, I would like to thank Marko Laine, whose advice, technical insight and deep expertise in data assimilation and weather prediction provided indispensable endowment into completion of this project. My special gratitude is dedicated to Professor Heikki Järvinen. Every of his precious comments during discussions we held in Lappeenranta and at FMI was a great impact that helped us to avoid many false approaches.

I also have to thank my colleges whose expertise, openness, and readiness for collaboration has made a great deal during the progress of this work. I want to specially emphasize contributions made by Antti Solonen, Vladimir Shemyakin, Idrissa Amour, and Alexey Kozarnikov.

By all means I have to underline the role of my parents: yet we are now mostly living far from each other, without them I would have never stayed a chance to become the person I am and this work would never have been possible.

My special heartfelt gratitude is dedicated to my beloved wife Anna. Now, staying in the end of this very long path I can sincerely say that you were my very grand reason that always kept and continues keeping me going towards my goals, regardless of how difficult and obscure they may look at the beginning. Inarguably, it was your endless indispensable support that helped me to make it so far.

Finally, I want to thank all those numerous people whose comments, arguments, and friendly support has been providing this great working environment one always needs so much. It is hard to state everybody by name, but I want all of you to know: without your assistance this project would not have become a quarter of what it is today.

Alexander Bibov
April 2017
Lappeenranta, Finland

To my beloved wife and
parents. For you are the ones to give me the strength to keep
going. Today and till the last of my days.

Alexander Bibov

Contents

List of publications

Publication I
Bibov, A., Haario, H., and Solonen, A. (2015). Stabilized Approximate Kalman Filter. *Inverse Problems and Imaging*, 9(4), pp. 1003-1024.

The author has proposed the stabilizing correction described in the paper, formulated and proved the related lemmas and conducted the numerical experiments.

Publication II
Bibov, A. and Haario, H. (2016). Parallel implementation of data assimilation. *International Journal for Numerical Methods in Fluids*, pp. 1-17, 2016.

The author has proposed and implemented the algorithms considered in the paper and conducted the numerical experiments

Publication III
Amour, I., Mussa, Z., Bibov, A., and Kauranne, T. (2013) Using ensemble data assimilation to forecast hydrological flumes. *Nonlinear Processes in Geophysics*, 20, pp. 955-964.

The author has provided his implementation of the variational Kalman filter and implemented a wrapper for the shallow water model that allowed to interface the model and the filter. In addition the author provided his implementation of the two-layer quasi-geostrophic model and conducted the related numerical experiments

Publication IV
Solonen, A., Hakkarainen, J., Ilin, A., Abbas, M., and Bibov, A. (2014) Estimating model error covariance matrix parameters in extended Kalman filtering. *Nonlinear Processes in Geophysics*, 21, pp. 919-927.

The author has provided his implementation of the two-layer quasi-geostrophic model and integrated it into the data assimilation toolbox used by the rest of the research team.

Other related publications
Solonen, A., Bibov, A., Bardsley, J.M., and Haario, H. (2014). Optimization-Based Sampling in Ensemble Kalman Filtering. *International Journal for Uncertainty Quantification*, 4(4), pp. 349–364.

The author has provided his implementation of the two-layer quasi-geostrophic model and conducted the related numerical experiments. The author has also provided implementation of the algorithm described in the paper and developed localization and inflation techniques to improve its accuracy.

Nomenclature

$\bar{0}$	all-zero vector of conformal dimension
$\bar{C}_{\epsilon_k}$	combined model error covariance at time instance k
$\bar{C}_{\eta_k}$	combined observation error covariance at time instance k
$\bar{x}$	vector x of conformal dimension
$\bar{X}_k$	combined state vector at time instance k
$\bar{Y}_k$	combined observation vector at time instance k
ϵ_k	prediction error at time instance k
η_k	observation error at time instance k
$\frac{\partial \mathcal{A}(x)}{\partial x}$	Jacobian matrix of operator $\mathcal{A}(x)$ calculated with respect to x
$\mathbb{E}[\cdot]$	the expected value
$\mathbb{E}^n$	n-dimensional Euclidean space
$\mathbb{R}^n$	n-dimensional real-valued vector space
$\mathcal{H}_k$	observation operator describing relation between the state and the observed data at time instance k
$\mathcal{M}^{AD}$	adjoint code of operator $\mathcal{M}$
$\mathcal{M}^{TL}$	tangent-linear code of operator $\mathcal{M}$
$\mathcal{M}_k$	transition operator at time instance k; identifies change between the state at time instance k and time instance $k+1$
$\mathbf{tr}(\cdot)$	trace of a matrix
$\mathrm{Img}(A)$	image of operator A
$\mathrm{Ker}(A)$	kernel of operator A
C_k^p	covariance matrix of the prior state x_k^p
C_k^{est}	covariance matrix of the estimate x_k^{est}
C_{ϵ_k}	covariance matrix of the prediction error ϵ_k
C_{η_k}	covariance matrix of the measurement error η_k
$f(x,\theta_1,\dots\theta_m) \underset{x}{\rightarrow} min$	minimization of function $f(x,\theta_1,\dots,\theta_m)$ with respect to x
G_k	the Kalman gain calculated for time instance k
H^+	pseudo-inverse matrix of matrix H
I	identity matrix of conformal dimension
O	all-zero matrix of conformal dimension
x_k	state vector of a dynamical system at time instance k
x_k^p	prediction (prior) for the state x_k at time instance k
x_k^{est}	estimate of the state vector x_k at time instance k
EKF	the extended Kalman filter
EnKF	the ensemble Kalman filter
KF	the Kalman filter
L-BFGS	low-memory Broyden-Fletcher-Goldfarb-Shanno unconstrained optimization scheme
MAP-estimate	maximum a posteriori estimate
PDF	probability density function
VKF	the variational Kalman filter

1 Introduction

Data assimilation is a process that splices numerically computed predictions and the corresponding measured data. The idea is to reduce natural errors of the mathematical predictions by correcting them using the real data in a certain "optimal" way. Here the optimality is presumed to have particular statistical sense, which usually means that the estimates calculated by data assimilation have to be unbiased and their variance must be bounded (or often minimal).

One of the common approaches that provides such optimal estimates is the Kalman filter (KF) Kalman (1960). The filter leverages known statistics of the observation and prediction errors in order to compute weighted average between the observed and the simulated data. The main constraint of this classical approach is that it only works when the evolution and the observation models are linear, which limits the application possibilities. A well-known extension that removes this restriction is the extended Kalman Filter (EKF) Simon (2006). The idea behind this method is to use the non-linear operators injected per-se into the usual framework of the classical Kalman filter and then apply linearizations when necessary. This implies that the EKF estimates converge towards that of the KF given infinitesimal time stepping of the evolution operator.

Both the KF and the EKF provide optimal data assimilation solutions. However, when dimension of the underlying system increases their implementation becomes impracticable due to the memory issues that arise from the formulation of these algorithms. Such high-dimensional systems are often considered for example in numerical weather prediction Fisher and Adresson (2001) and in oceanography Barth et al. (2010). Therefore, sub-optimal approximations of the classical methods are needed.

One of the natural approximations of the EKF is to draw many samples from the prediction distribution by slightly perturbing the given initial data and then computing prediction for each of the perturbations. The given set of prediction samples is then corrected by observation statistics calculated by the means of usual Kalman filter framework. This approach is known as the ensemble Kalman Filter (EnKF) Evensen (1994). Certain advantages of this algorithm are its simplicity and natural potential for parallelism Houtekamer et al. (2014). However, these benefits come at a cost as there is no rigourously justified method to define how many initial perturbations are needed to correctly capture the state of the underlying system at any time instance. In addition, such pre-defined set of perturbations may degrade over time, which leads to decay in overall performance of the filter Evensen (2004).

Another family of suboptimal methods that do not explicitly approximate the Kalman filter, but are based on similar bayesian framework, is constituted by variational data assimilation algorithms. These are the 3D-VAR, which roughly speaking is just a weighted least-square estimate that fits given prior to provided data and ignores the dynamics; the 4D-VAR, which adds evolution model but ignores presence of prediction error (see Kalnay (2007)), and the most recent by the moment of writing this text the weak-constraint 4D-VAR, which goes beyond the boundaries permitted by the usual Kalman filtering by enabling possibility for the predicted states to vary during the optimization pass Fisher (2009).

It is possible to relate the aforementioned variational approaches with the Kalman filter by using the fact that the data assimilation problem is essentially similar to parameter estimation and therefore the estimates computed by the Kalman filter can be replaced by usual MAP-estimates from parameter fitting routines. This allows to restate the Kalman filter (and its extended version) as a MAP cost function with the estimate being calculated as its minimizer. This way to represent the Kalman filter is called variational formulation.

Naturally, the variational formulation of the Kalman filter enables extra possibilities to do approximate data assimilation. Since the related MAP cost function can usually be implemented as a subroutine and does not require explicit memory storage, one immediate way to approximate the minimization procedure is to employ Quasi-Newton optimization routines (see for example Auvinen et al. (2009b) and Bardsley et al. (2011)). In addition, this very same idea can be applied also to the standard formulation of the Kalman filter by replacing the problematic matrix computations by related auxiliary optimization tasks that are then solved by Quasi-Newton optimization schemes. In this approach the restrictively large matrices are replaced by Hessians approximated during minimization. Due to iterative nature of the optimization, the approximate Hessians can often be implemented in low-memory fashion with the details of such implementation depending on the optimization routine. These ideas have been in detail considered by Auvinen et al. (2009a), Auvinen et al. (2009b) and Bardsley et al. (2011).

In this thesis we concentrate on explicit approximations of the extended Kalman Filter. We first demonstrate instability that may occur when implementing previously suggested approaches and then provide stabilizing correction that by the means of a simple change introduced into the original formulae ensures stability of the optimization process and improves convergence of the approximation towards the optimal solution of the EKF. Secondly, we use the improved estimation performance provided by our stabilizing correction in order to formulate and solve so-called parallel data assimilation task, which is equivalent to several data assimilation cycles combined together. We demonstrate our methods by implementing a data assimilation system based on them on top of the two-layer quasi-geostrophic model Fandry and Leslie (1984), which is one of the used toy-cases employed for benchmarking of data assimilation algorithms Fisher (2009). Finally, we apply our methods for estimation of amount of chlorophyll in the Finnish gulf based on high-resolution satellite images.

Not taking into account this introduction the thesis is organized as follows. The second chapter contains main motivation behind the data assimilation algorithms and provides overview and derivations of the classical formulations of the Kalman filters. We decided to include these derivations since they constitute the ground for our stabilizing corrections, which are amongst the main results of this work. The third chapter develops the ideas from the second chapter and introduces approximative data assimilation solutions based on the Kalman filter. The fourth chapter is the main part of this dissertation. It discusses the stabilizing corrections and sketches out the properties that ensure their mathematical consistency. In this chapter we also briefly describe the possibility to use the stabilizing corrections in order to come up with low-memory formulation of the fixed-lag Kalman smoother, which in this text we refer to as the parallel filtering task or the parallel filter.

The fourth chapter is wrapped out by a short review of the two-layer quasi-geostrophic model employed in our experimental runs. Finally, in the fifth chapter we consider a practical application of estimating the amount of chlorophyll in the coastal regions of the Finnish gulf during the summer months. In this chapter we use the data provided by Finnish Environmental Center (Suomen ympäristökeskus – SYKE) and demonstrate how the parallel filter can be implemented to solve the aforementioned estimation problem. The dissertation is finalized by a short summary section containing the main results of the present work and a brief overview of ongoing projects related to the problems considered in this text.

2 Classical data assimilation approaches

In this chapter we briefly consider canonical methods that historically and logically precede the main findings of the present work. We begin by introducing model parameter estimation problem and describe its classical solution obtained via least-squares method. Then we introduce the usual framework for data assimilation and discuss the linear Kalman filter and its extension for the non-linear case. In addition we discuss relation between the Kalman filter and the least-squares problem by shortly covering the variational formulation of the Kalman filter.

2.1 Parameter Fitting and Least Squares Method

Consider function $y = f(x, \theta)$. We will call $x \in \mathbb{R}^n$ vector of control variables (or for brevity control vector), $\theta \in \mathbb{R}^k$ is vector of adjustable model parameters (parameter vector), and $y \in \mathbb{R}$ is called the model yield. Assume that $x_i \in \mathbb{R}^n$, and $y_i \in \mathbb{R}$ where $i = 1..N$ are the set of control vectors that define certain experiment and the set of corresponding experimental results. Naturally, here N is the total number of experiments conducted. Next, assume that the experiment is analytically modelled by previously introduced function $y = f(x, \theta)$, and that the parameter vector θ is unknown. Our task is to identify θ in such way that the yields from the analytical model will be as close to the experimentally observed data as possible. More precisely, assume that $f(\bar{x}, \theta) = (f(x_1, \theta), \ldots, f(x_N, \theta))^T$ is vector of model yields and $\bar{y} = (y_1, \ldots, y_N)$ is vector of observed results. Then the problem we need to solve can be formulated as follows:

$$\|f(\bar{x}, \theta) - \bar{y}\| \underset{\theta}{\rightarrow} min. \tag{2.1}$$

Here $\|\cdot\|$ means certain norm (usually the Euclidean distance) allowing to compare the model yields against the observed values. Minimization task (2.1) is called least-squares problem and is generally non-linear and can be solved numerically by a wisely chosen optimization scheme. However, some unified theory can be developed for the linear case. Below we will briefly describe the main results of this theory as they lead to important motivations then applied in data assimilation algorithms.
We assume hereinafter that model function $f(x, \theta)$ is linear with respect to θ, or more precisely $f(\theta) = H\theta$ for some N-by-k matrix H. The rows of matrix H now play the role of control vectors. In addition, for the reasons of brevity we will from now on presume that all vectors are column-vectors and their dimension is such that the corresponding matrix-vector products are well defined. The problem (2.1) can be reformulated in the following way:

$$\|H\theta - \bar{y}\|_{\mathbb{E}} \underset{\theta}{\rightarrow} min. \tag{2.2}$$

Recall that in Euclidean space $\mathbb{E}^n$ for every linear subspace $L \subseteq \mathbb{E}^n$ each vector $x \in \mathbb{E}^n$ can be decomposed into its projection onto L and the corresponding orthogonal part, i.e. $x = x_o + x_p$, where x_o is orthogonal to L and $x_p \in L$. Hence, coming back to the problem (2.2) we can decompose observation vector $\bar{y} = \bar{y}_o + \bar{y}_p$, where $\bar{y}_p$ is projection of the

observation vector onto the image of matrix H and $\bar{y}_o$ is the corresponding orthogonal addendum. Hence, any vector $\hat{\theta} \in \mathbb{E}^k$ such that $H\hat{\theta} = y_p$ minimizes the cost function from (2.2). Indeed, consider the following sequence of equalities:

$$\begin{aligned} \|H\theta - \bar{y}\|_{\mathbb{E}}^2 &= \|H\theta - \bar{y}_p - \bar{y}_o\|_{\mathbb{E}}^2 \\ &= \|H\theta - \bar{y}_p\|_{\mathbb{E}}^2 + \|\bar{y}_o\|_{\mathbb{E}}^2 \end{aligned} \tag{2.3}$$

We used the fact that $(H\theta - \bar{y}_p)$ lies in the image of H and $\bar{y}_o$ is orthogonal to it and then employed the Pythagoras theorem. It is now obvious that (2.3) is minimized if and only if $H\hat{\theta} = y_p$.

Unfortunately, suggested solution is not always unique. This is often inconvenient and we want to provide an additional constraint in order to guarantee uniqueness in all cases. In the described framework of linear least-square problems the commonly applied constraint is that the norm of the obtained solution should also be minimal. Therefore the problem is reformulated as follows:

$$\begin{aligned} &\|H\theta - \bar{y}\|_{\mathbb{E}} \underset{\theta}{\rightarrow} min, \\ &\|\theta\|_{\mathbb{E}} \rightarrow min, \end{aligned} \tag{2.4}$$

i.e. we seek for solution of problem (2.1) having the minimal norm. This requirement is motivated by desire to obtain solution, which in physical sense is having minimal energy. In addition, it has certain statistical advantages corresponding to minimization of noise in the resolved set of parameters.

Let us now solve problem (2.4). We already know that the solution we seek for must satisfy equation $H\hat{\theta} = \bar{y}_p$. Notice that such solution always exists since $\bar{y}_p$ is drawn from the image of H. In order to also guarantee minimality of $\|\theta\|_{\mathbb{E}}$ we again use vector decomposition, i.e. for some θ such that $H\theta = \bar{y}_p$, we can write $\theta = \theta_o + \theta_p$, where now θ_p belongs to the kernel of matrix H and θ_o is the corresponding orthogonal counterpart. It can be trivially shown (see Albert (1972)) that θ_o lies in the image of matrix H^T. In addition, $\|\theta\|_{\mathbb{E}}^2 = \|\theta_p\|_{\mathbb{E}}^2 + \|\theta_o\|_{\mathbb{E}}^2$ and we can conclude that $\|\theta\|_{\mathbb{E}}$ is minimal if and only if $\theta_p = \bar{0}$. In other words we have proved, that for any solution $\hat{\theta}$ of (2.2) it is possible to find solution θ such that $\|\theta\|_{\mathbb{E}} \leq \|\hat{\theta}\|_{\mathbb{E}}$ and $\theta \in \text{Img}(H^T)$.

What remains to be shown is that the solution with the aforementioned properties is unique. Indeed, assume that there are two separate solutions θ_1 and θ_2 such that $\theta_{1,2} \in \text{Img}(H^T)$ and $H\theta_{1,2} = \bar{y}_p$ where $\bar{y}_p$ is as before the projection of vector $\bar{y}$ onto $\text{Img}(H)$. Since $\theta_{1,2} \in \text{Img}(H^T)$, then some $\tilde{y}_1$ and $\tilde{y}_2$ satisfy equations $H^T\tilde{y}_1 = \theta_1$ and $H^T\tilde{y}_2 = \theta_2$. Consider now the following sequence of identities:

$$\begin{aligned} &\bar{0} = H\theta_1 - H\theta_2 = HH^T\tilde{y}_1 - HH^T\tilde{y}_2 = \\ &HH^T\left(\tilde{y}_1 - \tilde{y}_2\right). \end{aligned}$$

Multiplying the rightmost and the leftmost part of this sequence by $\left(\tilde{y}_1 - \tilde{y}_2\right)^T$ we arrive

at the following:

$$0 = \left(\tilde{y}_1 - \tilde{y}_2\right)^T HH^T \left(\tilde{y}_1 - \tilde{y}_2\right) =$$
$$\left\| H^T \left(\tilde{y}_1 - \tilde{y}_2\right) \right\|_{\mathbb{E}}^2 = \left\| H^T \tilde{y}_1 - H^T \tilde{y}_2 \right\|_{\mathbb{E}}^2 =$$
$$\left\| \theta_1 - \theta_2 \right\|_{\mathbb{E}}^2,$$

which means that $\theta_1 = \theta_2$, i.e. the solution θ of (2.2) such that $\theta \in \text{Img}(H^T)$ is unique. On the other hand by the arguments above such solution can be constructed for any other solution of (2.2) without increasing the norm. Therefore, θ solves (2.4) if and only if $\theta \in \text{Img}(H^T)$ and $H\theta = \bar{y}_p$.
We can summarize our findings in the following lemma:

Lemma 1. *The problem* (2.4) *has unique solution θ such that:*

- $H\theta = \bar{y}_p$*, where $\bar{y}_p$ is projection of observation vector $\bar{y}$ onto* $\text{Img}(H)$*,*
- $\theta \in \text{Img}\left(H^T\right)$.

These characterization properties of solution are convenient for theoretical justifications, but the possibility to use them per se in calculations is rather limited. In order to derive convenient formula to calculate the solution, we will need one more definition.

Definition 1. *Consider an n-by-m matrix H. An m-by-n matrix H^+ is called pseudo-inverse matrix of matrix H when it is computed as follows:*

$$H^+ = \lim_{\sigma \to 0} H^T \left(HH^T + \sigma I\right)^{-1}, \tag{2.5}$$

where I is n-by-n identity matrix.

First of all we should notice, that matrix $\left(HH^T + \sigma I\right)$ from (2.5) is always invertible when σ is smaller then the smallest eigenvalue of HH^T and hence the limit is well-defined. Secondly, it is easy to see that when matrix H is itself invertible then $H^+ = H^{-1}$, i.e. pseudo-inversion is generalization of the normal matrix inversion. Finally, the pseudo-inverse matrix can be equivalently defined as follows:

$$H^+ = \lim_{\sigma \to 0} \left(H^T H + \sigma I\right)^{-1} H^T, \tag{2.6}$$

where now I is m-by-m identity matrix. Indeed, consider the following matrix identity:

$$H^T \left(HH^T + \sigma I\right) = \left(H^T H + \sigma I\right) H^T.$$

Multiplying this identity by $\left(HH^T + \sigma I\right)^{-1}$ from the left and by $\left(H^T H + \sigma I\right)^{-1}$ from the right we get the following:

$$\left(H^T H + \sigma I\right)^{-1} H^T = H^T \left(HH^T + \sigma I\right)^{-1},$$

which immediately implies equivalence of (2.5) and (2.6).
We are left to construct relation between pseudo-inversion and solution of the least-squares problems. This relation is completely clarified by the following lemma. This is a classical result, however it contains motivation for derivation of the Kalman filter, which is thereafter used to get the idea of the stabilizing correction. Since the latter is one of the main results of this work we decided to provide a short prove of relation between the least-squares problem and the pseudo-inversion here for the reasons of completeness. For more detailed discussion one can refer to Albert (1972).

Lemma 2. *The following suggestions hold:*

1. *For any given matrix H pseudo-inverse matrix H^+ always exists,*
2. *Solution for least-squares problem* (2.4) *is given by $H^+\bar{y}$*
3. *When H^TH is invertible, then $H^+ = \left(H^TH\right)^{-1}H^T$*

Proof. 1) We need to proof that the limit $H^+ = \lim\limits_{\sigma\to 0}\left(H^TH+\sigma I\right)^{-1}H^T$ exists for any matrix H. We begin by proving slightly simpler fact that for any symmetric matrix A the limit $\lim\limits_{\sigma\to 0}\left[(A+\sigma I)^{-1}A\right]$ always exists and moreover for any $\bar{y}$ the operator defined by this limit projects $\bar{y}$ onto $\mathrm{Img}(A)$. More precisely

$$\lim_{\sigma\to 0}(A+\sigma I)^{-1}A\bar{y} = \bar{y}_p,$$

where $\bar{y}_p$ is projection of $\bar{y}$ onto $\mathrm{Img}(A)$. We again decompose vector $\bar{y}$ so that $\bar{y} = \bar{y}_p + \bar{y}_o$, where $\bar{y}_p \in \mathrm{Img}(A)$ is projection of $\bar{y}$ onto $\mathrm{Img}(A)$ and $\bar{y}_o \in \mathrm{Ker}(A)$ is the corresponding orthogonal component. In addition, by the matrix spectral theorem $A = RDR^T$, where D is diagonal matrix of eigenvalues of A and R is orthogonal matrix, i.e. $RR^T = R^TR = I$. Notice that since $\bar{y}_p \in \mathrm{Img}(A)$, then there is such $\tilde{x}$ that $A\tilde{x} = \bar{y}_p$. Hence, we can write the following:

$$(A+\sigma I)^{-1}A\bar{y} = (A+\sigma I)^{-1}A\bar{y}_p = (A+\sigma I)^{-1}A^2\tilde{x} = \\ R(D+\sigma I)^{-1}D^2R^T\tilde{x}.$$

Considering the latter identity component-wise it is easy to see that $\lim\limits_{\sigma\to 0}(D+\sigma I)^{-1}D^2 = D$. Therefore:

$$\lim_{\sigma\to 0}(A+\sigma I)^{-1}A\bar{y} = RDR^T\tilde{x} = A\tilde{x} = \bar{y}_p.$$

Now, consider limit $H^+ = \lim\limits_{\sigma\to 0}\left(H^TH+\sigma I\right)^{-1}H^T$. For any $\bar{y} = \bar{y}_p + \bar{y}_o$, where $\bar{y}_p \in \mathrm{Img}(H)$ and $\bar{y}_o \in \mathrm{Ker}(H^T)$ we have

$$\lim_{\sigma\to 0}\left(H^TH+\sigma I\right)^{-1}H^T\bar{y} = \lim_{\sigma\to 0}\left(H^TH+\sigma I\right)^{-1}H^TH\tilde{x},$$

for some $\tilde{x}$ such that $\bar{y}_p = H\tilde{x}$. Therefore, using the justifications provided before we can conclude that $\lim\limits_{\sigma\to 0}\left(H^TH+\sigma I\right)^{-1}H^TH\tilde{x} = \tilde{x}_p$, where $\tilde{x}_p$ is projection of $\tilde{x}$ onto

Img $\left(H^T H\right)$. Since $\bar{y}$ was selected arbitrarily, the limit in question always exists and the first part of the lemma holds.

2) It is enough to verify that requirements from Lemma 1 hold. Assume that $\bar{y}$ is the observation vector. Then by the proof of part 1 of this lemma, we conclude that $H^+\bar{y} = \tilde{x}_p$ is projection of $\tilde{x}$ onto Img $\left(H^T H\right)$ where $\tilde{x}$ is a vector such that $\bar{y}_p = H\tilde{x}$ and $\bar{y}_p$ is projection of $\bar{y}$ onto Img(H). In order to prove that $\tilde{x}_p$ is solution for problem (2.4) it is enough to show that $H\tilde{x}_p = \bar{y}_p$. Notice however that Ker $\left(H^T H\right) =$ Ker (H). Indeed, obviously Ker$(H) \subseteq$ Ker $\left(H^T H\right)$. Conversely, if $x \in$ Ker $\left(H^T H\right)$, then $x^T H^T H x = \|Hx\|_E^2 = 0$, which means that $Hx = \bar{0}$ and $x \in$ Ker(H). Hence, $\tilde{x} = \tilde{x}_p + \tilde{x}_o$, where as before $\tilde{x}_p$ is projection of $\tilde{x}$ onto Img $\left(H^T H\right)$ and $\tilde{x}_o \in$ Ker $\left(H^T H\right) =$ Ker(H). Finally, we have:

$$\bar{y} = H\tilde{x} = H\tilde{x}_p + H\tilde{x}_o = H\tilde{x}_p.$$

This finalizes the proof of the second part of the lemma.

3) The third part of the lemma obviously follows from the matrix spectral theorem applied to matrix $H^T H$. □

Lemma 2 provides universal solution for problem (2.4). However, throughout the following text unless stated otherwise we will limit ourselves to more special case with additional constraint of the matrix $H^T H$ having inverse. Hence, again by Lemma 2 the solution for problem (2.4) is given by $\left(H^T H\right)^{-1} H^T \bar{y}$. We wrap out this chapter by considering slightly more general case of the least squares problem with weighted semi-norm (i.e. it is not required that such norm computed for a non-zero vector must be positive):

$$\begin{aligned} &(H\theta - \bar{y})^T L\,(H\theta - \bar{y}) \underset{\theta}{\to} min, \\ &\|\theta\|_{\mathbb{E}} \to min, \end{aligned} \tag{2.7}$$

where L is a non-negative definite matrix. Assume that $L^{1/2}$ is square root of matrix L (since the matrix is non-negative definite it always exists). Then the problem can be restated as follows:

$$\begin{aligned} &\|L^{1/2}H\theta - L^{1/2}\bar{y}\|_{\mathbb{E}} \underset{\theta}{\to} min, \\ &\|\theta\|_{\mathbb{E}} \to min. \end{aligned}$$

Applying Lemma 2 we obtain solution for the problem (2.7) (for simplicity below we assume that $H^T LH$ is invertible):

$$\theta = \left(L^{1/2}H\right)^+ L^{1/2}\bar{y} = \left(H^T LH\right)^{-1} H^T L\bar{y}. \tag{2.8}$$

It can be shown that formula (2.8) provides the MAP estimate for θ given observation $\bar{y}$ when the errors in $\bar{y}$ are normally distributed and have zero mean. In section 2.4 we will utilize this fact to derive the variational formulation of the Kalman filter.

2.2 The problem of data assimilation

Data assimilation is process of combining analytically computed predictions with (possibly indirect) observations of the studied phenomenon (see Apte et al. (2008)). We begin stating rigorous mathematical formulation of the data assimilation problem by firstly introducing the notion of observed dynamical system. Consider the following system of coupled stochastic equations:

$$x_{k+1} = \mathcal{M}_k\left(x_k\right) + \epsilon_k, \tag{2.9}$$

$$y_{k+1} = \mathcal{H}_{k+1}\left(x_{k+1}\right) + \eta_{k+1}. \tag{2.10}$$

Here $x_k \in \mathbb{R}^n$ is vector, which completely describes the state of certain system at time instance k; $\mathcal{M}_k$ is evolution (transition) operator that models change of the system state between time instances k and $k+1$; operator $\mathcal{H}_{k+1}$ relates state of the system x_{k+1} at time instance $k+1$ with the corresponding observation $y_{k+1} \in \mathbb{R}^m$; the terms $\epsilon_k \in \mathbb{R}^n$ and $\eta_{k+1} \in \mathbb{R}^m$ are stochastic components of the equations and are included to model prediction and observation errors respectively.

The standard task of data assimilation in terms of equations (2.9) and (2.10) is the following: given observation y_{k+1} at time instance $k+1$ and estimate x_k^{est} for system state x_k at time instance k provide estimate x_{k+1}^{est} for system state x_{k+1} at time instance $k+1$. In addition, it is often required that the estimate x_{k+1}^{est} should be in some sense optimal. Optimality criteria may vary, but the common constraints are that the estimate should be unbiased, i.e. $\mathbb{E}\left[x_{k+1}^{est}\right] = x_{k+1}$ and it should have the minimal variance, i.e. among all unbiased estimates the variance of the estimate produced by the data assimilation system is expected to have the smallest variance.

In the next subsection we describe the Kalman filter – a classical tool employed to solve the data assimilation task in case where evolution operator $\mathcal{M}_k$ and observation operator $\mathcal{H}_{k+1}$ are linear. We also provide derivation of the filter taking into consideration the optimality criteria mentioned above, i.e. we seek for an estimate, which is unbiased and has the minimal variance.

2.3 Linear Kalman filter and its extension for non-linear models

In this section we consider the task of estimating x_{k+1} from observed dynamical system (2.9)-(2.10) for the case, where both the evolution operator $\mathcal{M}_k$ and the observation operator $\mathcal{H}_{k+1}$ are linear. To emphasize the linearity of the operators we will use slightly altered notation and define M_k and H_{k+1} to be the matrices representing the corresponding evolution and observation operators at given time instances. Additionally, we assume that C_{ϵ_k} and $C_{\eta_{k+1}}$ are covariance matrices of stochastic terms ϵ_k and η_{k+1} from (2.9)-(2.10). As was already pointed out in the previous section we are seeking for an unbiased estimate with minimal variance for x_{k+1} given the corresponding observation y_{k+1}, the estimate x_k^{est} from the preceding time instance along with its covariance matrix C_k^{est}, and the covariance matrices C_{ϵ_k} and $C_{\eta_{k+1}}$ of the error terms. The solution to this problem is unique and is given by algorithm 1 called Kalman filter.

Input : x_k^{est}, M_k, y_{k+1}, H_{k+1}, C_k^{est}, C_{ϵ_k}, $C_{\eta_{k+1}}$
Output: x_{k+1}^{est}, C_{k+1}^{est}

1 **foreach** time instance k **do**
  /*Compute prediction and its covariance */
2 $x_{k+1}^{p} \leftarrow M_k x_k^{est}$;
3 $C_{k+1}^{p} \leftarrow M_k C_k^{est} M_k^T + C_{\epsilon_k}$;
  /*Calculate Kalman Gain */
4 $G_{k+1} \leftarrow C_{k+1}^{p} H_{k+1}^T \left(H_{k+1} C_{k+1}^{p} H_{k+1}^T + C_{\eta_{k+1}}\right)^{-1}$;
  /*Correct predicted state and covariance */
5 $x_{k+1}^{est} \leftarrow x_{k+1}^{p} + G_{k+1}\left(y_{k+1} - H_{k+1} x_{k+1}^{p}\right)$;
6 $C_{k+1}^{est} \leftarrow C_{k+1}^{p} - G_{k+1} H_{k+1} C_{k+1}^{p}$;
7 **end**

Algorithm 1: Kalman Filter

Algorithm 1 can be contingently divided into prediction part (steps 2 and 3) and correction part (steps 4–6). Note that the algorithm computes not only the estimate x_{k+1}^{est}, but also its covariance matrix C_{k+1}^{est}, which allows to iterate the algorithm continuously computing estimates for the future states as soon as the corresponding observations become available.

It remains to demonstrate that formulated algorithm indeed produces unbiased estimate with the minimal variance. In addition, it will be shown that such estimate is unique. To justify these statements, we propose the following lemma.

Lemma 3. *Assume that the noise terms ϵ_k and η_{k+1} have zero means and are independent from each other. In addition, suppose that η_i and η_j as well as ϵ_i and ϵ_j are also independent when $i \neq j$. Furthermore, suppose that ϵ_i and η_j are independent from x_k and are not correlated with x_k^{est} when $i \geq k$ and $j \geq k+1$. Finally, assume that x_k^{est} is an unbiased estimator of x_k. Then x_{k+1}^{est} computed by algorithm 1 is an unbiased estimate of x_{k+1}, ϵ_i and η_j are uncorrelated with x_{k+1}^{est} when $i \geq k+1$ and $j \geq k+2$, and C_{k+1}^{est} computed by algorithm 1 solves the following optimization problem:*

$$\begin{gathered} \boldsymbol{tr}(C) \underset{C}{\rightarrow} min, \\ C \in \{C = Cov(\hat{x}_{k+1} - x_{k+1}) | \hat{x}_{k+1} : \mathbb{E}\left[\hat{x}_{k+1} - x_{k+1}\right] = 0\}. \end{gathered} \tag{2.11}$$

Proof. The proof is based on restricting the estimator to have desired properties and then constructing its representation in a closed algebraic form. If such estimator exists and coincides with the one defined in algorithm 1 then the claim of the lemma is true.
We will seek for the needed estimator in the class of weighted averages. Therefore, we assume:

$$x_{k+1}^{est} = A_k x_k^{est} + B_{k+1} y_{k+1}, \tag{2.12}$$

where A_k and B_{k+1} are the matrices of conformal dimensions. Since we require the estimator to be unbiased and since by assumption of the lemma x_k^{est} is already unbiased

and the noise terms are having zero mean, then by taking the mean of the both sides of (2.12) and accounting for the equations (2.9) and (2.10) we arrive at the following sequence of identities:

$$\begin{aligned}
&\mathbb{E}\left[x_{k+1}^{est}\right] = \mathbb{E}\left[A_k x_k^{est} + B_{k+1} y_{k+1}\right], \\
&\mathbb{E}\left[x_{k+1}\right] = A_k \mathbb{E}\left[x_k\right] + B_{k+1}\mathbb{E}\left[H_{k+1}x_{k+1} + \eta_{k+1}\right], \\
&\mathbb{E}\left[M_k x_k + \epsilon_k\right] = A_k \mathbb{E}\left[x_k\right] + B_{k+1}H_{k+1}\mathbb{E}\left[M_k x_k + \epsilon_k\right], \\
&M_k \mathbb{E}\left[x_k\right] = A_k \mathbb{E}\left[x_k\right] + B_{k+1}H_{k+1}M_k\mathbb{E}\left[x_k\right], \\
&\left(I - B_{k+1}H_{k+1}\right) M_k \mathbb{E}\left[x_k\right] = A_k \mathbb{E}\left[x_k\right].
\end{aligned}$$

Thereby, $A_k = \left(I - B_{k+1}H_{k+1}\right) M_k$ and (2.12) can be rewritten as follows:

$$x_{k+1}^{est} = x_{k+1}^{p} + B_{k+1}\left(y_{k+1} - H_{k+1}x_{k+1}^{p}\right), \tag{2.13}$$

where the term x_{k+1}^{p} is defined as in algorithm 1 and hereinafter called prediction (or prior). Note that by construction the estimate x_{k+1}^{est} in (2.13) is unbiased for any matrix B_{k+1} of conformal dimension. We now need to choose matrix B_{k+1} so that the corresponding C_{k+1}^{est} solves problem (2.11). First, we plug the expression for x_{k+1}^{est} from (2.13) into the estimate error $x_{k+1}^{est} - x_{k+1}$ and account for the observation model (2.10):

$$x_{k+1}^{est} - x_{k+1} = \left(I - B_{k+1}H_{k+1}\right)\left(x_{k+1}^{p} - x_{k+1}\right) + B_{k+1}\eta_{k+1}. \tag{2.14}$$

Second, by definition of C_{k+1}^{est} and since the estimate error $x_{k+1}^{est} - x_{k+1}$ has zero mean we get to the following sequence of identities:

$$\begin{aligned}
&C_{k+1}^{est} = \mathbb{E}\left[\left(x_{k+1}^{est} - x_{k+1}\right)\left(x_{k+1}^{est} - x_{k+1}\right)^T\right] = \\
&\left(I - B_{k+1}H_{k+1}\right)\mathbb{E}\left[\left(x_{k+1}^{p} - x_{k+1}\right)\left(x_{k+1}^{p} - x_{k+1}\right)^T\right]\left(I - B_{k+1}H_{k+1}\right)^T + \\
&\left(I - B_{k+1}H_{k+1}\right)\mathbb{E}\left[\left(x_{k+1}^{p} - x_{k+1}\right)\eta_{k+1}^T\right]B_{k+1}^T + \\
&B_{k+1}\mathbb{E}\left[\eta_{k+1}\left(x_{k+1}^{p} - x_{k+1}\right)^T\right]\left(I - B_{k+1}H_{k+1}\right)^T + B_{k+1}\mathbb{E}\left[\eta_{k+1}\eta_{k+1}^T\right]B_{k+1}^T.
\end{aligned}$$

We begin analysis of the derived expression for C_{k+1}^{est} by computing the prediction error covariance. Since ϵ_k is independent from x_k and x_k^{est}, then making use of transition model (2.9) and resorting to the notation from algorithm 1 we arrive at the following:

$$\begin{aligned}
&C_{k+1}^{p} = \mathbb{E}\left[\left(x_{k+1}^{p} - x_{k+1}\right)\left(x_{k+1}^{p} - x_{k+1}\right)^T\right] = \\
&\mathbb{E}\left[\left(M_k(x_k^{est} - x_k) - \epsilon_k\right)\left(M_k(x_k^{est} - x_k) - \epsilon_k\right)^T\right] = \\
&M_k\mathbb{E}\left[\left(x_k^{est} - x_k\right)\left(x_k^{est} - x_k\right)^T\right]M_k^T - \\
&\mathbb{E}\left[\epsilon_k\left(x_k^{est} - x_k\right)^T\right]M_k^T - M_k\mathbb{E}\left[\left(x_k^{est} - x_k\right)\epsilon_k^T\right] + \mathbb{E}\left[\epsilon_k\epsilon_k^T\right] = \\
&M_k C_k^{est} M_k^T + C_{\epsilon_k}.
\end{aligned}$$

Since η_{k+1} does not depend on x_k, x_k^{est} and ϵ_k, then η_{k+1} and $x_{k+1}^p - x_{k+1}$ are uncorrelated and the corresponding covariances vanish. Therefore, using notation from algorithm 1 and leveraging the obtained formula for C_{k+1}^p we can simplify the expression for C_{k+1}^{est}:

$$C_{k+1}^{est} = (I - B_{k+1}H_{k+1})\, C_{k+1}^p\, (I - B_{k+1}H_{k+1})^T + B_{k+1}C_{\eta_{k+1}}B_{k+1}^T. \tag{2.15}$$

Using (2.15) we obtain expression for the trace of C_{k+1}^{est}:

$$\begin{aligned}\mathbf{tr}\left(C_{k+1}^{est}\right) =& \mathbf{tr}\left(C_{k+1}^p\right) - 2\mathbf{tr}\left(B_{k+1}H_{k+1}C_{k+1}^p\right) + \\ & \mathbf{tr}\left(B_{k+1}H_{k+1}C_{k+1}^pH_{k+1}^TB_{k+1}^T\right) + \mathbf{tr}\left(B_{k+1}C_{\eta_{k+1}}B_{k+1}^T\right)\end{aligned} \tag{2.16}$$

In order to solve problem (2.11) matrix C_{k+1}^{est} has to satisfy the following equation:

$$\frac{\partial \mathbf{tr}\left(C_{k+1}^{est}\right)}{\partial B_{k+1}} = O. \tag{2.17}$$

The term in the left-hand side of (2.17) denotes the matrix of partial derivatives of function $\mathbf{tr}\left(C_{k+1}^{est}\right)$ with respect to each element of B_{k+1}; the O in the right hand side here denotes all-zero matrix of the conformable dimensions. This equations allows us to obtain relation, which determines B_{k+1}.
In order to compute the gradient in (2.17) we use auxiliary propositions:

- If A and B are arbitrary matrices of conformal dimensions, then the following identity holds:
$$\frac{\partial \mathbf{tr}\,(BA)}{\partial B} = A^T.$$
- If A and B are arbitrary matrices of conformal dimensions and A is symmetric, then
$$\frac{\partial \mathbf{tr}\left(BAB^T\right)}{\partial B} = 2AB.$$

Verification of these propositions is straightforward, so we skip it here for conciseness.
Substitution of (2.16) into (2.17) and the aforementioned propositions imply the constraint for B_{k+1}:

$$\begin{aligned}&-2C_{k+1}^pH_{k+1}^T + 2H_{k+1}C_{k+1}^pH_{k+1}^TB_{k+1} + 2C_{\eta_{k+1}}B_{k+1} = 0,\\ &\left(H_{k+1}C_{k+1}^pH_{k+1}^T + C_{\eta_{k+1}}\right)B_{k+1} = C_{k+1}^pH_{k+1}^T,\\ &B_{k+1} = C_{k+1}^pH_{k+1}^T\left(H_{k+1}C_{k+1}^pH_{k+1}^T + C_{\eta_{k+1}}\right)^{-1}.\end{aligned} \tag{2.18}$$

Hereafter, we denote the right-hand side in (2.18) by G_{k+1} and call it the Kalman gain. Putting together (2.15) and (2.18) we get the formula for the estimate error covariance C_{k+1}^{est}, which solves (2.11):

$$C_{k+1}^{est} = C_{k+1}^p - G_{k+1}H_{k+1}C_{k+1}^p. \tag{2.19}$$

Substituting (2.18) into (2.13) we derive the final formula for the estimate:

$$x_{k+1}^{est} = x_{k+1}^{p} + G_{k+1}\left(y_{k+1} - H_{k+1}x_{k+1}^{p}\right). \tag{2.20}$$

Comparing (2.19) and (2.20) to the corresponding entities from algorithm 1 we are able to conclude that the constructed estimator is the same as the one defined by the algorithm. In order to finish the proof, we still need to check that ϵ_i and η_j are uncorrelated with x_{k+1}^{est} when $i \geq k+1$ and $j \geq k+2$. This fact however immediately follows from (2.20), observation equation (2.10) and the assumptions we made regarding the error terms ϵ_i and η_j. □

The results of Lemma 3 are well known, however the derivation we provided in this dissertation contains the main ideas to use when deriving the stabilizing corrections discussed in detail in section 4.1.
Based on Lemma 3 we see that Algorithm 1 gives optimal solution to the data assimilation problem. However, its practical applicability is severely limited due to assumptions of linearity applied to evolution and observation models. Therefore, we would like to generalize the result provided by Algorithm 1 for the non-linear case. In order to do that we replace matrix representations M_k and H_{k+1} of previously linear operators $\mathcal{M}_k$ and $\mathcal{H}_{k+1}$ by the corresponding linearizations, i.e. we re-define them as follows:

$$M_k = \frac{\partial \mathcal{M}_k(x)}{\partial x}, \quad H_{k+1} = \frac{\partial \mathcal{H}_{k+1}(x)}{\partial x}, \tag{2.21}$$

where $\frac{\partial \cdot(x)}{\partial x}$ denotes Jacobian of certain operator calculated at point x. We can now employ the new meaning of M_k and H_{k+1} in order to correct Algorithm 1 for non-linear cases as summarized below:

- In step 1 of Algorithm 1 use non-linear operator $\mathcal{M}_k$ instead of matrix M_k, i.e. prediction is now computed as $x_{k+1}^{p} = \mathcal{M}_k\left(x_k^{est}\right)$.
- Use new definitions of M_k and H_{k+1} everywhere else in the algorithm in the places where the corresponding matrix representations of the evolution and of the observation operators were used in the linear case.

This generalization of Algorithm 1 for non-linear cases is referred to as the Extended Kalman Filter (EKF). It can be shown that even in non-linear cases when some specific requirements hold, this algorithm remains in certain sense optimal Simon (2006). In addition, one could easily check that when the evolution and the observation operators are linear, the EKF becomes fully equivalent to the basic Kalman filter as given by Algorithm 1.
This wraps out our short review of the Kalman filter. In our work we often leverage the EKF as the "ground truth" reference tool to provide a reference baseline, which allows us to estimate performance of out approaches. In the following section we discuss one completely different way to derive the Kalman filter, which relates it with the least-squares minimization considered in the beginning of this chapter and also provides connection between the Kalman filter and unconstrained optimization problems.

2.4 Variational formulation of the Kalman filter

In this section we would like to derive Algorithm 1 using completely different approach from the one provided previously. Now, instead of attempting to estimate the state of (2.9)-(2.10) assuming that the estimate should possess certain properties, we reformulate the data assimilation task as a Bayesian problem, derive posterior distribution of the estimate given the observation and maximize it with respect to the unknown state. The estimation obtained this way is often called maximum a posteriori estimate or MAP. Let us consider PDF of state x_{k+1} given the corresponding observation y_{k+1}. Then, in accordance with the Bayes theorem, we get the following:

$$p(x_{k+1}|y_{k+1}) = \frac{p(y_{k+1}|x_{k+1})p(x_{k+1})}{p(y_{k+1})}. \tag{2.22}$$

Instead of seeking for an unbiased estimate of x_{k+1} having the minimal variance, we will try to maximize $p(x_{k+1}|y_{k+1})$ with respect to x_{k+1}.

From now on let us assume in addition to assumptions made in Lemma 3 that the noise terms ϵ_k and η_{k+1} from (2.9)-(2.10) and the estimate x_k^{est} are normally distributed. In addition, for clarity we will again revert to the linear case, which implies that Jacobians M_k and H_{k+1} from (2.21) are now again just matrix representations of operators $\mathcal{M}_k$ and $\mathcal{H}_{k+1}$ respectively. Then x_{k+1} is normally distributed and using (2.9) we can compute its mean value and covariance matrix:

$$\mathbb{E}\left[x_{k+1}\right] = M_k\mathbb{E}\left[x_k\right] = M_k\mathbb{E}\left[x_k^{est}\right] = x_{k+1}^p, \tag{2.23}$$

$$\mathbb{E}\left[\left(x_{k+1} - x_{k+1}^p\right)\left(x_{k+1} - x_{k+1}^p\right)^T\right] = C_{k+1}^p, \tag{2.24}$$

where C_{k+1}^p is as defined in algorithm 1 and (2.24) has already been derived in the proof of Lemma 3. Hence,

$$p(x_{k+1}) \propto \mathbf{exp}\left(-\frac{1}{2}\left(x_{k+1} - x_{k+1}^p\right)^T\left[C_{k+1}^p\right]^{-1}\left(x_{k+1} - x_{k+1}^p\right)\right). \tag{2.25}$$

Similarly to $p(x_{k+1})$ we can obtain $p(y_{k+1}|x_{k+1})$ by leveraging the observation model (2.10). Indeed, we first notice that in accordance with (2.10) and since the observation operator is linear the distribution determined by $p(y_{k+1}|x_{k+1})$ is a normal distribution. Hence, we again can compute its mean value and covariance matrix:

$$\mathbb{E}\left[y_{k+1}|x_{k+1}\right] = H_{k+1}x_{k+1}, \tag{2.26}$$

$$\mathbb{E}\left[\left(y_{k+1} - H_{k+1}x_{k+1}\right)\left(y_{k+1} - H_{k+1}x_{k+1}\right)^T\right] = C_{\eta_{k+1}}. \tag{2.27}$$

Here the last identity in (2.27) is implied by the fact that $\mathbb{E}\left[y_{k+1} - H_{k+1}x_{k+1}\right] = \mathbb{E}\left[\eta_{k+1}\right]$.

Therefore, the PDF $p(y_{k+1}|x_{k+1})$ can be stated as follows:

$$p(y_{k+1}|x_{k+1}) \propto \mathbf{exp}\left(-\frac{1}{2}\left(y_{k+1} - H_{k+1}x_{k+1}\right)^T C_{\eta_{k+1}}^{-1}\left(y_{k+1} - H_{k+1}x_{k+1}\right)\right) \tag{2.28}$$

Finally, plugging (2.25) and (2.28) into (2.22) and taking $-\mathbf{log}(\cdot)$ of the result we get the final formulation of our data assimilation problem:

$$\begin{aligned} -\mathbf{log}\left(p(x_{k+1}|y_{k+1})\right) \propto &\left(x_{k+1} - x_{k+1}^p\right)^T \left[C_{k+1}^p\right]^{-1}\left(x_{k+1} - x_{k+1}^p\right) + \\ &\left(y_{k+1} - H_{k+1}x_{k+1}\right)^T C_{\eta_{k+1}}^{-1}\left(y_{k+1} - H_{k+1}x_{k+1}\right). \end{aligned} \tag{2.29}$$

It is possible to restate (2.29) in the form of (2.7) with the following values for matrices C, H, and vector $\bar{y}$:

$$\begin{aligned} C &= \begin{pmatrix} \left[C_{k+1}^p\right]^{-1} & O \\ O & C_{\eta_{k+1}}^{-1} \end{pmatrix}, \\ H &= \begin{pmatrix} I \\ H_{k+1} \end{pmatrix}, \\ \theta &= x_{k+1}, \\ \bar{y} &= \begin{pmatrix} x_{k+1}^p \\ y_{k+1} \end{pmatrix}, \end{aligned} \tag{2.30}$$

where the O-entries are zero matrix blocks and I is identity matrix of respective dimensions. Then, in accordance with (2.8) we can deduce the estimate for x_{k+1}:

$$x_{k+1}^{est} = \left(\left[C_{k+1}^p\right]^{-1} + H_{k+1}^T C_{\eta_{k+1}}^{-1} H_{k+1}\right)^{-1}\left(\left[C_{k+1}^p\right]^{-1} x_{k+1}^p + H_{k+1}^T C_{\eta_{k+1}}^{-1} y_{k+1}\right). \tag{2.31}$$

It remains to show that (2.31) is equivalent to (2.20). Applying the Woodbury matrix identity to the first term in (2.31) and using definition of the Kalman gain G_{k+1} from (2.18) we can rearrange (2.31) as follows:

$$\begin{aligned} x_{k+1}^{est} &= \left(C_{k+1}^p - G_{k+1}H_{k+1}C_{k+1}^p\right)\left(\left[C_{k+1}^p\right]^{-1} x_{k+1}^p + H_{k+1}^T C_{\eta_{k+1}}^{-1} y_{k+1}\right) \\ &= \left(x_{k+1}^p - G_{k+1}H_{k+1}x_{k+1}^p\right) + \\ &\quad + \left(C_{k+1}^p H_{k+1}^T C_{\eta_{k+1}}^{-1} - G_{k+1}H_{k+1}C_{k+1}^p H_{k+1}^T C_{\eta_{k+1}}^{-1}\right) y_{k+1}. \end{aligned} \tag{2.32}$$

The second term in the expression above can be further expanded using identity $C_{k+1}^p H_{k+1}^T = G_{k+1}\left(H_{k+1}C_{k+1}^p H_{k+1}^T + C_{\eta_{k+1}}\right)$ implied by definition of the Kalman gain from (2.18), which yields the following

$$G_{k+1}\left(H_{k+1}C_{k+1}^p H_{k+1}^T + C_{\eta_{k+1}}\right) C_{\eta_{k+1}}^{-1} - G_{k+1}H_{k+1}C_{k+1}^p H_{k+1}^T C_{\eta_{k+1}}^{-1} = G_{k+1} \tag{2.33}$$

Finally, plugging (2.33) into (2.32) we get

$$x_{k+1}^{est} = \left(x_{k+1}^{p} - G_{k+1}H_{k+1}x_{k+1}^{p}\right) + G_{k+1}y_{k+1} = x_{k+1}^{p} + G_{k+1}\left(y_{k+1} - H_{k+1}x_{k+1}^{p}\right),$$

which coincides with (2.20).
Let us now compute the covariance matrix C_{k+1}^{est} of x_{k+1}^{est}. Since x_k^{est}, ϵ_k, and η_{k+1} were assumed to be Gaussian and $x_{k+1}^{est} = (H^T C H)^{-1} H^T C \bar{y}$ (see (2.8) and (2.30)), the covariance of x_{k+1}^{est} can be calculated directly (below we leverage the fact that $\bar{y}$ has covariance matrix equal to C^{-1}):

$$\begin{aligned} C_{k+1}^{est} &= (H^T C H)^{-1} H^T C C^{-1} C H (H^T C H)^{-1} = \\ &= (H^T C H)^{-1} H^T C H (H^T C H)^{-1} = \\ &= (H^T C H)^{-1} = \left(\left[C_{k+1}^{p}\right]^{-1} + H_{k+1} C_{\eta_{k+1}}^{-1} H_{k+1}\right)^{-1}. \end{aligned} \tag{2.34}$$

Again applying the Woodbury matrix identity and recalling definition of the Kalman gain G_{k+1} from (2.18), we can restate (2.34)

$$\begin{aligned} C_{k+1}^{est} &= C_{k+1}^{p} - C_{k+1}^{p} H_{k+1}^{T} \left(H_{k+1} C_{k+1}^{p} H_{k+1}^{T} + C_{\eta_{k+1}}\right)^{-1} H_{k+1} C_{k+1}^{p} \\ &= C_{k+1}^{p} - G_{k+1} H_{k+1} C_{k+1}^{p}, \end{aligned} \tag{2.35}$$

which coincides with formula (2.19) from the previous section. In addition, we should notice (and we will be using this fact intensively in the following chapter) that $H^T C H$ is the Hessian matrix of the weighted least squares cost function (2.7). Hence, C_{k+1}^{est} coincides with the inverse Hessian matrix of the cost function (2.29).
The argumentation provided in this section demonstrated that when the noise terms ϵ_k and η_{k+1} from observed dynamical system (2.9)-(2.10) and the estimate x_k^{est} are normally distributed, the unbiased estimate of the system's state with the minimal variance coincides with the MAP estimate of the state. This means that the algebraic formulae from Algorithm 1 can be replaced by numeric minimization of the cost function (2.29), whereas the covariance matrix of the estimate can be computed by inverting the Hessian matrix of this cost function. Furthermore, the assumption of the noise terms being Gaussian may be relaxed since the algebraic expressions of the MAP estimate and of its covariance matrix coincide with that of the Lemma 3 and the lemma does not require the noise terms to be normally distributed. Hence, the Kalman filter can always be implemented by coupled procedure of minimizing the cost function (2.29) and calculating its inverse Hessian. This fact plays a major role in this thesis, as we will deal with approximations of extended Kalman filter based on certain Quasi-Newton optimization algorithms.

3 Approximative Kalman Filters

This chapter is dedicated to certain family of algorithms that implement approximation of the extended Kalman filter formulae. The approximations considered in this sections are based on Quasi-Newton optimization approaches that calculate optimal point of the cost function by the means of computing approximation for its inverse Hessian. We begin this section by providing brief motivation for the demand to implement the Kalman filter using approximation techniques and describe the cases where the basic solutions provided in the previous chapter cannot be explicitly applied. Then we shortly cover the techniques that are employed to derive the desired approximation and relate them with optimization algorithms. The next part of this chapter introduces the notion of tangent-linear and adjoint models, which are special programmatic subroutines needed for efficient calculation of the gradient data leveraged by many optimization algorithms. Finally, this chapter is wrapped out by a concise description of the weak-constraint 4D-VAR, an alternative optimization approach based on the same Bayesian framework as the Kalman filter, but not completely equivalent to it. We wanted to include a short review of the Weak-Constraint 4D-VAR as it turns out to have numerous similarities to our parallel filtering approach discussed in chapter 4.

3.1 Large-Scale Models

Recall the observed dynamical system model (2.9)-(2.10) and Algorithm 1. As previously we denote dimension of the state space by n and dimension of the observation space by m. Then implementation of Algorithm 1 requires to multiply n-by-n matrices (see step 3), to invert an m-by-m matrix (see step 4) and to store at least one n-by-n matrix. Obviously, implementation of the algorithm becomes impracticable when the dimension of the state space n becomes "large". On the other hand, the operational numerical weather prediction systems that apply data assimilation frameworks to produce the daily forecasts employ dynamics with control equations having as many degrees of freedom as 10^9 (see Cardinali (2013)). Therefore, in order to implement Algorithm 1 for such system one would have to store matrices with about 10^{18} elements. Even if the elements of the matrix are stored in single-precision floating point format this would require at least 3637979TB of RAM storage, which is beyond the capability of any supercomputer existing by the moment of writing this text (Sunway TaihuLight located in national supercomputer center in Wuxi, China was leading the TOP500 list as of June, 2016 and was having only 1310TB of RAM). Hence, straightforward realization of the Kalman filter for such systems is impracticable, which creates demand for alternative approaches to be employed in the large-scale cases.

In the following parts of this chapter we will describe certain approaches to implement the original algebraic formulae from algorithm 1 using Quasi-Newton optimization techniques. These ideas are leveraged as the basis for the main results developed in the present dissertation and discussed in detail in chapter 4.

3.2 Quasi-Newton Optimization and Low-Memory representation of the Hessian Matrices

In this chapter we present several ways to implement the Kalman filter described by algorithm 1 without necessity to explicitly store or multiply the problematic matrices. These approaches are based on Quasi-Newton optimization and are approximate, i.e. they do not calculate the exact optimal estimate equivalent to the outcome of the Kalman filter, but instead approximate it in certain sub-optimal manner introducing a trade-off between accuracy of the solution and the amount of memory used.

In order to construct the desired approximations we need to describe the framework that is used as the starting point for all further derivations. The basic ideas borrow results obtained in optimization theory. We will give a brief snapshot of these results and then relate it to derivation of approximate Kalman filters.

We begin by considering the following auxiliary optimization problem:

$$f(x) = \frac{1}{2}x^T Ax - b^T x + c \to \min \tag{3.1}$$

where $x \in \mathbb{R}^n$, $b \in \mathbb{R}^n$, A is a symmetric n-by-n matrix and c is a real-valued constant. By simple differentiation one can show that $\nabla f(x) = Ax - b$ and therefore when A is invertible the cost function $f(x)$ has only one extremal point $x = A^{-1}b$. Moreover, when A is positive-definite this point minimizes $f(x)$. Hereinafter, we assume that the latter condition holds, i.e. A is positive-definite (which in turn, implies that A is also invertible). As will be demonstrated later in this section in our applications matrix A is usually high-dimensional and therefore we do not want to solve (3.1) explicitly and instead use certain Quasi-Newton minimization scheme. We will mostly consider the approach known as low-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) (see Nocedal and Wright (1999a)). This scheme repeatedly updates approximation of A^{-1} using few preceding steps from the iteration history. The main step of the algorithm is implemented by rank-2 matrix update known as the BFGS update formula, which reads as follows:

$$A_{k+1} = A_k + \frac{yy^T}{y^T s} - \frac{A_k s s^T A_k}{s^T A_k s}. \tag{3.2}$$

Here A_k is an n-by-n symmetric positive-definite matrix, which is the subject for update, whereas s and y is a pair of arbitrary n-dimensional vectors. It is easy to verify that the updated matrix A_{k+1} maps vector s into vector y, i.e. $A_{k+1}s = y$, and A_{k+1} is symmetric and positive-definite. Additionally, in Dennis and Moré (1977) and Dennis and Schnabel (1979) it was demonstrated that assuming certain matrix norm A_{k+1} the BFGS formula is the least change update that maintains symmetric and positive definiteness properties of the original matrix A_k and "injects" the additional information by ensuring that $A_{k+1}s = y$.

Instead of updating certain symmetric positive-definite matrix A_k to map given vector s onto given vector y it is possible to update its inverse A_k^{-1} with requirement that the updated matrix A_{k+1}^{-1} should now map y into s. The required update formula can be

easily obtained by applying the Woodbury matrix inversion lemma to (3.2), which leads to inverse BFGS update (3.3). The inverse update naturally maintains the symmetric and positive-definiteness properties of the original matrix A_k^{-1} and moreover provides the least change update similar to formula (3.2) (see Dennis and Schnabel (1980)).

$$A_{k+1}^{-1} = \frac{ss^T}{y^T s} + \left(I - \frac{sy^T}{y^T s}\right) A_k^{-1} \left(I - \frac{ys^T}{y^T s}\right). \tag{3.3}$$

Since the L-BFGS minimization is employed by the most of the Kalman filter approximations described later in this chapter, we would like to describe it in more detail. The optimization procedure is based on steepest path search approach, i.e. we attempt to minimize (3.1) iteratively with iteration update $x_{k+1} = x_k + \gamma p_k$, where p_k is the search path. It is easy to verify that when the cost function has the form of (3.1), $\gamma = -\frac{p_k^T \nabla f(x_k)}{p_k^T A p_k}$ provides the greatest decrease in the value of $f(x)$ along the search path p_k. The L-BFGS scheme follows the principle leveraged by Quasi-Newton optimizers and employs the search path $p_k = -A_k^{-1} \nabla f(x_k)$, where A_k^{-1} is the k-th approximation of the inverse Hessian A^{-1} of the cost function (3.1). Such selection of the search path is intuitively natural since when the approximation A_k^{-1} is exact, i.e. $A_k^{-1} = A^{-1}$, we get

$$\gamma = -\frac{\left[\nabla f(x_k)\right]^T A^{-1} \nabla f(x_k)}{\left[\nabla f(x_k)\right]^T A^{-1} \nabla f(x_k)} = -1$$
$$x_{k+1} = x_k - A^{-1} \nabla f(x_k) = x_k - A^{-1}(Ax_k - b) = A^{-1} b,$$

which means that the optimal point is reached within one step regardless of the preceding approximation x_k. Hence, it is reasonable expect that by properly approximating A^{-1} the similarly organized iterations would eventually converge towards the optimum.

In L-BFGS approximations A_k^{-1} are based on inverse BFGS update formula (3.3) and on the iteration history. The process begins by selecting certain initial approximation of the inverse Hessian A^{-1} of the cost function (3.1). Usually, this initial guess is selected to be a scaled identity matrix, i.e. $A_0^{-1} = \chi I$. The choice of the scaling factor χ is in general somewhat heuristic, but some general strategies exist and discussed in detail in Nocedal and Wright (1999a). After the first iteration step is computed we get two iteration points x_0 and x_1 (here x_0 is the initial point, from which the iteration process begins) and two corresponding gradients $\nabla f(x_0)$ and $\nabla f(x_1)$. Denoting $s_1 = x_1 - x_0$ and $y_1 = \nabla f(x_1) - \nabla f(x_0)$ one can easily verify that $As_1 = y_1$ or equivalently $A^{-1} y_1 = s_1$. Hence, we want our next approximation of the inverse Hessian A_1^{-1} to map y_1 to s_1, which is exactly the property provided by the inverse BFGS update formula (3.3). Therefore, we plug A_0^{-1}, s_1 and y_1 into (3.3) and obtain A_1^{-1}. We can now use it to compute the next approximation of the optimum x_2 and the corresponding gradient $\nabla f(x_2)$. From this point the iteration process proceeds in the same way as described above until certain termination conditions are satisfied (e.g. the norm of the gradient fails below given threshold value or the difference between two consequent iteration points becomes negligible). The described steps of the L-BFGS scheme are summarized in algorithm 2.

Input : $f(x)$ – quadratic cost function; $\nabla f(x)$ – gradient of the cost function; χ – initial scale of the inverse Hessian; x_0 – initial guess for the minimizing point; ϵ – algorithm termination tolerance value
Output: x_{min} – approximation of the point minimizing the cost function

1 Assign $k \leftarrow 0$;
2 Compute $g_0 \leftarrow \nabla f(x)$;
3 Assign $A_0^{-1} \leftarrow \chi I$;
4 Assign $d \leftarrow -A_0^{-1} g_0$;
5 Assign $\lambda \leftarrow -\frac{d^T g_0}{d^T(\nabla f(d) - \nabla f(\bar{0}))}$;
6 **repeat**
/*Calculate succeeding iteration */
7 $x_{k+1} \leftarrow x_k + \lambda d$;
/*Calculate succeeding gradient */
8 $g_{k+1} \leftarrow \nabla f(x_{k+1})$;
/*Calculate new mapped vector pair */
9 $s_{k+1} = x_{k+1} - x_k$; $y_{k+1} = g_{k+1} - g_k$;
10 Apply (3.3) for s_{k+1}, y_{k+1}, and A_k^{-1} to compute updated inverse Hessian approximation A_{k+1}^{-1};
/*Update search direction and the step scale */
11 $d \leftarrow -A_{k+1}^{-1} g_{k+1}$; $\lambda \leftarrow -\frac{d^T g_{k+1}}{d^T(\nabla f(d) - \nabla f(\bar{0}))}$
/*Update iteration counter */
12 $k \leftarrow k + 1$;
13 **until** $\|g_k - g_{k-1}\| \leq \epsilon$; Stop when difference between subsequent gradients falls below the tolerance value

Algorithm 2: L-BFGS minimization scheme

We should emphasize that step 10 of the algorithm can be implemented without necessity to explicitly store the intermediate matrices in memory, since we only need to compute matrix-vector products, while the recursion in (3.2) and (3.3) can be efficiently unrolled so that calculation would only require knowledge of the vector pairs $(s_1, y_1), \ldots, (s_k, y_k)$, which are generated during the iteration process. Additionally, it is not required to store the full history as formaly required by recursion in (3.2) and (3.3). In practice, the recursion can be reinitialized by dropping the history after few steps. Therefore, algorithm 2 enables possibility to minimize (3.1) without necessity to explicitly store n-by-n matrices. Moreover, the iteration history of the algorithm allows to approximate Hessian matrix A of the cost function (3.1) by the means of the BFGS update formula (3.2). The inverse Hessian A^{-1} can be in turn approximated by leveraging the inverse BFGS update formula (3.3). We want to particularly emphasize that these approximations use only the vector pairs (s_k, y_k) and thus implement the corresponding approximate operator in low-memory fashion that can be used when the dimension of the optimization problem n is restrictedly

large for explicit matrix storage or inversion. Here we do not discuss the details of how the recursion in (2) and (3.3) can be unrolled in practice. The most used approach is called the 2-loop recursion algorithm and is described thoroughly in Nocedal and Wright (1999b). Another less known way introduced by Byrd et al. (1994) allows to implement these recurrent relations in a closed form by explicit algebraic matrix formulae (without necessity to use any large-scale matrices). In our implementation we prefer to use the latter method as it proved to be more stable numerically in our experience.

3.3 Solving Kalman Filtering Equations Using Optimization Methods: L-BFGS

Now as we have introduced the necessary framework we are ready to construct approximation of algorithm 1. The approximation is straightforward and is based on replacement of problematic high-dimensional matrices by corresponding auxiliary optimization problems.

We begin by assuming that the prior covariance matrix C^p_{k+1} introduced in the step 3 of the algorithm can be implemented on the code level as a subroutine that does not employ explicit matrix storage. This assumption is not onerous since it is always possible to provide such implementation when the data assimilation framework provides subroutines implementing operators M, M^T, and C_{ϵ_k}. The subroutines for M and M^T are often supplied along with operational large-scale models like weather prediction systems (refer to section 3.5 for more details) and C_{ϵ_k} is often diagonal or having some other low-memory representation (see for example Fisher (2009) or Dee (1990)). Additionally, we assume that similar code-level implementations are available for observation operator H_{k+1} and its transpose H^T_{k+1}. Hence, we can define auxiliary cost function in the form of (3.1) with $A = H_{k+1}C^p_{k+1}H^T_{k+1} + C_{\eta_k}$, $b = H_{k+1}x^p_{k+1} - y_{k+1}$, and $c = 0$. Note that due to our assumptions both $f(x)$ and $\nabla f(x)$ can be computed without necessity for explicit matrix storage. Hence, we can apply the L-BFGS optimization as outlined in algorithm 2 to minimize this auxiliary cost function. Now presuming that x^* is the approximate minimizing point calculated by the L-BFGS, we can compute the updated estimate $x^{est}_{k+1} = x^p_{k+1} + C^p_{k+1}H^T_{k+1}x^*$. The validity of this statement may be verified directly by considering the theoretical analytical representation of the minimizing point of cost function (3.1) with the values of A, b, and c as selected above and comparing it with the formulae in steps 4 and 5 of algorithm 1. Additionally, the output from the L-BFGS provides not only the minimizing point x^*, but also a low-memory approximate implementation of A^{-1}. This, in turn, allows us to define operator implementing approximation for C^{est}_{k+1} by following the step 6 of algorithm 1. Due to the recurrent nature of the Kalman filter this approximation as such cannot be used in the succeeding step of the filter. Therefore, it is necessary to define yet another auxiliary optimization problem, which when solved by L-BFGS provides low-memory approximation for C^{est}_{k+1} represented by a handful of vector pairs as described before. To make these ideas more clear we outline the described approach in algorithm 3.

Algorithm 3 called LBFGS-EKF was initially introduced by Auvinen et al. (2009a). We should point out then even though above we leverage notation previously employed for

Input : x_k^{est}, M_k, M_k^T, y_{k+1}, H_{k+1}, H_{k+1}^T, $\widehat{C_k^{est}}$, C_{ϵ_k}, $C_{\eta_{k+1}}$
Output: x_{k+1}^{est}, $\widehat{C_{k+1}^{est}}$

foreach *time instance* k **do**
  /*Prediction step */
  $x_{k+1}^p \leftarrow \mathcal{M}_k\left(x_k^{est}\right)$;
  /*Define approximate covariance operator of the predicted state */
  function *PriorCovariance*$(x) \rightarrow C_{k+1}^p x$**:**
    $C_{k+1}^p x \leftarrow M_k \widehat{C_k^{est}} M_k^T x + C_{\epsilon_k} x$;
    return $C_{k+1}^p x$;
  end
  /*Correction step */
  function *AuxiliaryCostFunction1*$(x) \rightarrow y$**:**
    /*Define auxiliary cost function */
    $y \leftarrow H_{k+1}^T x$;
    $y \leftarrow PriorCovariance(y)$;
    $y \leftarrow x^T\left(H_{k+1} y + C_{\eta_{k+1}} x\right) - \left(y_{k+1} - H_{k+1} x_{k+1}^p\right)^T x$;
  end
  /*Subroutine LBFGS below takes a quadratic cost function and yields its minimizing point x_{min} and operator $\widehat{A^{-1}}$ approximating the inverse Hessian */
  $(x_{min}, \widehat{A^{-1}}) \leftarrow$LBFGS (*AuxiliaryCostFunction1*);
  /*Compute corrected estimate */
  $x_{k+1}^{est} \leftarrow x_{k+1}^p + PriorCovariance\left(H_{k+1}^T x_{min}\right)$;
  /*Compute approximate covariance matrix $\hat{C}_{k+1}^{est}$ of the state estimate x_{k+1}^{est} */
  $b \leftarrow$random();
  function *AuxiliaryCostFunction2*$(x) \rightarrow y$**:**
    $aux \leftarrow PriorCovariance(x)$;
    $y \leftarrow H_{k+1} aux$;
    $y \leftarrow \widehat{A^{-1}}(y)$;
    $y \leftarrow H_{k+1}^T y$;
    $y \leftarrow PriorCovariance(y)$;
    $y \leftarrow x^T(aux - y) - b^T x$;
    return y;
  end
  /*For conciseness we assume that LBFGS below yields approximation of the forward Hessian */
  $(\sim, \widehat{C_{k+1}^{est}}) \leftarrow$LBFGS (AuxiliaryCostFunction2);
end

Algorithm 3: Approximate L-BFGS Extended Kalman Filter

the linear Kalman Filter, same approximation obviously works for the EKF when the operator matrices from the linear case are replaced with the corresponding Jacobians as described in chapter 2.
Unfortunately, algorithm 3 has certain stability issues as will be demonstrated later in chapter 4. Reformulation of this algorithm in a way that avoids the stability problems is one of the main theoretical and practical results of the present dissertation.

3.4 Implementing Variational Kalman Filter Using Quasi-Newton Optimization

In this section we describe an alternative approach to implement approximation of the EKF using Quasi-Newton minimization. Recall the variational formulation of the Kalman filter defined by (2.29). In chapter 2 we proved that when the evolution and transition operators are linear, the point minimizing this cost function coincides with the estimate calculated by algorithm 1, whereas the inverse Hessian of (2.29) equals to the covariance matrix of this estimate. As was explained in chapter 2 in non-linear case the operator matrices in (2.29) should be replaced with the corresponding Jacobians, so minimization of this cost function becomes equivalent to EKF iteration. We can now leverage the framework provided by the L-BFGS introduced earlier in this chapter in order to perform minimization of (2.29) and to approximate its inverse Hessian. The caveat is however that implementation of (2.29) requires knowledge of inverse prior covariance $\left[C^p_{k+1}\right]^{-1}$, which should be computed implicitly in a low-memory manner. This can be done by introducing another auxiliary quadratic optimization task with the Hessian matrix equal to C^p_{k+1} and then minimizing it using the L-BFGS. The iteration history from this auxiliary minimization is then plugged into the inverse BFGS update formula (3.3), which provides low-memory approximation of the desired matrix $\left[C^p_{k+1}\right]^{-1}$. More rigorously, the method is summarized in algorithm 4.
Note that in algorithm 4 we apply L-BFGS to the first auxiliary cost function in order to obtain approximation for the inverse prior covariance matrix C^p_{k+1} and we are not interested in the minimizing point of this cost function per se. In turn, in the second auxiliary optimization task we leverage the L-BFGS to get a low-memory approximation of the forward covariance matrix C^{est}_{k+1} of the estimate x^{est}_{k+1}. In chapter 2 it was shown that matrix C^{est}_{k+1} equals to the inverse Hessian of the second auxiliary cost function, hence in practice the L-BFGS is again employed to approximate matrix inversion. We should point out that the algorithm uses only matrix-vector products, so by utilizing the inverse BFGS update formula (3.3) it is possible to avoid explicit storage of potentially high dimensional matrices.
Algorithm 4 initially introduced in Auvinen et al. (2009b) is called variational Kalman filter (VKF) . In Auvinen et al. (2009b) the concept of this approach was experimentally verified for several toy-case models. However, the only high-dimensional case studied in this work was not chaotic and therefore the algorithm's potential for operational use was not completely justified. As part of the present work we implemented algorithm 4 for the case of two-layer quasi-geostrophic model (introduced in detail in section 4.3). However,

Input : x_k^{est}, M_k, M_k^T, y_{k+1}, H_{k+1}, H_{k+1}^T, $\widehat{C_k^{est}}$, C_{ϵ_k}, $C_{\eta_{k+1}}$
Output: x_{k+1}^{est}, $\widehat{C_{k+1}^{est}}$

foreach *time instance* k **do**
  /*Prediction step */
  $x_{k+1}^p \leftarrow \mathcal{M}_k x_k^{est}$;
  /*Define auxiliary optimization problem */
  $b \leftarrow$ random();
  function *AuxiliaryCostFunction1*$(x) \rightarrow y$**:**
    $aux \leftarrow M_k^T x$;
    $aux \leftarrow \widehat{C_{k+1}^{est}} aux$;
    $aux \leftarrow M_k aux + C_{\epsilon_k} - b^T x$;
    return aux;
  end
  /*Use L-BFGS in order to retrieve low-memory approximation of $\left[C_{k+1}^p\right]^{-1}$ */
  $\left(\sim, \widehat{\left[C_{k+1}^p\right]^{-1}}\right) \leftarrow$ LBFGS(*AuxiliaryCostFunction1*);
  /*Correction step */
  /*Define the variational Kalman filter cost function following (2.29) */
  function *AuxiliaryCostFunction2*$(x) \rightarrow y$**:**
    $y \leftarrow \bar{0}$;
    $e \leftarrow x - x_{k+1}^p$;
    $aux \leftarrow \widehat{\left[C_{k+1}^p\right]^{-1}} e$;
    $y \leftarrow y + e^T aux$;
    $d \leftarrow y_{k+1} - H_{k+1} x$;
    $aux \leftarrow C_{\eta_{k+1}}^{-1} aux$;
    $y \leftarrow y + d^T aux$;
    return y;
  end
  /*Use L-BFGS to compute corrected estimate and its covariance matrix */
  $\left(x_{k+1}^{est}, \widehat{C_{k+1}^{est}}\right) \leftarrow$ LBFGS(*AuxiliaryCostFunction2*);
end

Algorithm 4: Variational Kalman Filter

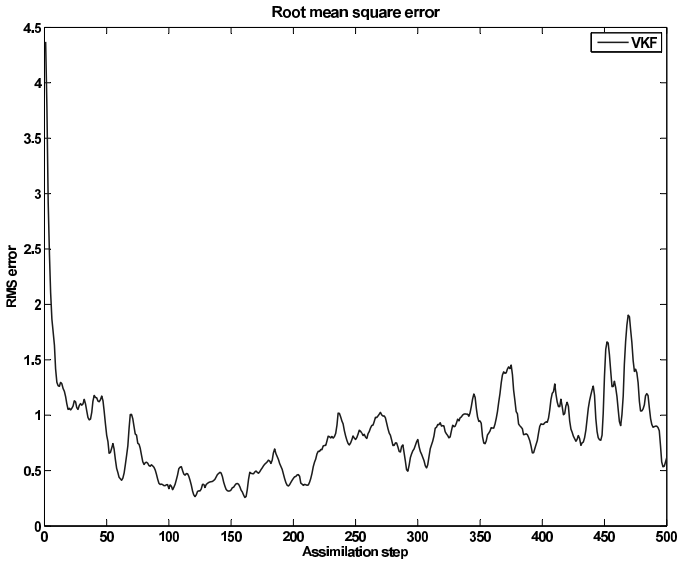

Figure 3.1: Example root mean square error of VKF estimates computed for the case of two-layer quasi-geostrophic model with 100 observations and 1600 degrees of freedom

the quality of the estimates calculated using this approach was poor and convergence of the filter exhibited signs of instability. Figure 3.1 provides an example root mean square error of the estimates computed by the filter. From the figure one can easily observe semi-stable behaviour of the filter. Further analysis has shown that this issue is caused by potentially "dangerous" operation of matrix inversion in (2.29). The inverse of the model error covariance matrix $C_{\epsilon_k}^{-1}$ can usually be computed with good accuracy, however this does not in general apply to $\left[C_{k+1}^{p}\right]^{-1}$. In figure 3.2 we demonstrate distribution of the eigenvalues of the prior covariance matrix computed by the EKF. From the figure it is easy to see that the matrix is ill-conditioned and therefore BFGS inversion cannot be done efficiently (see Broyden (1970)).

3.5 Tangent-Linear and Adjoint Models

We now want to take a closer look at implementing the Kalman filter for the cases, where evolution model (2.9) and observation model (2.10) are non-linear. In section 2.3 we introduced EKF derived as extension of the algorithm 1 to allow usage of non-linear models. In EKF it was suggested to replace the operator matrices by the corresponding Jacobians (2.21) everywhere except the very first prediction step of the original algorithm. In cases where dimensions of the state and observation spaces are relatively small it is feasible to calculate (2.21) explicitly, for example by finite differences. However, in large-scale cases such implementation is not practicable. In addition, as was pointed out in the beginning of

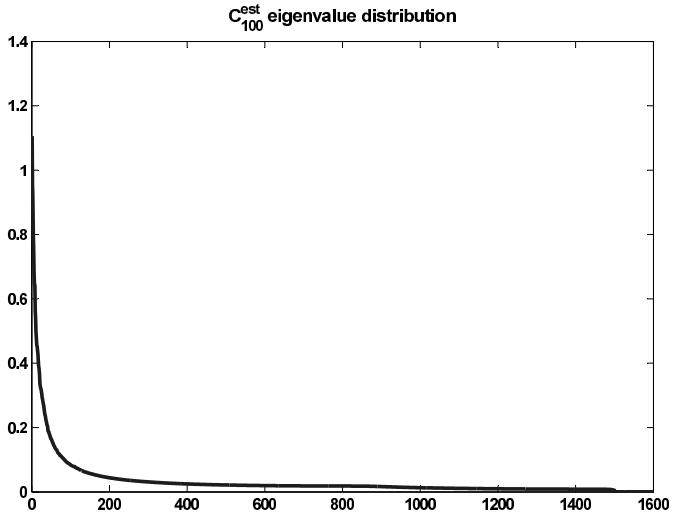

Figure 3.2: Distribution of eigenvalues of C^p_{100} compute by EKF for the case of two-layer quasi-geostrophic model with 100 observations and 1600 degrees of freedom

this chapter, when the state space dimension is very large, it is not even possible to store the Jacobian matrices explicitly. On the other hand, in order to be able to implement either the LBFGS-EKF (see algorithm 3) or the VKF (see algorithm 4) it would be enough to implement subroutines computing matrix-vector products between Jacobian matrix, transpose Jacobian matrix and an arbitrary vector of conformable size. In other words, Jacobian matrix and its transpose both of the evolution and of the observation operators have to be implemented as subroutines on the code level without involvement of explicit matrix storage. Such implementations of Jacobian matrices are called tangent-linear (TL-) codes (models). The implementations of transpose Jacobian matrices are called adjoint (AD-) codes (models). Sometimes the TL- and AD- codes are referred to as the gradient codes or gradient operators (to emphasize the role they play in implementations of filtering methods based on optimization). Hereinafter, assuming that $\mathcal{M}$ is certain operator we will interchangeably use notation M^{TL} and M to denote tangent-linear code or Jacobian of this operator and M^{AD} or M^T to denote its adjoint code or transposed Jacobian.

It is obvious that practical realization of TL- and AD- codes depends on the underlying non-linear dynamics and observation models. Theoretically, for each non-linear model (evolution or observation or both) one has to compute differential operator and its adjoint and then carefully analyse them to come up with an efficient implementation. However, few simple rules can be developed to tremendously automate this process. These rules are based on trivial observation that the non-linear model to be differentiated is in practice just a long series of program instructions, whereas each instruction can be considered as a tiny operator, which can be differentiated and for which it is possible to derive the adjoint. Then the full program is simply superposition of all of its instruction operators,

so the chain rule (see Marsden and Weinstein (1985)) can be applied to synthesize the corresponding TL- and AD- models. We illustrate this idea on a handy example of the Lorenz-3 model.
Consider the following system of three ODEs:

$$\begin{aligned}\frac{dx}{dt} &= \sigma(y-x),\\ \frac{dy}{dt} &= x(\rho-z)-y,\\ \frac{dz}{dt} &= xy-\beta z.\end{aligned} \tag{3.4}$$

Assuming that (x_k, y_k, z_k) is solution of (3.4) at time instance k we construct Euler approximation of solution at time instance $k+1$. Hence, we arrive at the following discrete-time non-linear evolution model:

$$\begin{aligned}x_{k+1} &= \sigma\left(y_k - x_k\right)\Delta t + x_k,\\ y_{k+1} &= x_k\left(\rho - z_k\right)\Delta t + \left(1-\Delta t\right)y_k,\\ z_{k+1} &= x_k y_k \Delta t + \left(1-\beta\Delta t\right)z_k,\end{aligned} \tag{3.5}$$

where Δt is the step of time discretization. Next we rewrite (3.5) by series of programmatic instructions and differentiate each instruction independently.

```
x1=sigma*(y0-x0)*dt+x0 ;//non-linear statement
dx1=sigma*(dy0-dx0)*dt+dx0 ;//TL-code

y1=x0*(rho-z0)*dt+(1-dt)*y0 ;//non-linear statement
dy1=dx0*(rho-z0)*dt+(1-dt)*dy0-dz0*x0*dt ;//TL-code

z1=x0*y0*dt+(1-beta*dt)*z0 ;//non-linear statement
dz1=y0*dt*dx0+x0*dt*dy0+(1-beta*dt)*dz0 ;//TL-code
```

As can be observed from the code listing each statement of the TL-code is obtained by simple differentiation of the corresponding statement of the non-linear model. In more complicated cases one will also have to apply the chain rule, but in this simple example differentiation boils down to trivial statement-by-statement calculation of derivatives. The resulting TL-model takes variables `dx0`, `dy0`, `dz0` and returns variables `dx1`, `dy1`, `dz1`. Variables `x0`, `y0`, `z0` determine the point, at which differentiation happens. These variables are called trajectory variables (or just trajectory) of the linearized model. For clarity we define the TL-model of (3.5) using more convenient math notation:

$$\begin{aligned}\partial x_{k+1} &= \sigma\left(\partial y_k - \partial x_k\right)\Delta t + \partial x_k\\ \partial y_{k+1} &= \left(\rho - z_k\right)\Delta t\partial x_k + \left(1-\Delta t\right)\partial y_k - x_k\Delta t\partial z_k\\ \partial z_{k+1} &= y_k\Delta t\partial x_k + x_k\Delta t\partial y_k + \left(1-\beta\Delta t\right)\partial z_k.\end{aligned} \tag{3.6}$$

Our next task is to derive adjoint model for system (3.5). To make this problem a bit

easier we first consider the relation for ∂z_{k+1} individually and derive the adjoint model just for the following single statement:

$$\partial z_{k+1} = y_k \Delta t \partial x_k + x_k \Delta t \partial y_k + (1 - \beta \Delta t)\, \partial z_k. \tag{3.7}$$

This statement can be rewritten in equivalent matrix form (3.8). Notice that we consider all the TL- input and output variables operated by the statement as the mapped values of the corresponding matrix operator.

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \Delta t y_k & \Delta t x_k & 1 - \beta \Delta t & 0 \end{pmatrix} \begin{pmatrix} \partial x_k \\ \partial y_k \\ \partial z_k \\ \partial z_{k+1} \end{pmatrix} \rightarrow \begin{pmatrix} \partial x_k \\ \partial y_k \\ \partial z_k \\ \partial z_{k+1} \end{pmatrix} \tag{3.8}$$

Now, the adjoint code for (3.8) can be deduced straightforwardly:

$$\begin{pmatrix} 1 & 0 & 0 & \Delta t y_k \\ 0 & 1 & 0 & \Delta t x_k \\ 0 & 0 & 1 & 1 - \beta \Delta t \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \partial x_k^* \\ \partial y_k^* \\ \partial z_k^* \\ \partial z_{k+1}^* \end{pmatrix} \rightarrow \begin{pmatrix} \partial x_k^* \\ \partial y_k^* \\ \partial z_k^* \\ \partial z_{k+1}^* \end{pmatrix}, \tag{3.9}$$

where $\partial \cdot^*$ denotes adjoint variables (i.e. the input and output variables of the adjoint model). Restating (3.9) we obtain

$$\begin{aligned} \partial x_k^* + \Delta t y_k \partial z_{k+1}^* &\rightarrow \partial x_k^* \\ \partial y_k^* + \Delta t x_k \partial z_{k+1}^* &\rightarrow \partial y_k^* \\ \partial z_k^* + (1 - \beta \Delta t)\, \partial z_{k+1}^* &\rightarrow \partial z_k^* \\ 0 &\rightarrow \partial z_{k+1}^* \end{aligned} \tag{3.10}$$

Analysing (3.10) we can formulate the following rules for derivation of adjoint codes for separate statements from a TL-model:

1. Each TL-statement generates as many adjoint statements as the number of its TL-inputs and TL-outputs combined.
2. For each term of the statement of form $y = \alpha \partial x$, where ∂x is certain TL-input variable and y is a TL-output variable generate adjoint statement $\partial x^* = \partial x^* + \alpha \partial y^*$.
3. For output variable ∂y of the considered TL-statement generate adjoint statement $\partial y^* = 0$.
4. All dummy inputs ∂x^* of the synthesized adjoint code should be initialized by 0.
5. Set variables ∂y^* to be the inputs of the code and return the resulting accumulated values of variables ∂x^* as the code's output.
6. The adjoint statements are generated from the corresponding TL-statements in reversed order, i.e. the adjoint of the last statement of the TL-code appears the first

in the AD-code and the adjoint of first statement of the TL-code appears the last in the AD-code.

We should pay particular attention to the 3rd rule: seemingly useless as it just assigns zero to an extra adjoint output it nevertheless plays an important role when combined with the other adjoint statements generated from TL-program. The 4th rule is needed to compensate inclusion of the dummy input variables like ∂x_k, ∂y_k, and ∂z_k from statement (3.7). There is also another way to understand this rule. Specifically, assuming that $(\partial x_1, \ldots, \partial x_k)$ are the input variables to the very last statement in the TL- source code and ∂y is the corresponding output variable and presuming for simplicity that ∂y is the only output variable of the TL-code we can suppose that at the very last "implicit" instruction of the code all the inputs $(\partial x_1, \ldots, \partial x_k)$ were just assigned to zero (indeed it must be zero for the code to be linear!) and then the output is returned. However, the adjoint instruction for "assignment to zero" is just "assignment to zero", which justifies the 4th rule of our adjoint derivation algorithm.
The formulated rules allow us to immediately produce the code for the full adjoint model of (3.5) (notice the inverse order of operations and how the adjoint outputs of instructions that appear later in the TL-code are plugged into the corresponding adjoint inputs of the instructions that precede them):

```
//Adjoint for dz
dx0_adj = 0;
dy0_adj = 0;
dz0_adj = 0;
aux_dx0=dx0_adj+y0*dt*dz1_adj;
aux_dy0=dy0_adj+x0*dt*dz1_adj;
aux_dz0=dz0_adj+(1-beta*dt)*dz1_adj;
aux_dz1=0;

//Adjoint for dy
aux_dx0=aux_dx0+(rho-z0)*dt*dy1_adj;
aux_dy0=aux_dy0+(1-dt)*dy1_adj;
aux_dz0=aux_dz0-x0*dt*dy1_adj;
aux_dy1=0;

//Adjoint for dx
dx0=aux_dx0+(1-sigma*dt)*dx1_adj;
dy0=aux_dy0+sigma*dt*dx1_adj;
dz0=aux_dz0;
aux_dx1=0;
```

The code takes `dx1_adj`, `dy1_adj`, and `dz1_adj` and returns the values of `dx0`, `dy0`, and `dz0` as the output. Note variables `aux_dx1`, `aux_dy1`, and `aux_dz1` that are assigned to zero and ignored thereon. In general case these variables must be kept in the code as the consequent instructions may use them, however in the considered simple example they just happen to be an artifact.

In considered Lorenz-3 case we did not cover the programmatic control structures like branches and loops that naturally appear in implementations of more complex models. However, it turns out that dealing with them is not much more complicated then assembling the adjoint code from individual adjoint instructions demonstrated so far. Indeed, a loop can be unrolled to simple sequence of instructions, so the adjoint loop is just a loop that acts in reversed direction with the body of the original loop replaced by the corresponding adjoint code. Branches are slightly more complicated as one has to remember exactly, which concrete path of the branch has been executed by the underlying non-linear model and include this information into trajectory data. With this information available the bodies of the branches in TL- and AD- models are implemented similar to the normal series of instructions.
The procedure of synthesizing such codes is obviously rather routine and therefore is a subject for automation. Naturally, numerous attempts have been made to generate adjoint models given a non-linear operator, a TL-operator obtained by automated differentiation or the corrseponding source codes (see for example Farrel et al. (2013) or Heimbach et al. (2002)). Unfortunately, as can be easily spotted even from our simple example the code produced by such automations is far from being optimal. Certainly, the approach that we demonstrated is fairly simple and lacks various sophisticated syntax analysis techniques (e.g. tracking of unused variables or redundant instructions) that could be implemented to achieve more optimal resulting code. However, optimization of the adjoint codes produced by automated generation systems is the topic of ongoing research and implementation of advanced adjoint operators used in demanding systems like numerical weather prediction models still requires participation of a human developer.

3.6 Weak-Constraint 4D-VAR

We wrap out this chapter by introducing the Weak-Constraint 4D-VAR, yet another implementation of statistical filtering for large-scale dynamics. This approach has many similarities with the parallel filter, which is introduced in chapter 4. The parallel filter in turn, is one of the main results of the present work, hence it is reasonable to take a closer look at the competing approach that is based on similar ideas. In this chapter we will derive the Weak-Constraint 4D-VAR cost function and shortly discuss how the related optimization routine can be practically implemented.
As was demonstrated in section 2.4, the Bayes theorem (2.22) is a good starting point for derivation of the Kalman Filter. We will present the Weak-Constraint 4D-VAR using the same framework.
Unlike the data assimilation methods discussed so far, the Weak-Constraint 4D-VAR exploits concept of assimilation window that is composed of several sub-windows. Each sub-window corresponds to a time period carrying new observation data to be processed. Hence, the algorithm operates over multiple packs of observation data and computes the corresponding state estimates within single iteration (figure 3.3 provides schematic illustration of the process). The idea comes close to that of the Kalman smoothing (Anderson and Moore (1979)) and it turns out that both approaches are in some sense equivalent (see Fisher et al. (2005)). Below we give the main concepts that lead to formulation of

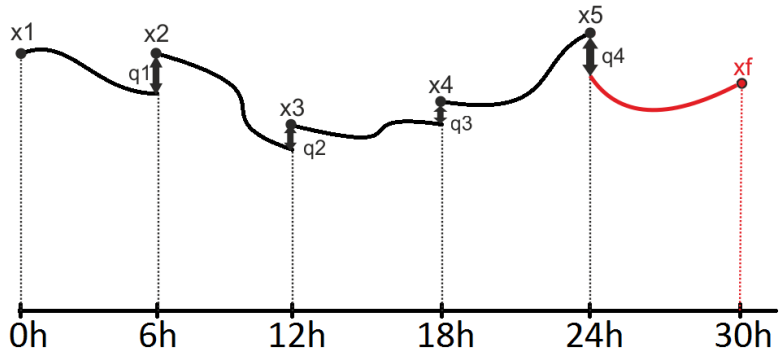

Figure 3.3: Example Weak-Constraint 4D-VAR setting. Here x_i denote the states and terms q_i are forecast errors in the end of each sub-window; xf is the forecast calculated from the last state estimate. It is assumed that observations are provided for each sub-window.

the Weak-Constraint 4D-VAR algorithm. More exhaustive discussion of this topic can be found in Zupanski (1996) or Apte et al. (2008).

We begin by considering assimilation window composed from N states $x_i, \ldots, x_{i+N-1}$ of certain dynamical system. We also consider the sequence of corresponding observation vectors $y_i, \ldots, y_{i+N-1}$. For the reasons of conciseness, let us introduce the following matrix notation:

$$X = \begin{pmatrix} x_i & x_{i+1} & \ldots & x_{i+N-1} \end{pmatrix},$$
$$Y = \begin{pmatrix} y_i & y_{i+1} & \ldots & y_{i+N-1} \end{pmatrix}.$$

Leveraging the Bayes theorem (2.22) we obtain the following product of the likelihood and prior, which should be maximized with respect to X:

$$p\left(X|Y\right) \propto p\left(Y|X\right) p\left(X\right).$$

To proceed we need to make several additional assumptions. First, we assume that an arbitrary j–th observation relies only upon the corresponding j–th state and that the measurement error is independent between observations acquired at different time instances. Therefore, since y_j depends only on x_j, the likelihood $p\left(Y|X\right)$ can be restated as follows:

$$p\left(Y|X\right) = \prod_{j=i}^{i+N-1} p(y_j|x_j). \tag{3.11}$$

The second assumption we make is that an arbitrary state x_j depends only on its immediate ancestor x_{j-1}, i.e. the series of state vectors form a Markov chain. Therefore, the prior PDF $p\left(X\right)$ can be represented by product $p\left(x_i\right) \prod_{j=i+1}^{i+N-1} p\left(x_j|x_{j-1}\right)$. Collecting the

equations, we obtain the following relation:

$$p\left(X|Y\right) \propto p\left(x_i\right) \prod_{j=i}^{i+N-1} p(y_j|x_j) \prod_{j=i+1}^{i+N-1} p\left(x_j|x_{j-1}\right).$$

It remains to determine the density functions $p(x_i)$, $p(y_j|x_j)$ and $p(x_j|x_{j-1})$. For the first state vector x_i we assume a Gaussian prior with mean x_b^i and covariance matrix B_i. In data assimilation literature, x_b^i is often referred to as the background state, whereas B_i is called background covariance matrix. In practice, x_b^i is often picked to be the forecast calculated using the most recent state estimate. The likelihoods $p(y_j|x_j)$ are assumed to be Gaussian, with error covariance matrices R_j. The densities $p(x_j|x_{j-1})$ can be determined by noticing that from (2.9) the model errors can be described as $\epsilon_j = x_{j+1} - M_j(x_j)$ and assuming they are Gaussian with mean $\bar{\epsilon}_j$ and covariance matrix Q_j. Substituting these expressions into posterior $p(X|Y)$ and taking the negative logarithm of the result we get the Weak-Constraint 4D-VAR least squares problem:

$$\begin{aligned}\mathcal{L}\left(X\right) =& \textbf{const} + \frac{1}{2}\left(x_b^i - x_i\right)^{\mathrm{T}} B_i^{-1}\left(x_b^i - x_i\right) + \\ &+ \frac{1}{2}\sum_{j=i}^{i+N-1}\left(y_j - \mathcal{H}_j(x_j)\right)^{\mathrm{T}} R_j^{-1}\left(y_j - \mathcal{H}_{|}(x_j)\right) + \\ &+ \frac{1}{2}\sum_{j=i}^{i+N-2}\left(\epsilon_j - \bar{\epsilon}_j\right)^{\mathrm{T}} Q_j^{-1}\left(\epsilon_j - \bar{\epsilon}_j\right) \rightarrow \textbf{min}, \end{aligned} \tag{3.12}$$

Minimization of function (3.12) brings us to so-called 4D-state formulation of the Weak-Constraint 4D-VAR. Other formulations, each having its advantages and disadvantages, are possible (various choices of control variable leading to different formulations of the cost function are discussed in Trémolet (2006)). We should note that operators $\mathcal{H}_j$ and $\mathcal{M}_j$ can be non-linear and so is the cost function $\mathcal{L}\left(X\right)$, which makes it non-trivial to minimize. One possibility to deal with this problem is provided by incremental minimization scheme suggested in Trémolet (2007). Below we briefly present this approach.

Assume that $X^{(n)}$ is the intermediate minimizer obtained on the n–th iteration while minimizing the cost function $\mathcal{L}\left(X\right)$. We would like to find an increment, which when added to $X^{(n)}$ would bring us to the optimum X, i.e. we look for $\delta X^{(n)} = X - X^{(n)}$. By using the Taylor expansions we get

$$\begin{aligned}\mathcal{H}_{j+1}(x_{j+1}) =& \mathcal{H}_{j+1}\left(x_{j+1}^{(n)}\right) + \\ &+ H_{j+1}^{TL}\delta x_{j+1}^{(n)} + O\left(\left\|\delta x_{j+1}^{(n)}\right\|^2\right), \end{aligned} \tag{3.13}$$

$$\begin{aligned}\mathcal{M}_j(x_j) =& \mathcal{M}_j\left(x_j^{(n)}\right) + \\ &+ M_j^{TL}\delta x_j^{(n)} + O\left(\left\|\delta x_j^{(n)}\right\|^2\right),\end{aligned} \tag{3.14}$$

where H_{j+1}^{TL} and M_j^{TL} are the tangent–linear codes of the observation and evolution operators. Moreover, from (3.14) we see that

$$\begin{aligned}\epsilon_j =& x_{j+1} - \mathcal{M}_j(x_j) \approx \\ \approx& x_{j+1} - x_{j+1}^{(n)} + x_{j+1}^{(n)} - \mathcal{M}_k\left(x_j^{(n)}\right) - M_j^{TL}\delta x_k^{(n)} = \\ =& \delta x_{j+1}^{(n)} - M_j^{TL}\delta x_j^{(n)} + \epsilon_j^{(n)},\end{aligned} \tag{3.15}$$

where $\epsilon_j^{(n)} = x_{j+1}^n - \mathcal{M}_j(x_j^n)$. To simplify the further discussion we introduce some auxiliary notation:

$$\begin{aligned}b^{(n)} &= x_b - x_i^{(n)}, \\ d_j^{(n)} &= y_j - \mathcal{H}_j\left(x_j^{(n)}\right), \\ \bar{\epsilon}_j^{(n)} &= \bar{\epsilon}_j - \epsilon_j^{(n)},\end{aligned}$$

where the terms $d_j^{(n)}$ are sometimes called observation departures.

The incremental algorithm is based on the assumption that the target increment $\delta X^{(n)}$ could be efficiently approximated by the point minimizing the following linearized cost function:

$$\begin{aligned}&L\left(\delta X^{(n)}\right) = \frac{1}{2}\left(b^{(n)} - \delta x_i^{(n)}\right)^{\mathrm{T}} B_i^{-1}\left(b^{(n)} - \delta x_i^{(n)}\right) + \\ &+ \frac{1}{2}\sum_{j=i}^{i+N-1}\left(d_j^{(n)} - H_j^{(n)}\delta x_j^{(n)}\right)^{\mathrm{T}} R_j^{-1}\left(d_j^{(n)} - H_j^{(n)}\delta x_j^{(n)}\right) + \\ &+ \frac{1}{2}\sum_{j=i}^{i+N-2}\left(\delta x_{j+1}^{(n)} - M_j^{TL}\delta x_j^{(n)} - \bar{\epsilon}_j^{(n)}\right)^{\mathrm{T}} Q_j^{-1}\times \\ &\left(\delta x_{j+1}^{(n)} - M_j^{TL}\delta x_j^{(n)} - \bar{\epsilon}_j^{(n)}\right),\end{aligned} \tag{3.16}$$

Obviously, the linearized cost function (3.16) is a quadratic function. Therefore, it can be efficiently minimized for instance by application of the Conjugate–Gradient optimization algorithm (CG algorithm) (see Magnus and Stiefel (1952)). Minimization of (3.16) is sometimes called "the inner loop". Assuming, that $\delta X^{(n)}$ is the point minimizing (3.16), we can define the next approximation for the optimum of the non-linear cost function (3.12) as $X^{(n+1)} = X^{(n)} + \delta X^{(n)}$. This iteration procedure is sometimes called "the outer loop". Essentially, it is easy to verify that this method is similar to the well-known Gauss-Newton scheme.

Summarizing the ideas described above, we can construct algorithmic layout for data assimilation performed with the Incremental Weak-Constraint 4D-VAR in the 4D-state formulation (see algorithm 5). Here we consider only this formulation; other approaches are discussed in detail in Trémolet (2006). Algorithm 5 provides only schematic outline of the method. In practice cost function (3.16) turns out to be ill-conditioned and therefore requires usage of preconditioning techniques. Here we do not discuss these techniques as they lie beyond the main scope of this work. A very detailed report on various formulations and implementation details of the Weak-Constraint 4D-VAR may be found in Fisher et al. (2011).

Input : $\mathcal{M}_i$ – prediction operator, $\mathcal{H}_i$ – observation operator, M_i^{TL}, M_i^{AD}, H_i^{TL}, H_i^{AD} – the corresponding tangent linear and adjoint models; I_{outer} – number of iterations of the outer loop; N – number of sub-windows in the data assimilation window; x_b – background state estimate; $\left(x_i^{est}, \ldots, x_{i+N-1}^{est}\right)$ – estimates from the last assimilation step; $(y_{i+1}, \ldots, y_{i+N})$ – observations corresponding to the states to be estimated, B_i – background covariance matrix, $(R_{i+1}, \ldots, R_{i+N})$ – observation covariance matrices, $(Q_{i+2}, \ldots, Q_{i+N})$ – model error covariance matrices

Output: $\left(x_{i+1}^{est}, \ldots, x_{i+N}^{est}\right)$ – estimates of the requested states

foreach *time instance* i **do**
  /*Prediction step */
  $x_{k+N}^p = \mathcal{M}_{k+N-1}\left(x_{k+N-1}\right)$;
  if mod(*i*,*N*) == *0* **then**
    /*When data assimilation window makes full transition update background estimate */
    $x_b \leftarrow x_i^{est}$;
  end
  /*Correction step */
  $k \leftarrow 0$;
  $X \leftarrow \left(x_{i+1}, \ldots, x_{i+N-1}, x_{i+N}^p\right)$;
  repeat
    /*Minimize quadratic function $L\left(\delta X^{(n)}\right)$ defined by (3.16) using conjugate-gradient optimization. The initial guess for minimization should be all-zero. Below we also assume that CG is a subroutine implementing conjugate-gradient minimization technique, which takes an initial guess for the optimum and a cost function to minimize and returns the minimizing point */
    $\Delta X \leftarrow \bar{0}$;
    $\Delta X \leftarrow$CG(ΔX, L);
    /*Update the guess for the states */
    $X = X + \Delta X$;
    $k \leftarrow k + 1$;
  until $k < I_{outer}$;
  $\left(x_{i+1}^{est}, \ldots, x_{i+N}^{est}\right) \leftarrow X$;
end

Algorithm 5: Weak-Constraint 4D-VAR

4 Stabilizing Correction and Parallel Filter

This section contains the main findings of the present dissertation. The detailed descriptions are provided in papers in Bibov et al. (2015) and in Bibov and Haario (2016). This chapter however covers the main ideas and motivations of the results presented in the papers.

We begin by once again reviewing the LBFGS approximation of the EKF and demonstrate the issues of this formulation that can arise when the approximation is not accurate enough, which is significant vulnerability in large-scale cases. After discussing this caveat of the previously known formulation we introduce our modification to the algorithm called stabilizing correction. We prove that this correction addresses the problems of the original approach and moreover establishes a higher rate of convergence towards the exact EKF formulae. In the next part of this chapter we develop our approach further by applying it to so-called parallel filtering task, which in the essence is just normal data assimilation problem formulated for few consequent stated combined together. Finally, the last part of the chapter introduces the toy-case model that we used to asses performance of our methods. Here we briefly describe the model and shortly discuss the numerical integration scheme that we have implemented for our experiments. In addition we emphasize the most important parameters of the model and the values of theirs employed in our experimental setups.

4.1 Instability Problem and Stabilizing Correction

Let us recall the LBFGS-EKF approximation of the Kalman filter formulated in algorithm 3. In chapter 3 we mentioned that the superior property of this algorithm when compared to the VKF is that it does not require inversion of the prior covariance matrix C^p_{k+1}. However, as we demonstrate below, the direct approximation of the basic Kalman filtering formulae as suggested in the LBFGS-EKF still leads to considerable stability issues.

In order to make relation of the arguments that follow to the terms used in algorithm 3 more simple, we will retain below the notation leveraged in the case of linear model. However, all our conclusions hold correct in exactly same form with only exception of the operator matrices that are subject for replacement by the corresponding tangent-linear and adjoint codes.

We start our analysis of algorithm 3 by considering the first auxiliary optimization problem defined in step 12. The results yielded by the LBFGS optimization subroutine are coupled auxiliary vector, which is then employed to compute the succeeding estimate in step 13, and a low-memory approximation of matrix A^{-1}, which, in turn, was defined as follows:

$$A^{-1} = \left(H_{k+1} C^p_{k+1} H^T_{k+1} + C_{\eta_{k+1}}\right)^{-1}. \tag{4.1}$$

This matrix is subsequently plugged into the second auxiliary optimization task introduced in the steps 15–23 of the algorithm. The Hessian matrix of the cost function minimized during this second optimization is defined as follows:

$$C^p_{k+1} - C^p_{k+1} H^T_{k+1} A^{-1} H_{k+1} C^p_{k+1} \approx C^{est}_{k+1}. \tag{4.2}$$

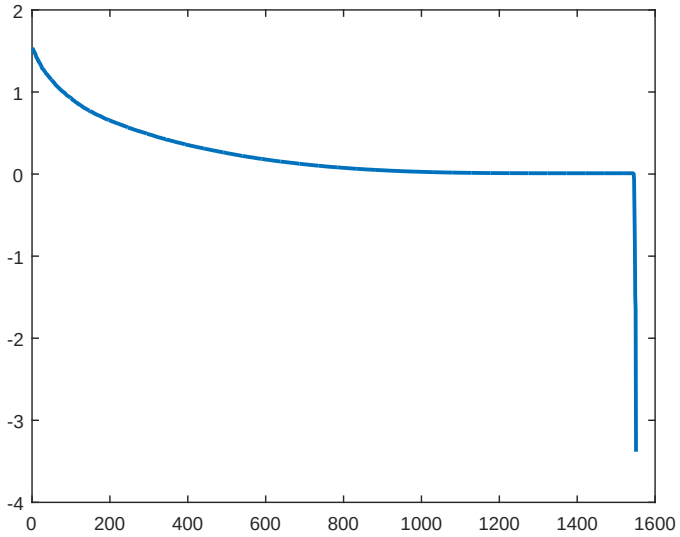

Figure 4.1: Eigenvalues of estimate covariance matrix computed by LBFGS-EKF

In (4.2) the matrix, which is the subject for succeeding low-memory approximation is expected to be non-negative definite as implied by the derivation of the Kalman filter (see chapter 2 for details). However, in algorithm 3 this matrix itself uses low-memory approximation of A^{-1}, which violates assumptions made during derivation of the original formulae of the Kalman filter. In addition, the L-BFGS optimization assumes that the target cost function is convex, whereas the approximate matrix from (4.2) can be indefinite. This implies that approximation of the estimate covariance matrix $\widehat{C_{k+1}^{est}}$ calculated in the last step of algorithm 3 is inadequate, which in certain cases could lead to divergence of the filter (see Bibov et al. (2015)). The problem is illustrated in Figure 4.1. The figure provides an example of eigenvalue distribution of an estimate covariance matrix computed by LBFGS-EKF. This example was generated using two-layer quasi geostrophic model (described in detail in section 4.3) with parameters of the model and of the data assimilation similar to those described in the paper by Bibov et al. (2015). From the figure one can see that the last few eigenvalues of the matrix clearly drop below zero, which immediately leads to inconsistent results produced by the algorithm.

As one of the main results of the present dissertation we suggest a simple correction for algorithm 3, which allows to alleviate the described stability problem and ensure sbetter convergence of the filter.

Our stabilizing correction is inspired by derivation of the linear Kalman filter as provided in chapter 2 and is based on a slightly different formulation of (4.2). Indeed, during derivation of lemma 3 we have shown that C_{k+1}^{est} can be represented as follows:

$$C_{k+1}^{est} = (I - G_{k+1}H_{k+1})\, C_{k+1}^{p}\, (I - G_{k+1}H_{k+1})^{T} + G_{k+1}C_{\eta_{k+1}}G_{k+1}^{T}, \qquad (4.3)$$

where $G_{k+1} = C^p_{k+1} H^T_{k+1} A^{-1}$ is the Kalman gain. When this value of G_{k+1} is plugged into (4.3) we get the following:

$$\begin{aligned} C^{est}_{k+1} =& C^p_{k+1} - G_{k+1} H_{k+1} C^p_{k+1} - C^p_{k+1} H^T_{k+1} G^T_{k+1} + \\ & G_{k+1} H_{k+1} C^p_{k+1} H^T_{k+1} G^T_{k+1} + G_{k+1} C_{\eta_{k+1}} G^T_{k+1} = \\ & C^p_{k+1} - 2C^p_{k+1} H^T_{k+1} A^{-1} H_{k+1} C^p_{k+1} + \\ & G_{k+1} \left(H_{k+1} C^p_{k+1} H^T_{k+1} + C_{\eta_{k+1}} \right) G^T_{k+1} = \\ & C^p_{k+1} - 2C^p_{k+1} H^T_{k+1} A^{-1} H_{k+1} C^p_{k+1} + \\ & C^p_{k+1} H^T_{k+1} A^{-1} A A^{-1} H_{k+1} C^p_{k+1} = C^p_{k+1} - \\ & C^p_{k+1} H^T_{k+1} \left(2A^{-1} - A^{-1} A A^{-1} \right) H_{k+1} C^p_{k+1} = \\ & C^p_{k+1} - C^p_{k+1} H^T_{k+1} A^{-1} H_{k+1} C^p_{k+1}. \end{aligned} \tag{4.4}$$

By analyzing this sequence of identities we can conclude that (4.3) is equivalent to (4.2) if and only if $2A^{-1} - A^{-1}AA^{-1} = A^{-1}$. The latter trivially always holds regardless of matrix A. However, in LBFGS-EKF we do not use the exact inverse of this matrix and replace it by low-memory BFGS approximation. In other words, leveraging the notation used in algorithm 3 we conclude that generally $\widehat{A^{-1}} \neq 2\widehat{A^{-1}} - \widehat{A^{-1}} A \widehat{A^{-1}}$ and therefore (4.3) is not equivalent to (4.2). On the other hand, careful revision of (4.4) suggests immediate correction to this problem, which is summarized in the following lemma.

Lemma 4. *Let $A = H_{k+1} C^p_{k+1} H^T_{k+1} L + C_{\eta_{k+1}}$, where C^p_{k+1} is computed in accordance with algorithm 3 and $\widehat{C^{est}_k}$ is assumed to be non-negative definite. Then for an arbitrary symmetric matrix B^* matrix*

$$\widehat{C^{est}_{k+1}} = C^p_{k+1} - C^p_{k+1} H^{AD}_{k+1} \left(2I - B^* A \right) B^* H^{TL}_{k+1} C^p_{k+1}, \tag{4.5}$$

where I is identity matrix of conformable dimension, is non-negative definite and $\widehat{C^{est}_{k+1}}$ is equal to C^{est}_{k+1} when $B^* = A^{-1}$.

Proof. The proof immediately follows from (4.4). □

Hence, leveraging lemma 4 we can replace the steps 15–23 of algorithm 3 to correspond to (4.5) and with this change the algorithm is guaranteed to generate correct approximation of estimate covariance. An example of the eigenvalue distribution of an estimate covariance matrix calculated using the stabilizing correction is demonstrated in figure 4.2. The data shown in this figure were generated using exactly same settings as the data from figure 4.1 with the only change being introduction of the stabilizing correction into the algorithm.

It turns out that the result given in lemma 4 can be further improved by introducing recurrence into the stabilizing correction (4). Indeed, we have replaced approximation $\widehat{A^{-1}}$ by $(2I - \widehat{A^{-1}}A)\widehat{A^{-1}}$. Let us now generalize this idea by the following recurrent relation:

$$\mathcal{F}_n(X) = 2\mathcal{F}_{n-1}(X) - \mathcal{F}_{n-1}(X) A \mathcal{F}_{n-1}(X), \tag{4.6}$$

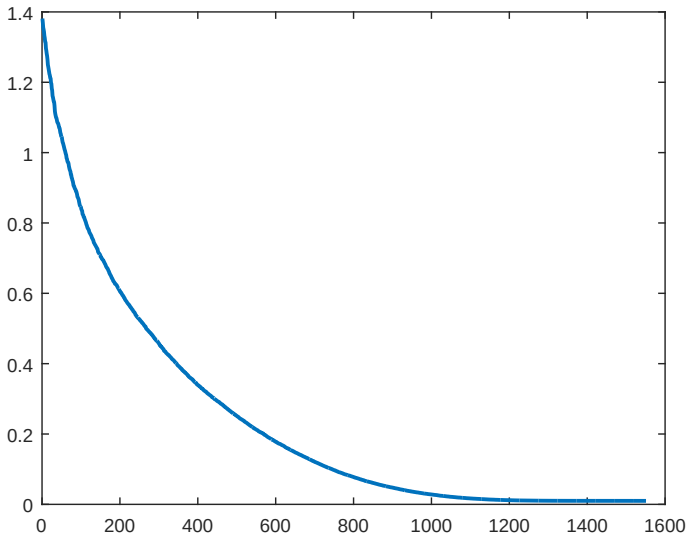

Figure 4.2: Eigenvalues of estimate covariance matrix computed by LBFGS-EKF with stabilizing correction

where $n \geq 2$, X is an arbitrary symmetric matrix and the recursion is initialized by $\mathcal{F}_1 = X$. It can be easily proved by induction that for any $n \geq 1$ matrix $\mathcal{F}_n(X)$ is symmetric and therefore it can be plugged as B^* into (4.5) and by lemma 4 the resulting approximation of the estimate covariance matrix will be non-negative definite. Moreover, it can be shown that in fact the sequence $\mathcal{F}_n$ converges to A^{-1} or under certain circumstances to pseudoinverse of the matrix A (refer to Ben-Israel (1966) and Pan and Schreiber (1991)). Therefore, not only (4.6) can be employed as a stabilizing correction, it also improves the approximation $\widehat{A^{-1}}$ provided by the L-BFGS. In fact, usage of (4.6) improves convergence of LBFGS-EKF towards the exact Kalman filter formulae with the rate of convergence growing as the value of n increases. The trade-off however is the computation cost, as calculation of values of $\mathcal{F}_n$ requires $3(n-1)+1$ runs of the tangent-linear and adjoint models. The relation between the convergence rate of the filter's approximation quality and the choice of n are clarified by the following lemma:

Lemma 5. *Suppose that the assumptions of lemma 4 hold. Let $\mathcal{F}_n(X)$ be the matrix identified by recurrence relation* (4.6)*. The the following inequality holds:*

$$\|C^{est}_{k+1} - \widehat{C^{est}_{k+1}}\| \leq \|A\|\|A^{-1}\|\|H_{k+1}C^{p}_{k+1}\|^2\|AX - I\|^{2^n-1}. \tag{4.7}$$

Proof. For the proof see Bibov et al. (2015). □

From lemma 5 one can see that when $\|AX - I\| < 1$ the estimate covariance $\widehat{C^{est}_{k+1}}$ calculated by the LBFGS-EKF with stabilizing correction provided by $\mathcal{F}_n$ converges to the

"optimal" estimate covariance C_{k+1}^{est} defined by the EKF with exponential convergence rate $O\left(\|AX - I\|^{2^n}\right)$. Therefore the actual speed of convergence depends on the choice of initial approximation X for A^{-1}. In contrast, if $\|AX - I\| > 1$, then not only is the convergence guaranteed, it can even become worse as the upper boundary for $\|C_{k+1}^{est} - \widehat{C_{k+1}^{est}}\|$ increases. Summing it up, one could suggest that usage of the generalized stabilizing correction (4.6) is feasible when it is possible to provide sufficiently good initial approximation for A^{-1}. However, when such approximation is not available it is recommended to fallback to the case where $n = 2$, i.e. to the basic correction defined by (4.5).

This wraps out our discussion of the stabilizing correction. In the next chapter we briefly introduce one use case of the LBFGS-EKF with this correction employed. This results in implementation of the parallel approximate Kalman filter, which is the second most significant result of the present dissertation.

4.2 Parallel Filtering Task

In this section we describe one application of the stabilizing correction introduced in 4.1, which turns out to be asymptotically equivalent to the fixed-lag Kalman smoother as formulated in formulated in Moore (1973). Let us firstly define the notions of combined state and combined observation. Consider the following tuples:

$$\bar{X}_k = \left(x_{k-L+1}, x_{k-L+2}, \ldots, x_k\right), \tag{4.8}$$

$$\bar{Y}_k = \left(y_{k-L+1}, y_{k-L+2}, \ldots, y_k\right), \tag{4.9}$$

where x_i and y_i for $i = k - L + 1, \ldots, k$ are the state and observation vectors for the respective time instants. Hereinafter, we will call the tuple (4.8) combined state and tuple (4.9) combined observation.

For the sets of combined states and combined observations it is possible to define the corresponding composite evolution and observation models:

$$\bar{X}_{k+1} = \bar{\mathcal{M}}_k\left(X_k\right) = \left(\mathcal{M}_{k-L+1}\left(x_{k-L+1}\right), \mathcal{M}_{k-L+2}\left(x_{k-L+2}\right), \ldots, \mathcal{M}_k\left(x_k\right)\right), \tag{4.10}$$

$$\bar{\mathcal{H}}_k\left(X_k\right) = \left(\mathcal{H}_{k-L+1}\left(x_{k-L+1}\right), \mathcal{H}_{k-L+2}\left(x_{k-L+2}\right), \ldots, \mathcal{H}_k\left(x_k\right)\right), \tag{4.11}$$

where $\bar{\mathcal{M}}_i$ and $\bar{\mathcal{H}}_i$ are evolution and observation operators as defined in (2.9) and (2.10). The beneficial property of the composite operators (4.10) and (4.11) is that they support natural parallelism. Therefore, we would like to leverage this property in our favour by performing data assimilation for the combined state $\bar{X}_k$. In order to do that, we still have to define error covariance matrices for the evolution and observation models. This again can be done in the most naïve way by arranging error covariance matrices C_{ϵ_k} and C_{η_k} (the notation was introduced in the beginning of section 2.3) into block-diagonal matrices.

More rigorously, the model error covariance can be defined as follows

$$\bar{C}_{\epsilon_k} = \begin{pmatrix} C_{\epsilon_{k-L+1}} & O & \dots & O \\ O & C\epsilon_{k-L+2} & \dots & O \\ \dots & \dots & \dots & \dots \\ O & O & \dots & C_{\epsilon_k} \end{pmatrix}, \tag{4.12}$$

and the observation error covariance is defined similarly by

$$\bar{C}_{\eta_k} = \begin{pmatrix} C_{\eta_{k-L+1}} & O & \dots & O \\ O & C\eta_{k-L+2} & \dots & O \\ \dots & \dots & \dots & \dots \\ O & O & \dots & C_{\eta_k} \end{pmatrix}, \tag{4.13}$$

where O denotes all-zero matrices of conformal dimensions. It is also possible to introduce non-zero off-diagonal elements into (4.12) and (4.13), which would effectively enable cross-time correlation for the error terms: property that is disallowed for the standard data assimilation task due to assumptions made during derivation of the Kalman filter.

It is necessary to emphasize that combined evolution model (4.10) and the corresponding observation operator (4.11) imply state and observation overlapping. Indeed, observation y_k from (4.9) appears in L combined observations $\bar{Y}_k, \bar{Y}_{k+1}, \dots, \bar{Y}_{k+L-1}$. Hence, the combined observations are strongly correlated in time. This breaks the assumption of uncorrelated measurement errors employed during derivation of the Kalman filter. It is possible however to alleviate this problem by using the estimates computed during preceding filtration passes in order to create artificial measurement noise without cross-time dependency. More precisely, given that x_k^{est} is an unbiased estimate of x_k obtained by filtration at certain time instance k and assuming that the observation operator $\mathcal{H}_k$ is linear, one can remove cross-time correlation between the re-used observations by simply redefining each re-used observation as $\tilde{y}_k = \mathcal{H}_k\left(x_k^{est}\right) + w$, where $w \sim \mathcal{N}(0, \sigma^2)$ and σ^2 is the variance of the measurement error. When the observation operator $\mathcal{H}_k$ is non-linear, we can apply the same trick when the time step is "small enough" as the estimates of the filter remain asymptotically optimal.

In practice the cross-time correlations can be reduced by introducing a lag in update of the analysis covariance C_k^{est}. This approach is used in some implementations of the Weak-Constraint 4D-VAR and is discussed by Fisher et al. (2011). In our practice even the latter was not needed as at each data assimilation step the new "injected" observation was introducing enough statistical variance and the quality of the resulting estimates was satisfactory.

We can now define data assimilation task for observed dynamical system (4.10)–(4.11) with error terms (4.12)–(4.13). The estimate X_k^{est} will then identify all the states within the time window $k-L+1, k-L+2, \dots, k$ and will simultaneously take into account all the observed data collected within this time window. We expect the later property to be beneficial since, say, the state located in the middle of the time window can take advantage both from the data collected in the past and the data obtained in the future relatively

to its own location in time. Naturally, this composite data assimilation task increases dimension of the original state space proportionally to the length of the time window. This further complicated the estimation problem when dimension of the original state space is large. However, it is possible to partially alleviate this caveat by leveraging the LBFGS-EKF with the stabilizing correction described in the previous section. In Bibov and Haario (2016) we demonstrate that the estimates obtained this way have better quality than with the basic stabilized LBFGS-EKF. In addition, the algorithm can be efficiently implemented on a parallel system taking into account natural parallelization properties of (4.10) and (4.13).

The proposed approach comes very close to the Weak-Constraint 4D-VAR described in section 3.6. In final part of this chapter we would like to briefly compare the differences and similarities of both approaches. Firstly, let us restate the task of estimating combined state X_k in variational form as was described in section 2.4. In other words, we need to minimize the following cost function:

$$\begin{aligned}\mathcal{L}_1(X_k) = &(X_k - X_k^p)^T \left(\bar{C}_k^p\right)^{-1} (X_k - X_k^p) + \\ &\left(Y_k - \bar{\mathcal{H}}_k^{TL}(X_k)\right)^T \bar{C}_{\eta_k}^{-1} \left(Y_k - \bar{\mathcal{H}}_k^{TL}(X_k)\right),\end{aligned} \tag{4.14}$$

where $\bar{\mathcal{H}}_k^{TL}$ is the tangent-linear code of the composite observation operator $\bar{\mathcal{H}}_k$. The cost function that one has to minimize as required by the Weak-Constraint 4D-VAR was derived in section 3.6 and in case where the mean model error $\bar{\epsilon}_j = \bar{0}$ reads as follows:

$$\begin{aligned}\mathcal{L}_2(X_k) = &\frac{1}{2}\left(x_b^k - x_k\right)^T B_k^{-1}\left(x_b^k - x_k\right) + \\ &\frac{1}{2}\sum_{j=k-L+1}^{k} \left(y_j - \mathcal{H}_j(x_j)\right)^T R_j^{-1}\left(y_j - \mathcal{H}_j(x_j)\right) + \\ &\frac{1}{2}\sum_{j=k-L+1}^{k-1} \left(x_{j+1} - \mathcal{M}_j(x_j)\right)^T Q_j^{-1}\left(x_{j+1} - \mathcal{M}_j(x_j)\right).\end{aligned} \tag{4.15}$$

Let us assume that matrix $\bar{C}_{\eta_k}$ is block-diagonal. Therefore, the matrix $\bar{C}_k^p$ is also block-diagonal. Let matrices C_{k-L+1}^p, $C_{k-L+2}^p, \ldots, C_k$ and $C_{\eta_{k-L+1}}, C_{\eta_{k-L+2}}, \ldots, C_{\eta_k}$ be the diagonal blocks of $\bar{C}_k^p$ and $\bar{C}_{\eta_k}$ respectively. Hence, it is possible to recap (4.14) in a form similar to (4.15):

$$\begin{aligned}\mathcal{L}_1(X_k) = &\sum_{j=k-L+1}^{k} \left(x_j - x_j^p\right)^T \left(C_j^p\right)^{-1}\left(x_j - x_j^p\right) + \\ &\sum_{j=k-L+1}^{k} \left(y_j - \mathcal{H}_j^{TL}(x_j)\right)^T C_{\eta_j}^{-1}\left(y_j - \mathcal{H}_j^{TL}(x_j)\right).\end{aligned} \tag{4.16}$$

Plain comparison of (4.15) and (4.16) suggests that the parallel filter has similar form to single-step inner-loop implementation of the Weak-Constraint 4D-VAR with constant lin-

earization applied to the forecast discrepancy term (see section 3.6 for the details regarding definition of the inner-loops). In other words, the model terms $\mathcal{M}_j(x_j)$ can be approximated with accuracy of $O(\|x - x_j\|)$ by constants $x_j^p = \mathcal{M}_j(x_j)$. This approximation is slightly weaker that the linear-order approximation $\mathcal{M}_j(x_j) \approx x_j^p + M_j^{TL}\left(x_j - x_j^p\right)$ suggested in the original formulation of the Weak-Constraint 4D-VAR. On the other hand, the Weak-Constraint 4D-VAR does not identify the model error covariance terms Q_j, which should be obtained by hand-tuning. In parallel filter this problem is gracefully alleviated by the standard Kalman filter framework as the prediction error covariance matrices are computed by propagating covariance of the preceding estimate using the tangent-linear code of the evolution model. In addition, (4.14) allows the individual states x_j to have cross-time correlations, whereas this feature is completely ignored by (4.15). Finally, the parallel filter can be implemented in a very similar manner to the Weak-Constraint 4D-VAR by repeated minimization of (4.16), i.e. one has to firstly compute the forecasts x_j^p, minimize (4.16), then update x_j^p using the result of minimization, minimize (4.16) again and so on. Such implementation of the parallel filter closely mimics the inner-loops used in the 4D-VAR.

This finalizes our discussion of the parallel filter. In the last section of this chapter we briefly introduce the two-layer quasi-geostrophic model that was employed to estimate performance of our data assimilation algorithms.

4.3 Overview of The Toy-Case Model

This section is dedicated to discussion of the two-layer Quasi-Geostrophic model (see Pedlosky (1987)), which we employed as a reference model to estimate performance of data assimilation algorithms. We begin this section with a brief description of model's geometric layout and continue by introducing the system of the governing partial differential equations. In addition, we emphasize the main steps of the numerical solver that we developed to integrate the model: semi–Lagrangian advection and s set of finite-difference integration stages applied to Poisson and Helmholtz equations that appear as intermediate steps of the integration. We conclude this section by a concise review of the tangent-linear and adjoint codes of the QG-model and provide schematic diagrams of their implementations.

The two-layer Quasi-Geostrophic model simulates behaviour of quasi-geostrophic (slow) wind motion. The model is defined on a cylindrical surface vertically divided into two "atmospheric" layers. Periodic boundary conditions are employed along the latitude direction. The boundary conditions on the northern and southern poles (the bottom and top of the cylinder) are set to predefined constant values. In addition, the model takes into account surface irregularities (orography) which affect the bottom layer. Aside from the boundary conditions and orography, the model parameters comprise the depths of the atmospheric layers, the potential temperature flow rates across the layer interface and the mean potential temperature. A possible geometrical topology of the model is illustrated in Figure 4.3. In this figure U_1 and U_2 denote mean zonal flows in the top and the bottom layers of the model.

Prior to integration the model is reduced to non-dimensional representation. That non-

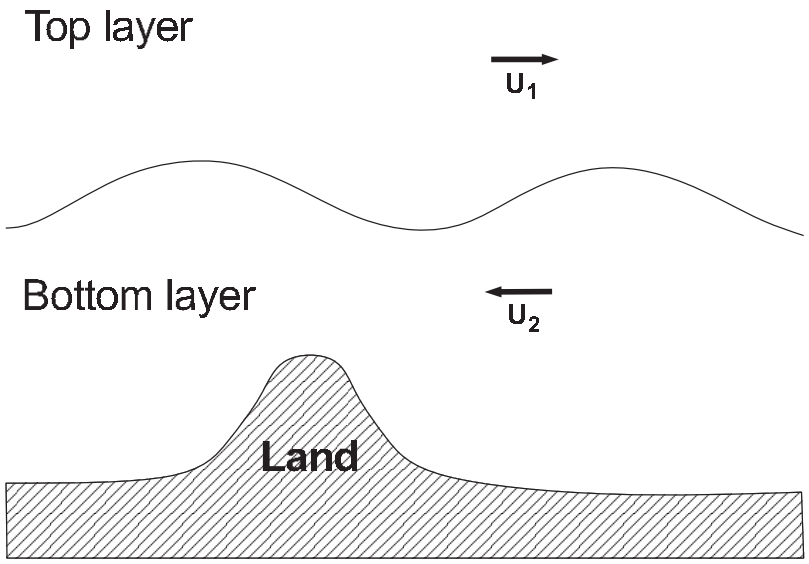

Figure 4.3: Geometric layout of the two-layer quasi-geostrophic model

dimensionalization procedure is determined by constant terms L and U (the length scale and the speed scale of the system), and by depths D_1 and D_2 of the top and of the bottom layers of the model.

The two-layer Quasi-Geostrophic model operates on the components of potential vorticity and on the stream function, which is analogous to pressure. The dynamics of the model assume that potential vorticity respects certain conservation law, which is mathematically defined as follows:

$$\frac{D_1 q_1}{Dt} = 0, \ \frac{D_2 q_2}{Dt} = 0. \tag{4.17}$$

Here the terms q_i denote the potential vorticity function; index $i = 1, 2$ indicates the top and bottom atmospheric layers of the model respectively; operators D_i denote substantial derivatives assuming zonal and longitudinal wind speeds u_i and v_i, i.e. $\frac{D_i \cdot}{Dt} = \frac{\partial \cdot}{\partial t} + u_i \frac{\partial \cdot}{\partial x} + v_i \frac{\partial \cdot}{\partial y}$, where the terms u_i and v_i are computed by the wind operator below:

$$u_i = -\frac{\partial \psi_i}{\partial y}, \ v_i = \frac{\partial \psi_i}{\partial x}. \tag{4.18}$$

In (4.17) ψ_i is the stream function which is related to potential vorticity as described by the following coupled system of equations:

$$q_1 = \nabla^2 \psi_1 - F_1 (\psi_1 - \psi_2) + \beta y, \tag{4.19}$$

$$q_2 = \nabla^2 \psi_2 - F_2 (\psi_2 - \psi_1) + \beta y + R_s. \tag{4.20}$$

Here R_s and β specify non-dimensional orography and northward gradient of the Coriolis parameter hereafter denoted as f_0. More precisely, the non-dimensional parameters that

appear in (4.19) and (4.20) are related to dimensional terms by the following identities:

$$F_1 = \frac{f_0^2 L^2}{\acute{g} D_1},\ F_2 = \frac{f_0^2 L^2}{\acute{g} D_2},$$
$$\acute{g} = g\frac{\Delta\theta}{\bar{\theta}},$$
$$R_s = \frac{S\left(x,y\right)}{\eta D_2},$$
$$\beta = \beta_0 \frac{L}{U},$$

where $\Delta\theta$ specifies the potential temperature change across the layer interface, $\bar{\theta}$ is the mean potential temperature, g is acceleration of gravity, $S(x,y)$ defines dimensional orography surface, and $\eta = \frac{U}{f_0 L}$ is the Rossby number associated with the given system. Hence, the two-layer Quasi-Geostrophic model is defined by system of partial differential equations (4.17), (4.18), (4.19) and (4.20).
For numerical solution it is assumed that potential vorticities q_1 and q_2 for certain time instance are provided. Then (4.19) and (4.20) define a system of PDE's with respect to spatial variables only. Let us apply ∇^2 to equation (4.19) and subtract the sum of F_1 times (4.20) and F_2 times (4.19) from the result, which leads to the following equation:

$$\begin{aligned}&\nabla^2\left[\nabla^2\psi_1\right] - \left(F_1+F_2\right)\left[\nabla^2\psi_1\right] = \\ &= \nabla^2 q_1 - F_2\left(q_1 - \beta y\right) - F_1\left(q_2 - \beta y - R_s\right)\end{aligned} \tag{4.21}$$

We note that (4.21) forms a non-homogeneous Helmholtz equation with negative parameter $-\left(F_1+F_2\right)$ (refer to Riley et al. (2004) for details) with respect to unknown $\nabla^2\left[\psi_1\right]$. The boundary conditions for this problem are inherited from the main QG-model formulation (4.17)-(4.20). In our implementation we integrate the Helmholtz equation (4.21) using finite-difference approximation. Let us assume, that χ solves the task (4.21). Hence, the problem converts to the Poisson equation (see Evans (1997)) defined with respect to the stream function in the top atmospheric layer ψ_1:

$$\nabla^2\psi_1 = \chi \tag{4.22}$$

This equation can be integrated using the standard 5-point Laplacian approximation. It has been shown that the discretized Poisson equation can always be reduced to a system of linear equations, which has unique solution and is well-posed (the proof can be found for example inLeveque (2007)). Finally, we implement the time stepping by calculating potential vorticity $q_i\left(t+\Delta t\right)$ given $q_i(t)$. This is done by means of the standard semi-Lagrangian scheme described for instance in Staniforth and Côté (1991).
Summarizing the aforesaid, the discrete solver for the two-layer Quasi-Geostrophic model is composed of advection procedure evolving potential vorticity and of doubled application of finite–difference method, first to solve the Helmholtz equation (4.21) and second to integrate the Poisson equation (4.22). This allows to compute discretized stream ψ_1 in the top model layer. The stream ψ_2 in the bottom layer can be computed by plugging the

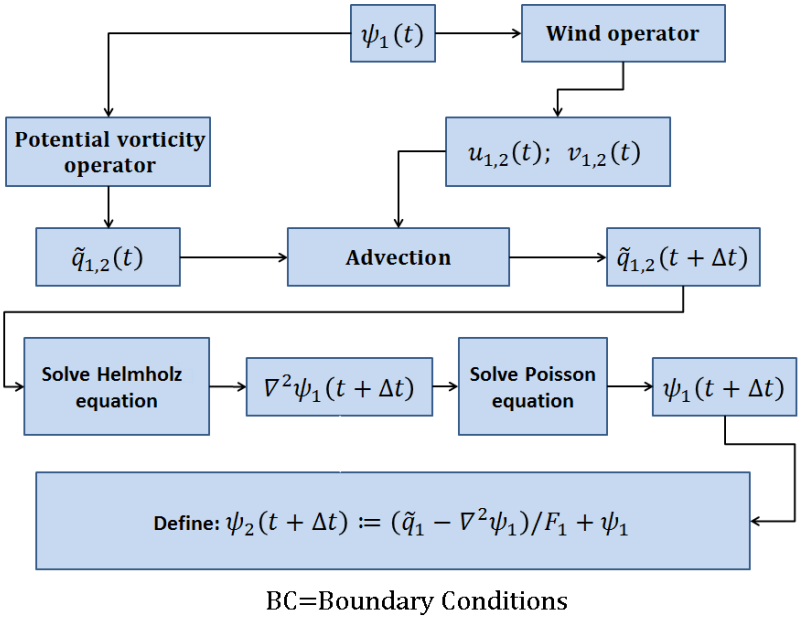

Figure 4.4: Non-linear solver for the two-layer quasi-geostrophic model

value obtained for ψ_1 into the formulae (4.19),(4.20), which leads to an algebraic equation with respect to ψ_2. The integration flowchart illustrating our implementation of the solver is given in Figure 4.4.

The tangent-linear and adjoint codes for the two-layer quasi-geostrophic model can be derived by firstly determining the tangent-linear and adjoint codes for each step of the solver independently. This can be done by analyzing implementation of the non-linear solver and then leveraging the rules formulated in section 3.5. These individual codes are thereafter combined by means of the chain rule (Marsden and Weinstein (1985)). Since the derivation details of these models are routine yet ponderous, we omit them here and only sketch the general layouts. Figure 4.5 provides schematic diagram of our implementation of the TL-code. It can be seen from the figure that the TL–model solver closely mimics the non-linear pipeline illustrated in Figure 4.4. The main difference is that the procedure of advection is now linearized. In addition, in contrast to the non-linear solver the intermediate equations now employ homogeneous boundaries instead of the constant settings as the constants have vanished during linearization of the model.

Finally, the AD-code for the qg-model is presented in Figure 4.6. This scheme can be obtained by carefully analysing each step of the TL-integration, however we skip the details for the reasons of brevity.

This wraps out our review of the two-layer quasi-geostrophic model. This model was used in our numerical experiments aimed to estimate performance of new data assimila-

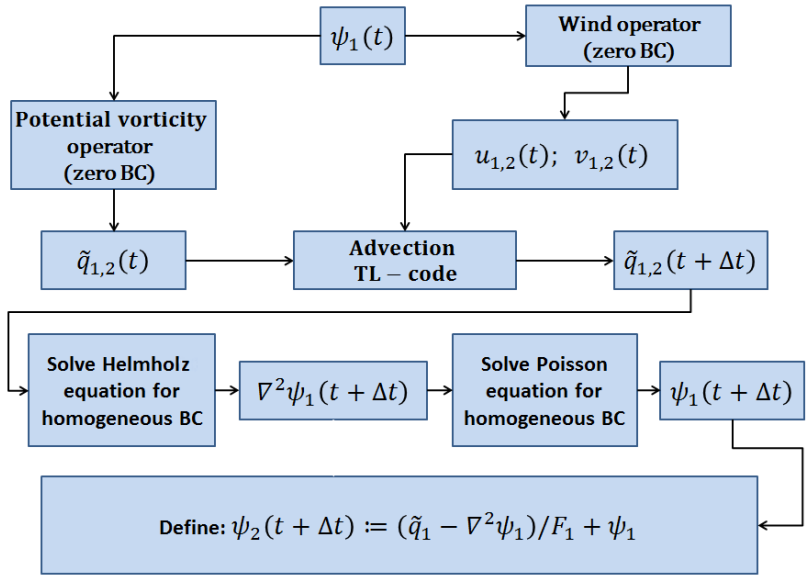

Figure 4.5: Tangent-linear code of the quasi-geostrophic model

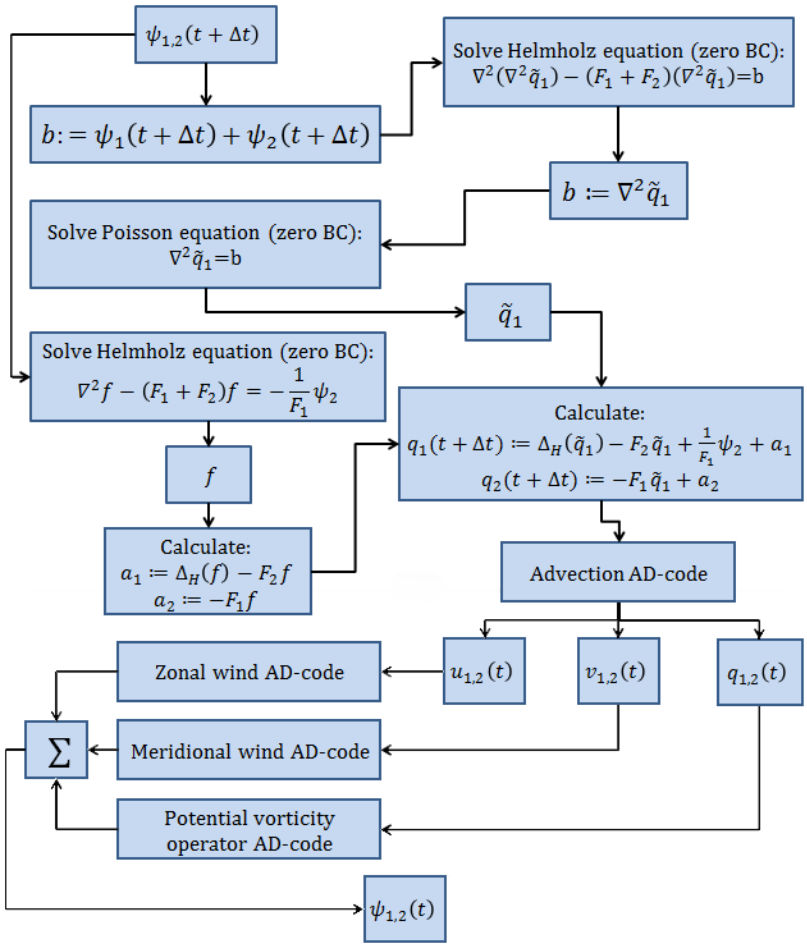

Figure 4.6: Adjoint code of the quasi-geostrophic model

tion approaches and to compare them with existing solutions (see Solonen et al. (2013), Solonen et al. (2014), Bibov et al. (2015), and Bibov and Haario (2016)).

5 Estimating Chlorophyll in The Finnish Gulf

The last chapter of this dissertation is dedicated to the case provided by Finnish Environmental Institute (Suomen ympäristökeskus – SYKE). The task was to estimate concentration of chlorophyll in the Helsinki bay area over certain period of time given observation data collected from various sources. In the first part of this chapter we describe the formal definition of the problem and discuss the difficulties that prevent implementation of standard solutions. In the second part of this chapter we consider realization of our parallel approximate Kalman filter introduced in section 4.2 and how it can be leveraged to solve the chlorophyll estimation problem.

5.1 Problem Formulation

In this section we consider the problem of estimating concentration of chlorophyll in the region of Finnish gulf illustrated in figure 5.1. The concentrations are to be identified on a discrete grid with 85 latitudinal and 210 longitudinal nodes, which means that the state space has dimension of 17850 components. This dimension is already too restrictive to allow solution of the estimation problem by the means of standard tools like the extended Kalman filter. Indeed, if the state space has dimension of 17850, its covariance matrix would have 318622500 entries, which corresponds to roughly 1.19GBs of RAM in single precision or twice this amount in double precision. Moreover, inversion of such matrix or computing a matrix-matrix product with this dimension would require about $5.7 \cdot 10^{12}$ multiplications, which would take about 8 hours of compute time on a 200GFlops CPU (roughly equivalent to an Intel Core i7 6700K operating at 4GHz). Ultimately, the practical time of the computation may be significantly longer than that as the CPU does not always work at its peak performance and most programs cannot achieve the full occupancy. Therefore, even though such implementation is feasible on today's consumer-level equipment (top-level at the moment of writing this text), an increase in resolution of the grid or reduction of the time step between consequent estimates would certainly make it impracticable.

A simple way to reduce dimension of the problem in order to make the implementation of the standard Kalman filter more efficient is to average the discretization grid and the corresponding observed data. More precisely, the idea is to pre-process the data fields by, say, Gaussian filter and apply the standard data assimilation tools (e.g. Kalman filter and Kalman smoother) to the pre-processed values of lower dimension. Naturally, due to the coarser resolution this inevitably leads to precision losses in the estimates. An alternative approach is to apply one of the low-memory data assimilation algorithms. In particular, in the following section of this chapter we will discuss how the chlorophyll concentration estimation task introduced above can be solved by the means of the parallel Kalman filter considered in section 4.2 and how its performance compares against the estimates generated by the standard Kalman filter and smoother applied to the averaged data fields.

Before delving into detailed discussion of our solution for the problem we still need to clarify the nature of the observed data. In the considered case the data was collected from

Figure 5.1: Chlorophyll estimation problem domain: considered area of the Finnish gulf in between 59.891 and 60.143 latitudes and 24.415 and 25.042 longitudes.

three sources: satellite measurements, monitoring conducted in stationary stations, and the Alg@line researh initiative (refer to SYKE (2016)). The satellite measurements are obtained by spectral analysis of the sea area. These observations are the most dense and cover the whole discretization grid (i.e. there is an observation associated with each node of the grid). However, the technicalities of these observations require the observed water region to be at least few meters deep and the region should not be occluded by clouds. Therefore, on a cloudy day some of the satellite measurements (or even all the measurements) may not be available. The second source of the observations is the stations. The data collected from these stations are less vulnerable to uncontrollable phenomena like cloudiness, but the observed locations and the time rate of collecting the measurements are scarce. Finally, the third source is from the Alg@line: this is a research program which among other things includes collection of various data (including concentration of chlorophyll) by ships during operation on their normal duties. These observations are available on a regular basis, but their coverage is limited by itineraries of the ships. In our solution we want to implement a "data fusion", i.e. utilize all three sources of the observed data using the same concise algorithmic interface.
Based on the nature of the observation sources, we expect that the satellites will have the most significant impact on the estimates as they have the best aerial coverage. However, when making an estimate we need to account for several observation instances simultaneously in order to gain extra data for the scarcely observed regions (i.e. the regions that were badly observed due to the presence of clouds). In other words, if certain region was cloudy on June 17 and clear on June 18, we have to take into account both days when making an estimate for, say, June 17. This exact feature is provided by the parallel filter introduced in section 4.2, which makes it a perfect candidate for solving the proposed chlorophyll estimation problem. In the following section we explain its implementation details and consider the results of data assimilation.

5.2 Implementation of The Parallel Approximate Kalman Filter

In our realization of the parallel filter we assumed the window length of 5 steps, i.e. a single data assimilation loop allowed simultaneous estimation of 5 consequent states given the observed data collected within related time period. Since in the beginning of assimilation the prior estimates for the total window included into analysis are not available we needed to deploy a spin-up period. In our implementation we began by considering only a single state vector and computed its estimate using the stabilized L-BFGS EKF described in section 4.1. Then we extended our analysis by considering this estimate together with the forecast computed using this estimate, which corresponds to the window length of two states. Thereafter, we re-estimated these two states by using the parallel filter and again extended the window by including the forecast computed using the last estimate produced by the analysis. This process continues until the desired length of the window (5 state vectors in the considered case) is achieved. A graphical illustration of this procedure is presented in figure 5.2. After the full length window is established the data assimilation continues by adding the new forecasts into analysis similarly to how it was done during the spin-up period and at the same time discarding the state corresponding

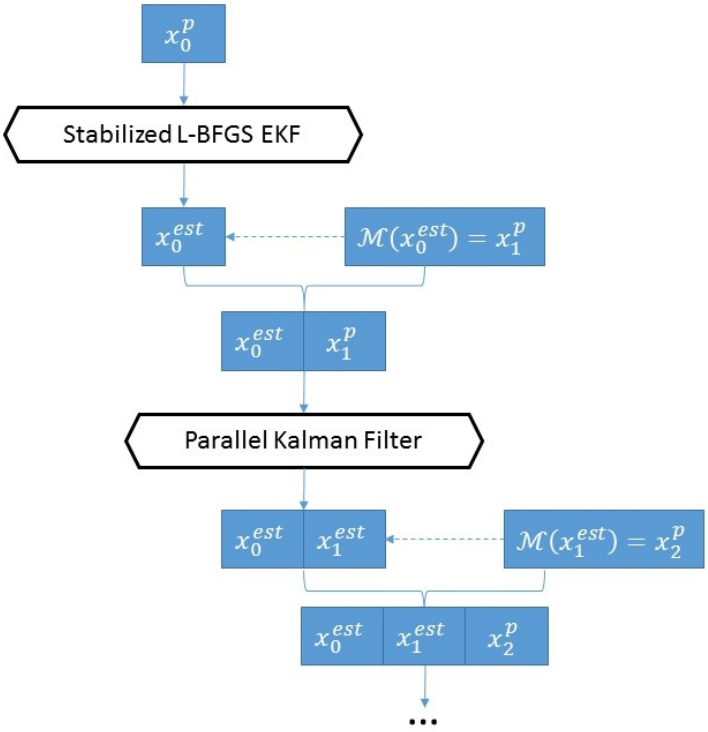

Figure 5.2: Spin-up period of data assimilation implemented using the parallel Kalman filter

to the oldest time instance. In other words after the spin-up period each data assimilation round shifts the window forward for one assimilation time step. This procedure is illustrated in figure 5.3.

In order to fully describe the estimation process we need to identify the evolution model from (2.9), the observation model from (2.10) and the related error covariance matrices C_{ϵ_k} and C_{η_k} (see section 2.3 for details regarding this notation).

The transition model $\mathcal{M}_k$ was implemented by the moving average filter with Gaussian 9-by-9 kernel. This means in accordance with (2.9) that forecasts were computed by averaging their preceding state and adding Gaussian noise with pre-defined variance. In our implementation we set this variance to 10^{-3}, i.e. $C_{\epsilon_k} = 10^{-3}I$. This particular value was chosen to be slightly less then the variance of the satellite measurements, which was done in order to give the model certain preference over the measured data and reduce time discontinuities that occur in regions with scarce observational updates. The reason to choose the moving average filter as the prediction model is alleviation of spatial discontinuities that may occur in the estimated chlorophyll concentration fields due to scarcity

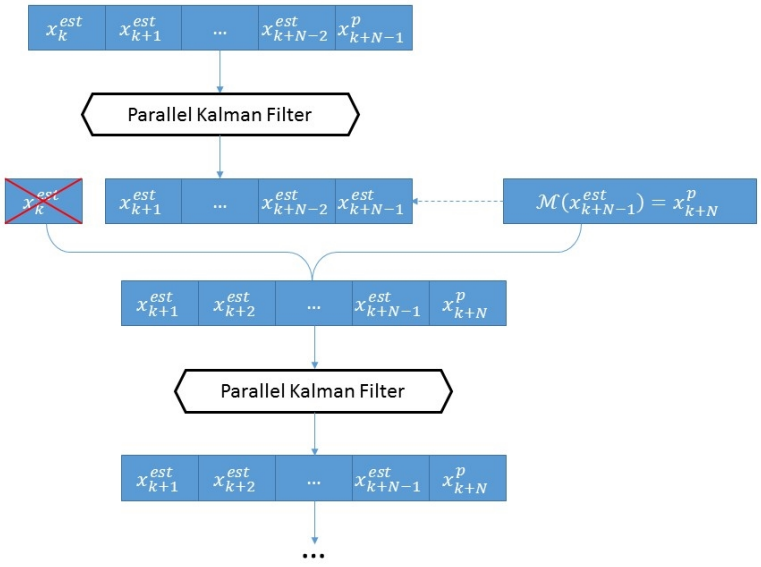

Figure 5.3: Data assimilation with parallel Kalman filter after the spin-up period

of observed data in some regions. By using the moving average model to compute the forecasts we essentially introduce spatial correlations between the nodes of discretization grid. Similar result can be achieved by using distance-dependent correlation matrix to define the forecast error covariance C_{ϵ_k}. Such correlation matrix is often defined using exponential decay:

$$A_{i,j} = L \cdot \mathbf{exp}\left(-\|p_i - p_j\|_{\mathbb{E}}\right) + \delta_i^j \tau, \tag{5.1}$$

where L is a scaling factor, p_i and p_j are the spatial points corresponding to the indices i and j of the state vector, δ_i^j is Dirac discrete delta-function, and τ is an auxiliary nugget value. Unfortunately, implementation of the correlation matrix (5.1) in low-memory operator form is not trivial as it quickly becomes compute-bound due to demanding exponent compute operation. Therefore, it cheaper and easier to model analogous spacial correlation effect by using the moving average filter as the prediction model and let the Kalman filter framework compute the otherwise expensive covariance data. In addition, the averaging operator is obviously linear and symmetric. Therefore, its tangent-linear and adjoint codes coincide with the operator itself and there is no need to implement them separately.

Our observation operator was defined by simple bilinear interpolation, i.e. the input state vector was mapped onto a set of observed locations (which were different at each time instance) by interpolating the values of the state onto these locations. Since this operator is linear its tangent-linear code coincides with the operator itself and the adjoint model can be derived using the set of rules introduced in chapter 3.5.

The observation error covariance matrix C_{η_k} was diagonal. The variance values for this

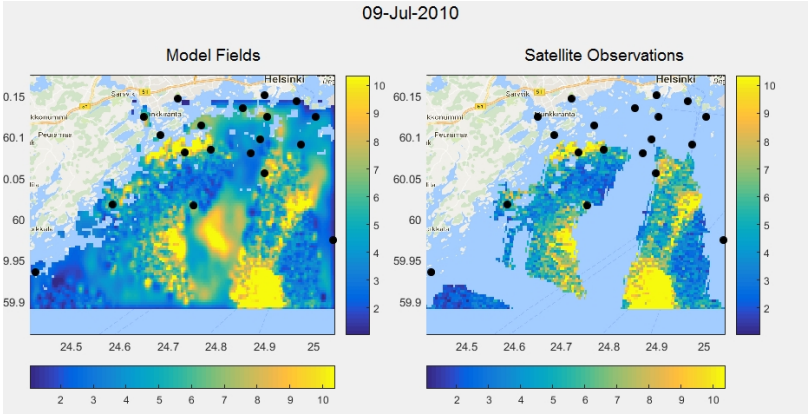

Figure 5.4: Estimated chlorophyll concentrations on 9.07.2010 computed using the parallel Kalman filter

matrix were provided by SYKE together with the measured data. However, for the satellite observations a fixed value of 0.0025 was used for the variance.

In our experiment we study performance of the parallel Kalman filter by comparing its results with the estimates calculated using the EKF combined with the Rauch-Tung-Striebel smoother (see Rauch et al. (1965)). As was emphasized in the previous section the natural dimension of the system being analysed prevents straightforward implementation of the standard Kalman filter framework. Therefore, in order to overcome this restriction we average the provided data fields and re-use the result of the averaging as new observations, which essentially reduces the dimension of the system. For this case we model the transition operator by trivial identity mapping, which means that the state evolution can be described by equation

$$x_{k+1} = x_k + \sigma\epsilon_k, \tag{5.2}$$

where $\epsilon_k \sim \mathcal{N}(0, 1)$ and as before the variance of the state components is set to $\sigma^2 = 10^{-3}$. However, with this transition operator the model error covariance C_{ϵ_k} cannot be diagonal, like it was in the case of the parallel Kalman filter implementation, since we now need to take into account spatial correlations between the state elements. Therefore, we leverage the distance correlation matrix (5.1) to define C_{ϵ_k}, which turns out to be computationally feasible for the averaged system having reduced dimension.

In figure 5.4 we present as an example the chlorophyll concentration for the 9th of July, 2010 estimated using the parallel filter. From the illustration one can observe how the areas with missing observations are filled by the filter, which utilizes the data from the past and from the future dates (hence, this "filter" is essentially a low-memory fixed-lag smoother). In figure 5.5 the concentrations for the same date are estimated using the averaging (as described above). The averaged estimates demonstrate similarities with the results produced by the parallel Kalman filter. However, due to the coarser observation fields many smaller details are lost, which suggests that the parallel filter can be the method of choice when there is a need to capture smaller-scale phenomena.

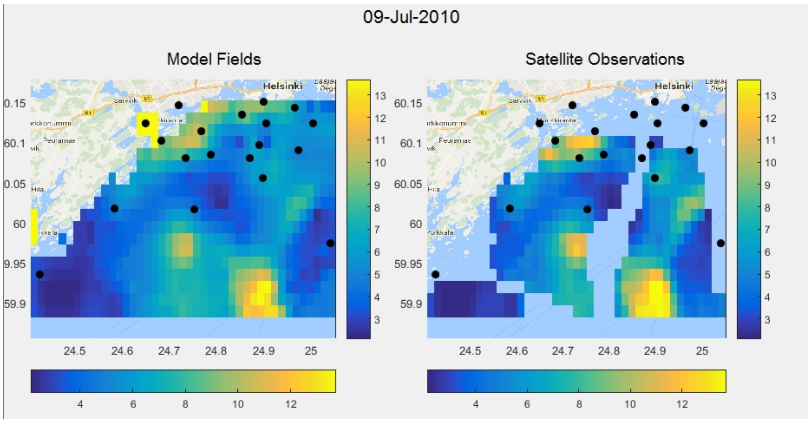

Figure 5.5: Estimated chlorophyll concentrations on 9.07.2010 computed using data averaging, extended Kalman filter and Rauch-Tung-Striebel smoother

We should also notice that the parallel filter can provide substantially better estimates than the aforementioned average-then-filter approach. We have considered a synthetic case based on the "concentration" fields generated using the fractal noise (i.e. the standard Perlin noise scaled multiple times and added to itself). We conducted data assimilation twice firstly following the average-then-filter approach as explained above and then using the parallel filter and averaging its estimates in order to make the results of both runs comparable. The root mean square errors computed during the experiment are presented in figure 5.6. From the figure it is easy to observe that the filter-then-average strategy that was using the parallel filter produces considerably more accurate estimates than that of the average-then-filter low dimensional approach. This result is natural since the parallel filter is able to take into account smaller scale structures of the states that are virtually ignored by the average-then-filter strategy.
This finalizes the present chapter. In the final section we summarize the main results of this work and present the initiatives aimed at continuation of our research.

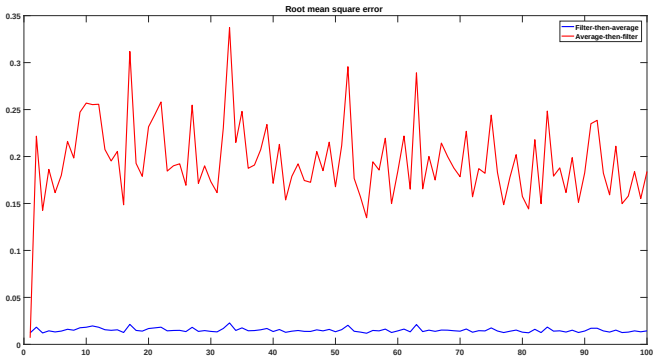

Figure 5.6: Root mean square errors computed using average-then-filter and filter-then-average strategies

6 Summary and Conclusions

6.1 Main goals and results overview

The main goal of the present work was to extend the known optimal filtering methods for the cases where the dimension of the state space of the underlying dynamical system is restrictively large. For that purposes we have studied a number of approximate sub-optimal algorithms based on various optimization techniques. Among the others we considered the family of L-BFGS approximate Kalman filters (L-BFGS EKF and VKF). During our research we have demonstrated that the previously known L-BFGS EKF suggested by Auvinen et al. (2009a) has certain stability issues due to the approximation technique leveraged in its derivation. The first main result of the present dissertation is re-formulation of the L-BFGS EKF, which corrects the issue. In addition, the idea used in the corrected algorithm is extended to define a family of stabilizing corrections (4.6). In the present work we have shown that the suggested stabilizing corrections successfully alleviate the instability issues appeared in the original L-BFGS EKF and moreover guarantee advantageous asymptotic behaviour of the low-memory approximation (i.e. the approximate filter ends up having higher convergence rate towards the exact Kalman filter formulae).

In the second part of this work we considered various techniques aimed at parallelization of the data assimilation routines. When data assimilation process needs to be parallelized (which is often demanded today due to the growing parallelism potential of the hardware) there are essentially two methods of choice: the ensemble Kalman filter or the Weak-Constraint 4D-VAR. For the ensemble Kalman filter there are large number of different formulations and implementations available (see for example Anderson (2001), Anderson (2006), Bardsley et al. (2013), Barth et al. (2010), Bengtsson et al. (2003), Houtekamer and Mitchell (1998), or Houtekamer et al. (2014)). As a side result of the present study we developed a localization scheme based on correlation functions suggested by Gaspari and S.E. (1999), which can be used together with randomize-then-optimize ensemble Kalman filter proposed in Solonen et al. (2013). However, we wanted to take full advantage of the tangent-linear and adjoint models, which are not used by the ensemble-based methods. Therefore, we wanted to find an analogue of the Weak-Constraint 4D-VAR formulated using the usual Kalman filter framework. For this purpose we decided to leverage our stabilizing correction and apply it to solve the so-called parallel filtering task, which essentially includes into analysis several consequent time instances to be considered simultaneously. We have shown the stabilizing correction is able to efficiently solve the parallel filtering task and that the resulting estimates are having better accuracy in comparison to the stabilized L-BFGS EKF.

In the present study we considered and compared a large number of approximate data assimilation algorithms. All these methods were implemented and compared using the two-layer quasi-geostrophic model, for which a numerical integration scheme was proposed. We used this model and our implementations of the approximate filtering techniques suggested in the previous literature in order to estimate potential performance of our methods. The results of these comparisons as well as related analysis are provided in

the publications accompanying this work.
Finally, we demonstrated the practical use of the techniques proposed in this dissertation by applying our methods to assess concentrations of chlorophyll in the Finnish gulf. For this case we used real observations provided by Finnish environmental institute (SYKE) and compared obtained results with the estimates computed using data fields averaging combined with standard data assimilation routines. Our experiments show notable improvement in the estimates on the smaller-scale level without considerable increase in compute time, which suggests that the low-memory data assimilation algorithms proposed in the present work may be recognized as the methods of choice for the cases, where straightforward application of the standard estimation framework is not possible due to the size of the problem.

6.2 Future research topics not covered in this dissertation

Since the low-memory approximate methods considered in this study are essentially suboptimal estimation schemes it becomes hard to predict their performance in application to a given large-scale dynamical system. For example, previous studies (see for example Auvinen et al. (2009a) or Bardsley et al. (2011)) assumed efficacy of their proposed approaches based on numerical experiments built around the Lorenz-95 model, which is a widely used toy-case. However, when applied to the two-layer quasi-geostrophic model (even in resolution as low as having only 1600 discretization grid nodes), the methods that seemingly behaved well with the Lorenz-95 model often tend to produce the estimates of much worse accuracy then expected. The reason for this behaviour is that the two-layer quasi-geostrophic model is advection-dominated, i.e. its state is mainly determined by the advection operator. Therefore, for the future benchmarking of the data assimilation methods we decided to employ a model, which would experience a highly chaotic dynamics similar to that of the two-layer quasi-geostrophic model. In adddtion, we wanted to be able to run this model on consumer-level hardware at resolutions having several millions of discretization grid nodes, which effectively makes it similar to the operation numerical weather prediction systems. For this purpose we developed a high-performance implementation of the shallow water system, which runs of single or multiple graphics processing units (GPUs). The method we use to integrate the shallow water equations is a central-upwind scheme of second order proposed by Kurganov and Petrova (2007). This scheme is essentially separable and can be decomposed into large number of smaller integration tasks with limited interactions, which enables efficient realization of this method for GPUs. With such implementation we were able to run simulations with resolution of the discretization grid up to 3000-by-3000 nodes. In order to be able to efficiently visualize a system that large we have developed specialized simulation graphics engine based on OpenGL 4.5 and modern shading techniques. An example of simulation visualized using our graphics engine is presented in figures 6.1 – 6.4

Figure 6.1: Shallow water and atmospheric scattering simulation

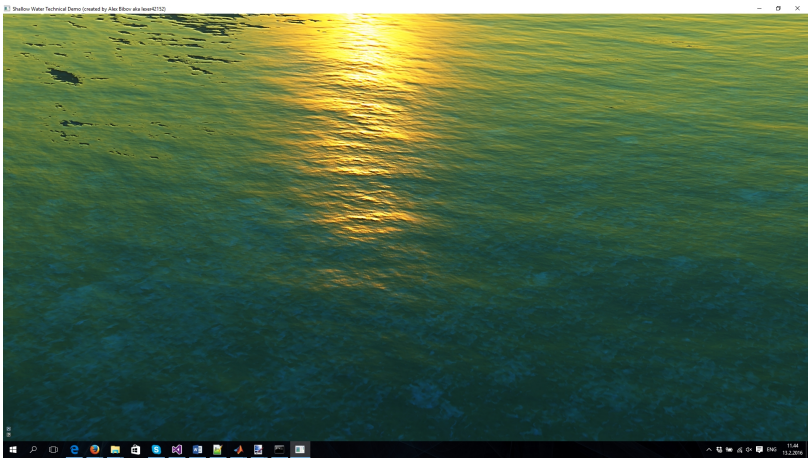

Figure 6.2: Shallow water and reflection/refraction models

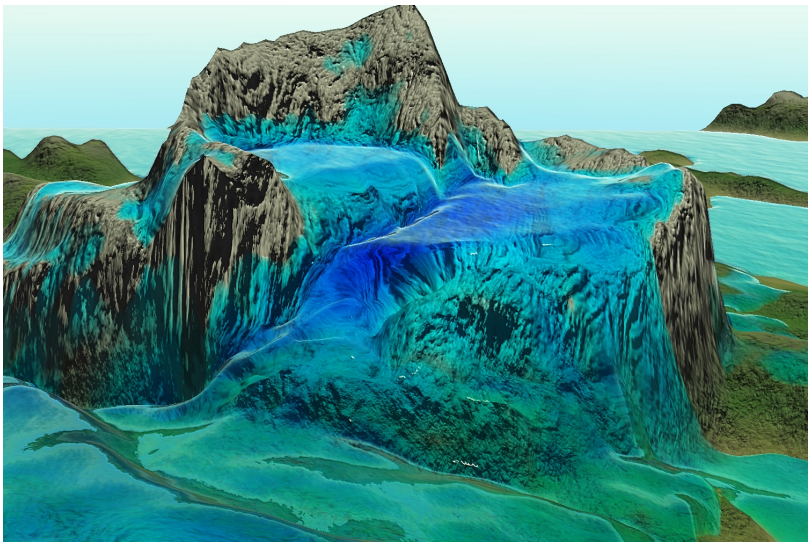

Figure 6.3: Shallow water simulation on irregular surface

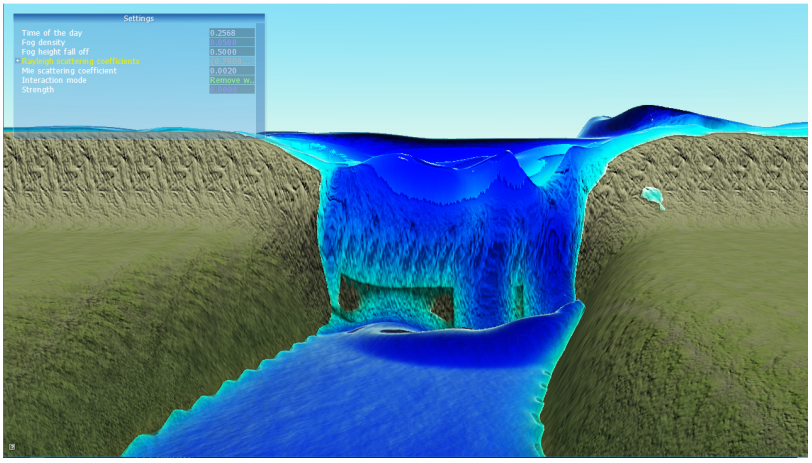

Figure 6.4: Dam break experiment

References

Albert, A. (1972). *Regression and the Moore-Penrose Pseudoinverse*, vol. 94. Academic Press.

Anderson, B. and Moore, J. (1979). *Optimal Filtering*. PRENTICE-HALL, INC.

Anderson, J.L. (2006). An adaptive covariance inflation error correction algorithm for ensemble filters. *Tellus-A*, 59, pp. 210–224.

Anderson, J. (2001). An ensemble adjustment Kalman filter for data assimilation. *Monthly Weather Review*, 129, pp. 2884–2903.

Apte, A., Jones, C., Stuart, A., and Voss, J. (2008). Data assimilation: Mathematical and statistical perspectives. *International Journal on Numerical methods in Fluids*, 56, pp. 1033–1046.

Auvinen, H., Bardsley, J., Haario, H., and Kauranne, T. (2009a). Large-Scale Kalman Filtering using the Limited Memory BFGS Method. *Electronic Transactions on Numerical Analysis*, 35, pp. 217–233.

Auvinen, H., Bardsley, J., Haario, H., and Kauranne, T. (2009b). The variational Kalman filter and an efficient implementation using limited memory BFGS. *International Journal on Numerical methods in Fluids*, 64, pp. 314–335.

Bardsley, J., Parker, A., Solonen, A., and Howard, M. (2011). Krylov space approximate Kalman filtering. *Numerical Linear Algebra with Applications*, n/a, pp. n/a–n/a. doi: 10.1002/nla.805.

Bardsley, J., et al. (2013). An ensemble Kalman filter using the conjugate gradient sampler. *International Journal for Uncertainty Quantification*, 3(4), pp. 357–370. doi: 10.1615/Int.J.UncertaintyQuantification.2012003889.

Barth, A., et al. (2010). Ensemble perturbation smoother for optimizing tidal boundary conditions by assimilation of High-Frequency radar surface currents — application to the German Bight. *Ocean Science*, 6(1), pp. 161–178. doi:10.5194/os-6-161-2010.

Ben-Israel, A. (1966). A Note on Iterative Method for Generalized Inversion of Matrices. *Math. Computation*, 20, pp. 439–440.

Bengtsson, T., Snyder, C., and Nychka, D. (2003). Towards a nonlinear ensemble filter for high-dimensional systems. *Journal of Geophysical Research*, 108, p. 8775. doi: 10.1029/2002JD002900.

Bibov, A. and Haario, H. (2016). Parallel Implementation of Data Assimilation. *International Journal for Numerical Methods in Fluids*, n/a, p. n/a. doi:DOI:10.1002/fld.4278.

Bibov, A., Haario, H., and Solonen, A. (2015). Stabilized BFGS Approximate Kalman Filter. *Inverse Problems and Imaging*, 9(4), pp. 1003–1024. doi:doi:10.3934/ipi.2015.9.1003.

Broyden, C. (1970). The Convergence of a Class of Double-rank Minimization Algorithms 1. General Considerations. *IMA Journal of Applied Mathematics*, 6(1), pp. 76–90. doi:10.1093/imamat/6.1.76.

Byrd, R., Nocedal, J., and Schnabel, R. (1994). Representations of Quasi-Newton Matrices and Their Use in Limited Memory Methods. *Mathematical Programming*, 63, pp. 129–156.

Cardinali, C. (2013). *Observation influence diagnostic of a data assimilation system*. url: `http://www.ecmwf.int/sites/default/files/ObservationInfluence.pdf`.

Dee, D. (1990). Simplification of the Kalman filter for meteorological data assimilation. *Quarterly Journal of the Royal Meteorological Society*, 117(498), pp. 365–384. doi:10.1002/qj.49711749806.

Dennis, J. and Moré, J. (1977). Quasi-Newton methods, motivation and theory. *SIAM Review*, 19, pp. 46–89.

Dennis, J. and Schnabel, R. (1979). Least change secant updates for quasi-Newton methods. *SIAM Review*, 21, pp. 443–459.

Dennis, J. and Schnabel, R. (1980). *A New Derivation of Symmetric Positive Definite Secant Updates*. University of Colorado at Boulder. Department of Computer Science.

Evans, L. (1997). *Partial Differential Equations*, chap. Laplace's equation, pp. 20–43. Providence: American Mathematical Society.

Evensen, G. (1994). Sequential data assimilation with a non-linear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research*, 99(C5), pp. 10,143–10,162. doi:10.1029/94JC00572.

Evensen, G. (2004). Sampling strategies and square root analysis schemes for the EnKF. *Ocean Dynamics*, 54(6), pp. 539–560. doi:10.1007/s10236-004-0099-2.

Fandry, C. and Leslie, L. (1984). A Two-Layer Quasi-Geostrophic Model of Summer Trough Formation in the Australian Subtropical Easterlies. *Journal of the Atmospheric Sciences*, 41, pp. 807–818.

Farrel, P.E., Ham, D.A., Funke, S.W., and Rognes, M.E. (2013). Automated Derivation of Adjoint of High-Level Transient Finite Element Programs. *Software and High-Performance Computing*, 35(4), pp. 369–393. doi:10.1137/120873558.

Fisher, M. (2009). *An Investigation of Model Error in a Quasi-Geostrophic, Weak-Constraint, 4D-Var Analysis System*. Oral presentation. ECMWF.

Fisher, M. and Adresson, E. (2001). *Developments in 4D-var and Kalman filtering*. ECMWF Technical Memorandum. 347. ECMWF.

Fisher, M., Leutbecker, M., and Kelly, G. (2005). On the Equivalence between Kalman smoothing and weak-constraint four-dimensional variational data assimilation. *Quarterly Journal of the Royal Meteorological Society*, 131(613), pp. 3235–3246. doi: 10.1256/qj.04.142.

Fisher, M., et al. (2011). *Weak-Constraint and Long-Window 4D-Var*. Technical report. ECMWF. url: `http://www.ecmwf.int/sites/default/files/elibrary/2011/9414-weak-constraint-and-long-window-4dvar.pdf`.

Gaspari, G. and S.E., C. (1999). Construction of correlation functions in two and three dimensions. *Quarterly Journal of the Royal Meteorological Society*, 125(554), pp. 723–757.

Heimbach, P., Hill, C., and Giering, R. (2002). Automatic Generation of Efficient Adjoint Code for a Parallel Navier-Stokes Solver. *Lecture Notes in Computer Science LNCS*, pp. 1019–1028. doi:10.1007/3-540-46080-2_107.

Houtekamer, P. and Mitchell, H. (1998). Data assimilation using an ensemble Kalman filter technique. *Monthly Weather Review*, 126, pp. 796–811. doi:10.1175/1520-0493(1998)126⟨0796:DAUAEK⟩2.0.CO;2.

Houtekamer, P., He, B., and L.M., H. (2014). Parallel Implementation of an Ensemble Kalman Filter. *Monthly Weather Review*, 142, pp. 1163–1182. doi:DOI:10.1175/MWR-D-13-00011.1.

Kalman, R. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME - Journal of Basic Engineering*, 82, pp. 35–45.

Kalnay, E. (2007). *Atmospheric modeling, Data Assimilation and Predictability*, chap. Data assimilation: determination of the initial conditions for the computer forecasts, pp. 12–17. Cambridge: Cambridge University Press.

Kurganov, A. and Petrova, G. (2007). A Second-Order Well-Balanced Positivity Preserving Central-Upwind Scheme For The Saint-Venant System. *Communications in Mathematical Sciences*, 5(1), pp. 133–160.

Leveque, R. (2007). *Finite Difference Methods for Ordinary and Partial Differential Equations*, chap. Elliptic Equations, pp. 59–68. Seattle: SIAM.

Magnus, R. and Stiefel, E. (1952). Methods of Conjugate Gradients for Solving Linear Systems. *J. Res. Natl. Bur. Stand.*, 49, pp. 409–436.

Marsden, J. and Weinstein, A. (1985). *CALCULUS III*, chap. Partial Differentiation, pp. 779–791. New York: Springer-Verlag.

Moore, J. (1973). Discrete-Time Fixed-Lag Smoothing Algorithms. *Automatica*, 9, pp. 163–173.

Nocedal, J. and Wright, S. (1999a). *Numerical Optimization*, chap. Limited-Memory BFGS, pp. 224–227. New York: Springer-Verlag.

Nocedal, J. and Wright, S. (1999b). *Numerical Optimization*. New York: Springer-Verlag.

Pan, V. and Schreiber, R. (1991). An Improved Newton Iteration for the Generalized Inverse of a Matrix, with Applications. *SIAM Journal on Scientific and Statistical Computing*, 12, pp. 1109–1131.

Pedlosky, J. (1987). *Geophysical Fluid Dynamics*, chap. Geostrophic Motion, pp. 22–57. New York: Springer-Verlag.

Rauch, H., Striebel, C., and Tung, F. (1965). Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, 3(8), pp. 1445–1450. doi:10.2514/3.3166.

Riley, K., Hobson, M., and Bence, S. (2004). *Mathematical methods for physics and engineering*, chap. Partial differential equations: separation of variables and other methods, pp. 671–676. Cambridge: Cambridge University Press.

Simon, D. (2006). *Optimal state estimation, Kalman, H_∞, and nonlinear approaches*, chap. The Extended Kalman filter, pp. 400–403. Hoboken: WILEY-INTERSCIENCE.

Solonen, A., Bibov, A., Bardsley, J., and Haario, H. (2013). Optimization-Based Sampling in Ensemble Kalman Filtering. *International Journal for Uncertainty Quantification*, 4, pp. 349–364. doi:10.1615/Int.J.UncertaintyQuantification.2014007658.

Solonen, A., et al. (2014). Estimating model error covariance matrix parameters in extended Kalman filtering. *Nonlinear Processes in Geophysics*, 21, pp. 919–927. doi: doi:10.5194/npg-21-919-2014.

Staniforth, A. and Côté, J. (1991). Semi-lagrangian integration schemes for atmospheric models review. *Monthly Weather Review*, 119, pp. 2206–2223.

SYKE (2016). *Itämeren tosiaikainen leväseuranta (Alg@line)*. url: `http://www.syke.fi/hankkeet/algaline`.

Trémolet, Y. (2006). Accounting for an imperfect model in 4D-Var. *Quarterly Journal of the Royal Meteorological Society*, 132(621), pp. 2483–2504. doi:10.1256/qj.05.224.

Trémolet, Y. (2007). Incremental 4D-Var convergence study. *Tellus*, 59A, pp. 706–718.

Zupanski, D. (1996). A General Weak Constraint Applicable to Operational 4DVAR Data Assimilation Systems. *Monthly Weather Review*, 125, pp. 2274–2292.

Publication I

Bibov, A., Haario, H., and Solonen, A. (2015). Stabilized Approximate Kalman Filter. *Inverse Problems and Imaging*, 9(4), pp. 1003-1024.

Inverse Problems and Imaging
Volume 9, No. 4, 2015, 1003–1024
doi:10.3934/ipi.2015.9.1003

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.

2010 *Mathematics Subject Classification.* Primary: 60G35, 93E11; Secondary: 62M20.

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], [33], [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 [34] and later suggested as a replacement for the 4D-VAR, the dominating method for data assimilation in weather prediction [31], [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 [29]). 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 [26].

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:

$$x_{k+1} = \mathcal{M}_k(x_k) + \epsilon_k, \tag{1}$$

$$y_{k+1} = \mathcal{H}_{k+1}(x_{k+1}) + \eta_{k+1}. \tag{2}$$

In the first equation x_k denotes $n\times 1$ state vector at time instance k, $\mathcal{M}_k$ is transition operator from time instance k to time instance $k+1$, ϵ_k is a random $n \times 1$ vector representing modelling error of the transition operator $\mathcal{M}_k$. In the second equation y_{k+1} denotes $m \times 1$ observed vector at time instance $k+1$, $\mathcal{H}_{k+1}$ is observation operator, and η_{k+1} is an $m \times 1$ random vector representing the observation error.

From now on we assume that the error terms ϵ_k and η_{k+1} are mutually independent, with zero mean and covariance matrices equal C_{ϵ_k} and C_{η_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^{est}_{k+1} given the estimate x^{est}_k of x_k together with its analysis covariance C^{est}_k, observed data y_{k+1}, and covariance matrices C_{ϵ_k} and C_{η_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 $\mathcal{M}_k$ and $\mathcal{H}_k$ 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^{est}_k, C^{est}_k, y_{k+1}, C_{ϵ_k}, $C_{\eta_{k+1}}$.
OUTPUT: x^{est}_{k+1}, C^{est}_{k+1}.

(i) Prediction step
  a) Compute state prediction $x^p_{k+1} = M_k x^{est}_k$;
  b) Compute covariance matrix for predicted state

$$C^p_{k+1} = M_k C^{est}_k M^T_k + C_{\epsilon_k}.$$

(ii) Correction step
  a) Compute the Kalman gain

$$G_{k+1} = C^p_{k+1} H^T_{k+1} \left(H_{k+1} C^p_{k+1} H^T_{k+1} + C_{\eta_{k+1}}\right)^{-1};$$

  b) Compute the next state estimate

$$x^{est}_{k+1} = x^p_{k+1} + G_{k+1}\left(y_{k+1} - H_{k+1} x^p_{k+1}\right);$$

  c) Compute covariance of the next state estimate

$$C^{est}_{k+1} = C^p_{k+1} - G_{k+1} H_{k+1} C^p_{k+1}.$$

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 $\mathcal{H}_k$:

$$M_k = \frac{\partial \mathcal{M}_k(x_k^{est})}{\partial x}, \; H_k = \frac{\partial \mathcal{H}_k(x_k^{est})}{\partial x}. \tag{3}$$

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_k^T and H_k^T 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:

$$\operatorname*{argmin}_x \frac{1}{2}(Ax, x) - (b, x), \tag{4}$$

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 "$\breve{}$" on top of the regular matrix notation introduced in algorithm I. For instance, an approximation of the estimate covariance matrix C_k^{est} introduced in algorithm I is denoted by $\breve{C}_k^{est}$, 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_k^{est}, $\breve{C}_k^{est}$, y_{k+1}, C_{ϵ_k}, $C_{\eta_{k+1}}$.
OUTPUT: x_{k+1}^{est}, $\breve{C}_{k+1}^{est}$.

(i) Prediction step

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

b) On the code level define prior covariance operator

$$C_{k+1}^p = M_k^{TL} \breve{C}_k^{est} M_k^{AD} + C_{\epsilon_k}.$$

(ii) Correction step

a) Define optimization problem of type (4) assuming

$$A = H^{TL}_{k+1} C^{p}_{k+1} H^{AD}_{k+1} + C_{\eta_{k+1}},\ b = y_{k+1} - H^{TL}_{k+1} x^{p}_{k+1}.$$

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 $x^{est}_{k+1} = x^{p}_{k+1} + C^{p}_{k+1} H^{AD}_{k+1} x^*$.

c) Define optimization problem of type (4) assuming

$$A = C^{p}_{k+1} - C^{p}_{k+1} H^{AD}_{k+1} B^* H^{TL}_{k+1} C^{p}_{k+1},\ b = 0.$$

Update estimate covariance matrix by assigning $\breve{C}^{est}_{k+1}$ 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 C^{est}_{k+1}. Furthermore, this error migrates to the next iteration of the filter and potentially affects the prior covariance in step (i. b). Thence, 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 $A = H^{TL}_{k+1} C^{p}_{k+1} H^{AD}_{k+1} + C_{\eta_{k+1}}$, where C^{p}_{k+1} is computed in accordance with step (i. b) of algorithm II and $\breve{C}^{est}_{k}$ is assumed to be nonnegative definite. Then for an arbitrary symmetric matrix B^* the matrix*

$$\breve{C}^{est}_{k+1} = C^{p}_{k+1} - C^{p}_{k+1} H^{AD}_{k+1} (2I - B^* A) B^* H^{TL}_{k+1} C^{p}_{k+1}. \tag{5}$$

is non-negative definite and $\breve{C}^{est}_{k+1}$ is equal to C^{est}_{k+1} when $B^* = A^{-1}$ (see algorithm I, step (ii. c)). Here I denotes the identity matrix of respective dimension.

Proof. For conciseness we denote $C = C^p_{k+1}$, $H = H^{TL}_{k+1}$, $R = C_{\eta_{k+1}}$, $G = CH^TB^*$. The non-negative definiteness of $\breve{C}^{est}_{k+1}$ is proved by the following sequence of obvious matrix equalities:

$$\begin{aligned}
\breve{C}^{est}_{k+1} &= C - CH^T\left(2I - B^*A\right)B^*HC = \\
&= C - CH^T\left(B^* - (B^*A - I)B^*\right)HC = \\
&= C - CH^T\left(B^*HC - (B^*A - I)B^*HC\right) = \\
&= C - CH^TB^*HC + CH^T(B^*A - I)B^*HC = \\
&= C - GHC + CH^T(B^*A - I)G^T = \\
&= C - GHC + \left(CH^TB^*\left(HCH^T + R\right) - CH^T\right)G^T = \\
&= C - GHC + \left(GHCH^T + GR - CH^T\right)G^T = \\
&= C - GHC - CH^TG^T + GHCH^TG^T + GRG^T = \\
&= (I - GH)C(I - GH)^T + GRG^T.
\end{aligned}$$

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

The identity $\breve{C}^{est}_{k+1} = C^{est}_{k+1}$ 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 [25] 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 $\breve{C}^{est}_{k+1}$ 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^{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^{est}_{k+1} as B^* approaches A^{-1}. Furthermore, the following inequalities hold:*

$$\|\breve{C}^{est}_{k+1} - C^{est}_{k+1}\|_{Fr} \le \|A\|\|H^{TL}_{k+1}C^p_{k+1}\|^2_{Fr}\|B^* - A^{-1}\|^2. \tag{6}$$

$$\|\breve{C}^{est}_{k+1} - C^{est}_{k+1}\| \le \|A\|\|H^{TL}_{k+1}C^p_{k+1}\|^2\|B^* - A^{-1}\|^2. \tag{7}$$

Here $\|\cdot\|_{Fr}$ 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 $\|W^TPW\|_{Fr} \le \|P\|\|W\|^2_{Fr}$. Indeed, it is easy to verify

that $\|W^T PW\|_{Fr}^2 = \sum_{i,j} \left|\left(PW^i, W^j\right)\right|^2$, 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

$$\begin{aligned}\|W^T PW\|_{Fr}^2 &\leq \sum_{i,j} \|PW^i\|_{Fr}^2 \|W^j\|_{Fr}^2 \leq \\ &\leq \sum_{i,j} \|P\|^2 \|W^i\|_{Fr}^2 \|W^j\|_{Fr}^2 \\ &= \|P\|^2 \|W\|_{Fr}^4.\end{aligned}$$

Therefore, we proved that $\|W^T PW\|_{Fr}^2 \leq \|P\|^2 \|W\|_{Fr}^4$ 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 $\check{C}_{k+1}^{est}$ and C_{k+1}^{est}.

$$\|\check{C}_{k+1}^{est} - C_{k+1}^{est}\|_{Fr} = \|C_{k+1}^p H_{k+1}^{AD}(2B^* - B^*AB^* - A^{-1})H_{k+1}^{TL} C_{k+1}^p\|$$

Recalling the auxiliary inequality $\|W^T PW\|_{Fr} \leq \|P\|\|W\|_{Fr}^2$ and assigning $P = 2B^* - B^*AB^* - A^{-1}$, $W = H_{k+1}^{TL} C_{k+1}^p$ we arrive to the following estimate:

$$\|\check{C}_{k+1}^{est} - C_{k+1}^{est}\|_{Fr} \leq \|H_{k+1}^{TL} C_{k+1}^p\|_{Fr}^2 \|2B^* - B^*AB^* - A^{-1}\|. \tag{8}$$

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

$$\|2B^* - B^*AB^* - A^{-1}\| = \|\Delta A \Delta\| \leq \|A\|\|\Delta\|^2 = \|A\|\|B^* - A^{-1}\|^2. \tag{9}$$

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 $\check{C}_{k+1}^{est}$ in the step (ii. c) of algorithm II converges to C_{k+1}^{est} when B^* approaches A^{-1}. Furthermore, the following inequalities hold:*

$$\|\check{C}_{k+1}^{est} - C_{k+1}^{est}\|_{Fr} \leq \|H_{k+1}^{TL} C_{k+1}^p\|_{Fr}^2 \|B^* - A^{-1}\|. \tag{10}$$

$$\|\check{C}_{k+1}^{est} - C_{k+1}^{est}\| \leq \|H_{k+1}^{TL} C_{k+1}^p\|^2 \|B^* - A^{-1}\|. \tag{11}$$

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}^{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:

$$\mathcal{F}_n(X) = 2\mathcal{F}_{n-1}(X) - \mathcal{F}_{n-1}(X) A \mathcal{F}_{n-1}(X), \tag{12}$$

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. $\mathcal{F}_n\left(A^{-1}\right) = A^{-1}$, $n \geq 1$ 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 shows that the matrix-valued function sequence $\mathcal{F}_n(X)$ provides an easy way to improve approximation for A^{-1} preserving the advantageous properties of the estimate given in (5).

Lemma 2.4. *Suppose that assumptions made in lemma 2.1 are met. Let $\mathcal{F}_n(X)$ be a matrix-valued function sequence generated by the recurrent relation (12). Then for any symmetric X the following matrix is nonnegative definite:*

$$\breve{C}^{est}_{k+1}(n) = C^p_{k+1} - C^p_{k+1} H^{AD}_{k+1} \mathcal{F}_n(X)\, H^{TL}_{k+1} C^p_{k+1},\ n \geq 2. \tag{13}$$

Moreover, the series of inequalities hold:

$$\|\mathcal{F}_n(X) - A^{-1}\| \leq \|A^{-1}\| \|AX - I\|^{2^{n-1}}, \tag{14}$$

$$\|C^{est}_{k+1} - \breve{C}^{est}_{k+1}(n)\|_{Fr} \leq \|A\| \|A^{-1}\| \|H^{TL}_{k+1} C^p_{k+1}\|^2_{Fr} \|AX - I\|^{2^{n-1}}, \tag{15}$$

$$\|C^{est}_{k+1} - \breve{C}^{est}_{k+1}(n)\| \leq \|A\| \|A^{-1}\| \|H^{TL}_{k+1} C^p_{k+1}\|^2 \|AX - I\|^{2^{n-1}}. \tag{16}$$

Proof. Before delving into the details of the proof we note that $\breve{C}^{est}_{k+1}(n)$ 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 $\breve{C}^{est}_{k+1}(n)$ is nonnegative definite. Foremost, we note that as far as X is symmetric, the matrix $\mathcal{F}_n(X)$ is also symmetric for all $n \geq 1$. Then, accounting for the claim of lemma 2.1 we conclude that $\breve{C}^{est}_{k+1}(n)$ is nonnegative definite when $n \geq 2$.

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

$$\mathcal{F}_n(X) = A^{-1} - \Delta \left(A\Delta\right)^{2^{n-1}-1}, \tag{17}$$

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

$$\begin{aligned}
\mathcal{F}_{n_0+1}(X) &= 2\mathcal{F}_{n_0}(X) - \mathcal{F}_{n_0}(X) A \mathcal{F}_{n_0}(X) = \\
&= 2A^{-1} - 2\Delta \left(A\Delta\right)^{2^{n_0-1}-1} - \\
&- \left(A^{-1} - \Delta \left(A\Delta\right)^{2^{n_0-1}-1}\right) A \left(A^{-1} - \Delta \left(A\Delta\right)^{2^{n_0-1}-1}\right) = \\
&= A^{-1} - \Delta \left(A\Delta\right)^{2^{n_0-1}-1} A\Delta \left(A\Delta\right)^{2^{n_0-1}-1} = A^{-1} - \Delta \left(A\Delta\right)^{2^{n_0}-1}.
\end{aligned}$$

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

$$\begin{aligned}
&\|\mathcal{F}_n(X) - A^{-1}\| = \|\Delta \left(A\Delta\right)^{2^{n-1}-1}\| = \|\left(A^{-1} - X\right)\left(I - AX\right)^{2^{n-1}-1}\| = \\
&\|A^{-1}\left(I - AX\right)^{2^{n-1}}\| \leq \|A^{-1}\| \|AX - I\|^{2^{n-1}}.
\end{aligned}$$

Let us assume that $\|AX - I\| < 1$. This implies $\mathcal{F}_n(X) \underset{n\to\infty}{\longrightarrow} A^{-1}$, where the rate of convergence is estimated by an infinitesimal $O\left(\|AX - I\|^{2^n}\right)$.

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. □

In essence, sequence (12) corresponds to a matrix inversion formula of Newton-Schultz type. As follows from Lemma 2.4 $\mathcal{F}_n$ 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 $\mathcal{F}_n$ converges to Moore pseudoinverse A^+ as shown in [6]. In addition, properties of Chebyshev polynomials allow to generalize quadratic sequence $\mathcal{F}_n$ by cubic recurrence formula, which provides accelerated convergence rate (see [26] for details). For the case of symmetric A it was proved in [26] that the initializer $B^* = (1/\|A\|_{Fr})\, I$ guarantees that $\|AB^* - I\| \leq 1 - 1/\left(m^{1/2}k\right)$, 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 [27]. 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], [32].

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 I. 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:

$$q_1 = \nabla^2\psi_1 - F_1\left(\psi_1 - \psi_2\right) + \beta y, \tag{18}$$

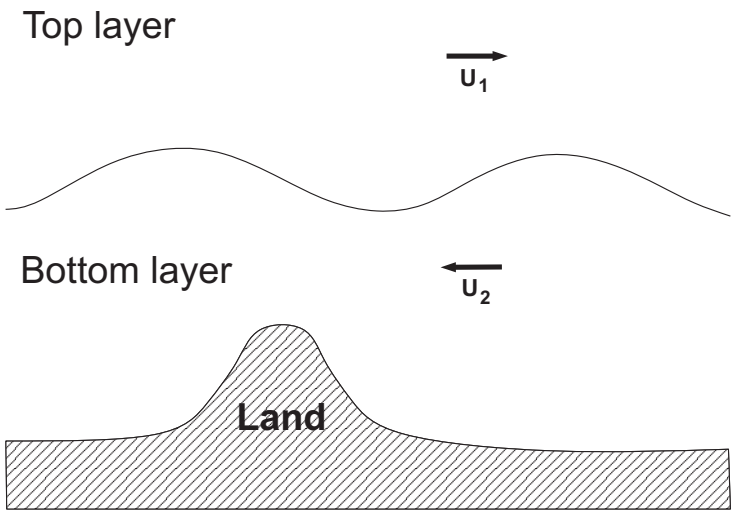

Figure I. Geometric topology of the two-layer quasi-geostrophic model

$$q_2 = \nabla^2\psi_2 - F_2\left(\psi_2 - \psi_1\right) + \beta y + R_s. \tag{19}$$

Here q_i and ψ_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 ψ_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:

$$\frac{D_1 q_1}{Dt} = 0, \ \frac{D_2 q_2}{Dt} = 0. \tag{20}$$

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

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:

$$F_1 = \frac{f_0^2 L^2}{\acute{g} D_1}, \ F_2 = \frac{f_0^2 L^2}{\acute{g} D_2}, \tag{21}$$

$$\acute{g} = g\frac{\Delta\theta}{\bar{\theta}}, \tag{22}$$

$$R_s = \frac{S\left(x, y\right)}{\eta D_2}, \tag{23}$$

$$\beta = \beta_0 \frac{L}{U}, \tag{24}$$

where L and U are the length and speed main scales, β_0 is the northward gradient of the Coriolis parameter, g is the gravity acceleration, $\Delta\theta$ is temperature change across the layer interface, $\bar{\theta}$ 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 ∇^2 to (18) and subtracting the sum of F_1 times (19) and F_2 times (18) from the result we arrive at:

$$\begin{aligned} &\nabla^2\left[\nabla^2\psi_1\right] - (F_1+F_2)\left[\nabla^2\psi_1\right] = \\ &= \nabla^2 q_1(t_0) - F_2\left(q_1(t_0) - \beta y\right) - F_1\left(q_2(t_0) - \beta y - R_s\right). \end{aligned} \tag{25}$$

Equation (25) is nonhomogeneous Helmholtz equation with negative constant parameter $-(F_1+F_2)$, where $\nabla^2\psi_1$ is unknown [28]. 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 ψ_1 by solving the corresponding Poisson problem [17]. The solution is obtained by finite-difference method [24]. The value for ψ_2 can be deduced from (18) and (19) by substituting the obtained value for ψ_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 [30].

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 $\mathcal{M}_k$ 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 matrixes.** The prediction and observation uncertainties were modelled by normally distributed random variables ϵ_k, η_{k+1} (see equations (1) and (2)) with covariance matrices defined as follows:

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

where σ^2 is the variance of model uncertainty, α is correlation coefficient between the layers, ξ^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 α and σ were estimated from model simulations. The value of ξ^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: $f_0 = 10^{-4}$, $\frac{\Delta\theta}{\theta} = 0.1$, $\beta_0 = 1.5 \cdot 10^{-11}$. 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_0^{est}. The uncertainty of this estimate was modeled "naively" by identity covariance, i.e. $C_0^{est} = I$.

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 ψ_1 and ψ_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:

$$\mathcal{R}\left(x^{est}\right) = \frac{\|x^{truth} - x^{est}\|}{\sqrt{N}}. \tag{26}$$

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

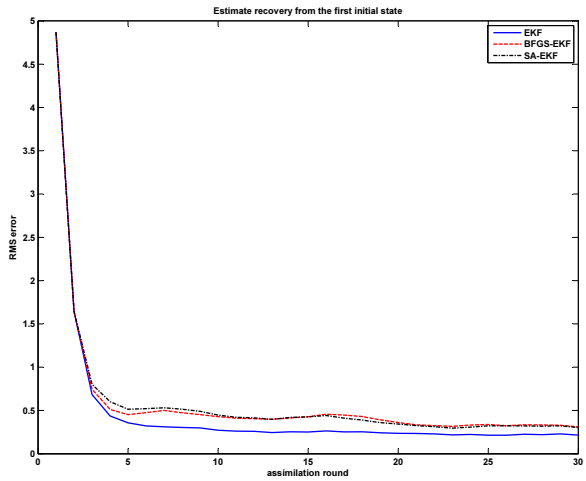

Figure II. Filter recovery from the errors in the first initial estimate.

Table I. Benchmark options depending on model resolution.

No.	Truth run res.	Biased run res.	Obs. num.	State dim.
I	10-by-10	9-by-9	50	162
II	15-by-15	12-by-12	85	288
III	80-by-40	40-by-20	500	1600

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

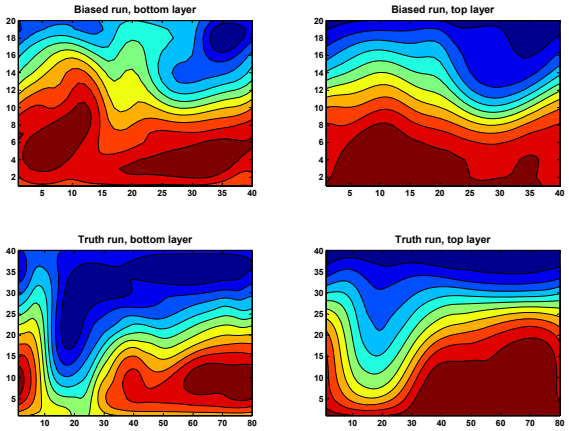

FIGURE III. Truth and biased run states after 10 days of model time simulation. X-, and Y- axes correspond to longitude and latitude respectively. The isolines describe values of the stream function ψ_i with color mapping from blue to red.

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

		Benchmark					
		I		II		III	
		BFGS-EKF	SA-EKF	BFGS-EKF	SA-EKF	BFGS-EKF	SA-EKF
BFGS vectors	5	0.5860	0.5508	–	0.6476	–	0.4102*
	10	0.5573	0.5445	0.7021	0.6718	–	0.4412
	15	0.4959	0.4788	0.5807	0.5690	–	0.3815
	20	0.4686	0.4695	0.5195	0.5152	0.3686	0.3656
EKF ref.		0.4247		0.4346		0.2694	

marked by "–" correspond to experiments, where the filter did not converge due to some of the eigenvalues of the estimate covariance being negative. For instance, the estimates diverged from the truth when L-BFGS-EKF was run having BFGS storage capacity of 5 iterations. In addition, numerical values marked by symbol "*" emphasize experiments that used inflation in the forecast covariance matrix. The inflation factor was equal to $\frac{1}{\xi^2\|C_k^p\|}$, where C_k^p is the forecast covariance matrix. This trick is heuristic and was employed to prevent spurious growth of magnitude in the estimated eigenvalues, which happens in case of high uncertainty modeled by coarse approximations. The accuracy of the estimates might be further improved with more judicious choice of the inflation factor [1]. Nevertheless, in our benchmarks the given choice was enough for the filter to recover from the strongly biased first initial value. It should also be emphasized, that we employed covariance inflation only in those experiments, which did not converge otherwise. In the experiments

TABLE III. Expected forecast length with respect to varied capacity of the BFGS storage.

		Benchmark					
		I		II		III	
		BFGS-EKF	SA-EKF	BFGS-EKF	SA-EKF	BFGS-EKF	SA-EKF
BFGS vectors	5	78	78	–	54	–	42*
	10	84	84	54	54	–	42
	15	84	84	60	60	–	48
	20	84	84	60	60	48	48
EKF ref.		90		66		48	

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:

$$\operatorname*{argmin}_{D} \|D\|_{Fr} : \ (A + D)s = y. \tag{27}$$

Lemma 6.1. *The task (27) is solved by* $D = \frac{y-As}{s^T s} s^T$.

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

$$\|D\|_{Fr} = \mathbf{tr}\left(D^T D\right),$$

where $\mathbf{tr}(\cdot)$ denotes the trace defined as the sum of matrix diagonal elements. Hence, in accordance with the matrix spectral theorem $\|D\|_{Fr} = \mathbf{tr}\left(PLP^T\right)$, 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, $s^T D^T D s = r^T r$. 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.

$$\|D\|_{Fr} = \sum_i \lambda_i \to min, \tag{28}$$

$$\sum_i \lambda_i \tilde{s}_i^2 = r^T r, \tag{29}$$

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 λ_i). Hence, from the constraint equation (29) we obtain

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

Substituting the expression for λ_1 into minimization problem (28) we eliminate the constraint (29).

$$\frac{r^T r}{\tilde{s}_1^2} + \sum_{i \neq 1} \lambda_i \left(1 - \frac{\tilde{s}_i^2}{\tilde{s}_1^2}\right) \to min. \tag{30}$$

Obviously, the sum in (30) is non-negatve 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, λ_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 $D^T D = s \frac{r^T r}{\|s\|^4} s^T$, which leads to the obvious solution $D = \frac{y-As}{s^T s} s^T$. 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 [25], 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 $\tilde{B} = \tilde{J}\tilde{J}^T$ such that $\tilde{B}s = y$ if and only if $y^T s > 0$. If the latter requirement is satisfied, then*

$$\tilde{J} = J + \frac{\left(y - \sqrt{\frac{y^T s}{s^T Bs}}Bs\right)\left(J^T s\right)^T}{\sqrt{\frac{y^T s}{s^T Bs}}s^T Bs}$$

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 $z = L^T s$. Hence, $z^T z = \left(s^T L\right)\left(L^{-1}y\right) = s^T y > 0$.

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

$$\tilde{J} = J + \frac{y - Jz}{z^T z}z^T. \tag{31}$$

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

$$\tilde{J}^T s = J^T s + \left(\frac{y^T - z^T J^T}{z^T z}s\right)z = z.$$

This implies that $z = \alpha J^T s$. Plugging the expression for z back into the latter equation, we obtain

$$\alpha = 1 + \frac{y^T s - \alpha s^T Bs}{\alpha s^T Bs}.$$

Hence, $\alpha^2 = (y^T s)/(s^T Bs)$ and as far as $y^T s > 0$ and B is positive definite we can get a real value for α (it is not important whether the chosen value for α is positive or negative). Therefore $z = \sqrt{(y^T s)/(s^T Bs)}J^T s$. Substitution of the expression for z into (31) yields the desired formula for $\tilde{J}$. □

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 $\tilde{B} = \tilde{J}\tilde{J}^T$. Skipping the onerous calculations the substitution yields:

$$\tilde{B} = B + \frac{yy^T}{y^T s} - \frac{Bss^T B}{s^T Bs}. \tag{32}$$

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

$$\tilde{B}^{-1} = \frac{ss^T}{y^T s} + \left(I - \frac{sy^T}{y^T s}\right) B^{-1} \left(I - \frac{ys^T}{y^T s}\right). \tag{33}$$

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:

$$\underset{\tilde{B}:\tilde{B}>0,\ \tilde{B}^{-1}y=s}{\operatorname{argmin}} \|B^{-1} - \tilde{B}^{-1}\|_{Fr},$$

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 [25].

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 $h_0 I$, 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 $H_0 = h_0 I$.
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, $As_i = y_i$. 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 out 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

[1] J. L. Anderson, An adaptive covariance inflation error correction algorithm for ensemble filters, *Tellus-A*, **59** (2006), 210–224.
[2] H. Auvinen, J. Bardsley, H. Haario and T. Kauranne, Large-scale Kalman filtering using the limited memory BFGS method, *Electronic Transactions on Numerical Analysis*, **35** (2009), 217–233.
[3] H. Auvinen, J. Bardsley, H. Haario and T. Kauranne, The variational Kalman filter and an efficient implementation using limited memory BFGS, *International Journal on Numerical methods in Fluids*, **64** (2009), 314–335.
[4] J. Bardsley, A. Parker, A. Solonen and M. Howard, Krylov space approximate Kalman filtering, *Numerical Linear Algebra with Applications*, **20** (2013), 171–184.
[5] A. Barth, A. Alvera-Azcárate, K.-W. Gurgel, J. Staneva, A. Port, J.-M. Beckers and E. Stanev, Ensemble perturbation smoother for optimizing tidal boundary conditions by assimilation of high-frequency radar surface currents – application to the German bight, *Ocean Science*, **6** (2010), 161–178.
[6] A. Ben-Israel, A note on iterative method for generalized inversion of matrices, *Math. Computation*, **20** (1966), 439–440.
[7] G. J. Bierman, *Factorization Methods for Discrete Sequential Estimation*, Vol. 128, Academic Press, 1977.
[8] R. Bucy and P. Joseph, *Filtering for Stochastic Processes with Applications to Guidance*, John Wiley & Sons, New York, 1968.
[9] R. Byrd, J. Nocedal and R. Schnabel, Representations of quasi-Newton matrices and their use in limited memory methods, *Mathematical Programming*, **63** (1994), 129–156.
[10] M. Cane, A. Kaplan, R. Miller, B. Tang, E. Hackert and A. Busalacchi, Mapping tropical pacific sea level: Data assimilation via reduced state Kalman filter, *Journal of Geophysical Research*, **101** (1996), 22599–22617.
[11] L. Canino, J. Ottusch, M. Stalzer, J. Visher and S. Wandzura, Numerical solution of the Helmholtz equation in 2d and 3d using a high-order Nyström discretization, *Journal of Computational Physics*, **146** (1998), 627–663.
[12] J. L. Crassidis and J. L. Junkins, *Optimal Estimation of Dynamic Systems*, 2nd edition, CRC Press, 2012.
[13] D. Dee, Simplification of the Kalman filter for meteorological data assimilation, *Quarterly Journal of the Royal Meteorological Society*, **117** (1991), 365–384.
[14] J. Dennis and J. Moré, Quasi-Newton methods, motivation and theory, *SIAM Review*, **19** (1977), 46–89.
[15] J. Dennis and R. Schnabel, Least change secant updates for quasi-Newton methods, *SIAM Review*, **21** (1979), 443–459.
[16] J. Dennis and R. Schnabel, A new derivation of symmetric positive definite secant updates, in *Nonlinear Programming* (Madison, Wis., 1980), 4, Academic Press, New York-London, 1981, 167–199.
[17] L. Evans, *Partial Differential Equations*, Graduate Studies in Mathematics, 19, American Mathematical Society, Providence, RI, 1998.
[18] G. Evensen, Sequential data assimilation with a non-linear quasi-geostrophic model using monte carlo methods to forecast error statistics, *Journal of Geophysical Research*, **99** (1994), 143–162.

[19] C. Fandry and L. Leslie, A two-layer quasi-geostrophic model of summer trough formation in the australian subtropical easterlies, *Journal of the Atmospheric Sciences*, **41** (1984), 807–818.
[20] M. Fisher, *Development of a Simplified Kalman Filter*, ECMWF Technical Memorandum, 260, ECMWF, 1998.
[21] M. Fisher, *An Investigation of Model Error in a Quasi-Geostrophic, Weak-Constraint, 4D-Var Analysis System*, Oral presentation, ECMWF, 2009.
[22] M. Fisher and E. Adresson, *Developments in 4D-var and Kalman Filtering*, ECMWF Technical Memorandum, 347, ECMWF, 2001.
[23] R. Kalman, A new approach to linear filtering and prediction problems, *Transactions of the ASME - Journal of Basic Engineering*, **82** (1960), 35–45.
[24] R. Leveque, *Finite Difference Methods for Ordinary and Partial Differential Equations. Steady-State and Time-Dependent Problems*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2007.
[25] J. Nocedal and S. Wright, *Numerical Optimization*, Springer-Verlag, New York, 1999.
[26] V. Pan and R. Schreiber, An improved newton iteration for the generalized inverse of a matrix, with applications, *SIAM Journal on Scientific and Statistical Computing*, **12** (1991), 1109–1130.
[27] J. Pedlosky, Geostrophic motion, in *Geophysical Fluid Dynamics*, Springer-Verlag, New York, 1987, 22–57.
[28] K. Riley, M. Hobson and S. Bence, Partial differential equations: Separation of variables and other methods, in *Mathematical Methods for Physics and Engineering*, Cambridge University Press, Cambridge, 2004, 671–676.
[29] D. Simon, The discrete-time Kalman filter, in *Optimal State Estimation, Kalman, H_∞, and Nonlinear Approaches*, Wiley-Interscience, Hoboken, 2006, 123–145.
[30] A. Staniforth and J. Côté, Semi-lagrangian integration schemes for atmospheric models review, *Monthly Weather Review*, **119** (1991), 2206–2223.
[31] Y. Trémolet, Incremental 4d-var convergence study, *Tellus*, **59A** (2007), 706–718.
[32] Y. Tremolet and A. Hofstadler, *OOPS as a common framework for Research and Operations*, Presentation 14th Workshop on meteorological operational systems, ECMWF, 2013.
[33] A. Voutilainen, T. Pyhälahti, K. Kallio, H. Haario and J. Kaipio, A filtering approach for estimating lake water quality from remote sensing data, *International Journal of Applied Earth Observation and Geoinformation*, **9** (2007), 50–64.
[34] D. Zupanski, A general weak constraint applicable to operational 4dvar data assimilation systems, *Monthly Weather Review*, **125** (1996), 2274–2292.

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

Publication II

Bibov, A. and Haario, H. (2016). Parallel implementation of data assimilation. *International Journal for Numerical Methods in Fluids*, pp. 1-17, 2016.

INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids (2016)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/fld.4278

Parallel implementation of data assimilation

Alexander Bibov[1,*,†] and Heikki Haario[1,2]

[1]*Lappeenranta University Of Technology, Skinnarilankatu 34, FI-53850 Lappeenranta, Finland*
[2]*Finnish Meteorological Institute, Erik Palmnin aukio 1, PO 503, FI-00560 Helsinki, Finland*

SUMMARY

Kalman filter is a sequential estimation scheme that combines predicted and observed data to reduce the uncertainty of the next prediction. Because of its sequential nature, the algorithm cannot be efficiently implemented on modern parallel compute hardware nor can it be practically implemented on large-scale dynamical systems because of memory issues. In this paper, we attempt to address pitfalls of the earlier low-memory approach described in and extend it for parallel implementation. First, we describe a low-memory method that enables one to pack covariance matrix data employed by the Kalman filter into a low-memory form by means of certain quasi-Newton approximation. Second, we derive parallel formulation of the filtering task, which allows to compute several filter iterations independently. Furthermore, this leads to an improvement of estimation quality as the method takes into account the cross-correlations between consequent system states. We experimentally demonstrate this improvement by comparing the suggested algorithm with the other data assimilation methods that can benefit from parallel implementation. Copyright © 2016 John Wiley & Sons, Ltd.

Received 17 December 2015; Revised 5 July 2016; Accepted 11 July 2016

KEY WORDS: dynamical systems; extended Kalman filter; parallelization; stabilized method; probabilistic method; error estimation

1. INTRODUCTION

The classical formulation of the Kalman filter employs linear prediction and observation operators. An extension to nonlinear cases can be obtained by straightforward linearization techniques, but the linearization is proved to be 'legal' only if certain conditions are satisfied [2]. This generalization is usually referred to as the extended Kalman filter (EKF). These algorithms require propagating the covariance information between consequent time points in a way that needs the storage of full covariance matrixes. Naturally, because of the memory issues and computational expenses, this approach becomes inefficient when dimension of the system increases. This problem is, in principle, solved by replacing the covariance information with a low-memory approximation. Different approximations have been proposed in numerous literature sources. For instance, Evensen in [3] suggests to sample covariance matrices directly using predictions from numerical simulation initialized by perturbed initial states. This is the basic idea behind various ensemble Kalman filter (EnKF) formulations, which can be naturally implemented in a parallel computing framework, as predictions used to sample the covariance data can be executed independently from each other [4].

Another approach to address the problem of covariance data storage is to define lower-dimensional subspace of the full system space and project covariance matrix onto this 'smaller' subspace. This idea is used in the family of reduced-rank Kalman filters and reduced-order Kalman filters (see [5–8] for details). However, the success of this approach depends on choice of the rank

*Correspondence to: Alexander Bibov, Lappeenranta University Of Technology, Skinnarilankatu 34, FI-53850 Lappeenranta, Finland.
†E-mail: aleksandr.bibov@lut.fi

reduction subspace, which should be defined appropriately depending on additional knowledge about the system. To avoid the problem of choice, one may also interpret the problematic matrices as Hessians related to the underlying quadratic optimization problems and use classical quasi-Newton-based or Krylov space-based optimization methods that are able to produce low-memory approximations of the Hessians (see, for example, [1, 9–11]).

A weakness of all the aforementioned EKF variants is that the filtration procedure they describe is necessarily sequential and thus cannot be implemented using the advantages provided by parallel computing hardware. On the other hand, the EnKF methods, while naturally implemented in parallel, may suffer from numerous problems like, for example, the ensemble degradation discussed in [12]; additionally, selection of the size of the ensemble sufficient for adequate approximation of covariance data suggests another challenging problem.

A number of endeavors to provide a parallelizable data assimilation solution free from the downsides of the EnKF have been undertaken in the literature. For example, in [13], authors describe a parallel-in-time gradient method for solving linear control problems. Another algorithm, suggested in [14], extends numerical solution of the strong-constraint 4D-VAR formulation effectively enabling possibilities for parallel implementation. Finally, the approach from [15] adds another level of parallelism into EnKF by considering several subsequent time points simultaneously and combining them into a single-estimation problem similarly to how it is performed in our parallel filtering task discussed later.

In this paper, we consider a parallel implementation for non-ensemble approximate Kalman filters. We formulate a parallel filtering task on top of a given data assimilation problem. This task boils down to several independently executed model prediction steps that have the ability to communicate with each other by sharing their covariance data, which, as we demonstrate later, eventually increases the quality of the estimates.

The data assimilation problem induced by the parallel filtering task has a dimension several times larger than the original system. This drawback can be however alleviated by using the low-memory approximate Kalman filters discussed earlier. We employ the approximation based on limited-memory Broyden–Fletcher–Goldfarb–Shanno optimizer (L-BFGS) [16]. This optimizer belongs to a family of unconstrained optimization methods of quasi-Newton type and allows to represent covariance matrices generated by the Kalman filter by a full-rank yet low-memory implicit approximations. This approach has been extensively described in [1] and is referred to as the L-BFGS EKF.

However, in [17], it was demonstrated that the L-BFGS EKF as originally formulated in [1] is vulnerable to accumulating errors, because of indefinite covariance approximations. In [17], a solution to circumvent this problem was proposed and stabilized approximate L-BFGS EKF free from the drawbacks of the method from [1] was derived. The proposed approach is based on a small 'stabilizing' correction introduced into original formulae of the L-BFGS EKF that ensures that the approximate covariance matrices remain at least positive semi-definite. This justifies the usage of the L-BFGS optimizer, which can only be applied if the target cost function is convex.

By using the stabilized approximate L-BFGS EKF, it is possible to further reduce the memory requirements of the algorithm in comparison with the original approximate filter of [1]. We use this fact to efficiently solve the parallel filtering task and perform data assimilation over several assimilation windows in a single pass, with the benefit of an improvement in the quality of the estimates. It is also worth mentioning that the proposed method can be employed for retrospective analysis with no extra computational cost, as the needed ingredients are produced along with the standard output of our algorithm.

As a 'toy-case' for numerical benchmarking, we selected the two-layer quasi-geostrophic model (QG-model) introduced in [18]. This model is a simple simulator of quasi-geostrophic wind flows defined in two layers located on top of each other. Each layer is discretized by a regular spatial grid of desired density. Therefore, the model can be made large scale by increasing density of the spatial discretization, yet it is still cheap to run in terms of computation resources.

This paper contains five sections not counting the Section 1. In Section 2, we briefly cover the background theory regarding the EKF and again point out the problematic parts of this algorithm more precisely. In Section 2, we also discuss the L-BFGS approximation of the EKF and briefly

DOI: 10.1002/fld

describe its stabilized counterpart. In Section 3, we formulate the parallel filtering task and consider its advantageous properties concerning parallelism and also possible complications that may arise when solving the task. Section 4 contains overall review of the two-layer QG-model, whilst Section 5 is devoted to numerical experiments. Lastly, the paper is wrapped out by a short discussion of results obtained from the numerical simulations.

2. STABILIZED APPROXIMATE EXTENDED KALMAN FILTER

We briefly describe the EKF and our earlier results on the approximate versions of it. Consider the following system of two stochastic equations:

$$x_{k+1} = \mathcal{M}_k\left(x_k\right) + \epsilon_k, \tag{1}$$

$$y_{k+1} = \mathcal{H}_{k+1}\left(x_{k+1}\right) + \eta_{k+1}. \tag{2}$$

The first equation describes a dynamical process evolving over discrete time steps; the second models how the current state relates to observed data. The operator $\mathcal{M}_k$ is called the evolution (or transition) operator and $\mathcal{H}_{k+1}$ is the observation operator. Both $\mathcal{M}_k$ and $\mathcal{H}_{k+1}$ can be nonlinear. The terms ϵ_k and η_{k+1} are stochastic and included to represent prediction and measurement errors in (1) and (2), respectively. We denote by C_{ϵ_k} and $C_{\eta_{k+1}}$ the covariance matrices of the corresponding stochastic parameters.

The classical filtering task is defined as follows: using the estimated state x_k^{est} at time instant k and the observed data y_{k+1} at time instant $k+1$, provide an 'optimal' estimate for the subsequent state x_{k+1}. Here, the optimality may have different meanings. The widely assumed understanding is that assuming that the covariance matrices C_{ϵ_k} and $C_{\eta_{k+1}}$ are known, the resulting estimate should be unbiased and have the minimal variance among all the other unbiased estimates.

The task formulated earlier is solved by EKF. The main idea behind the algorithm is to first make prediction for x_{k+1} by following evolution equation (1) and using previous estimate x_k^{est} and, second, to correct this prediction by the observation y_{k+1}. The correction is calculated by weighted average with contributions from the predicted state and from the observation and with the weights based on error terms ϵ_k and η_{k+1}. More precisely, the EKF reads as follows:

Input : x_k^{est}, $\mathcal{M}_k$, M_k^{TL}, M_k^{AD}, y_{k+1}, $\mathcal{H}_{k+1}$, H_{k+1}^{TL}, H_{k+1}^{AD}, C_k^{est}, C_{ϵ_k}, $C_{\eta_{k+1}}$
Output: x_{k+1}^{est}, C_{k+1}^{est}

foreach *time instance* k **do**
  /*Compute state prediction and its covariance matrix */
  $x_{k+1}^{p} \leftarrow \mathcal{M}_k\left(x_k^{est}\right)$;
  $C_{k+1}^{p} \leftarrow M_k^{TL} C_k^{est} M_k^{AD} + C_{\epsilon_k}$;
  /*Calculate Kalman Gain */
  $G_{k+1} \leftarrow C_{k+1}^{p} H_{k+1}^{AD}\left(H_{k+1}^{TL} C_{k+1}^{p} H_{k+1}^{AD} + C_{\eta_{k+1}}\right)^{-1}$;
  /*Correct predicted state and covariance */
  $x_{k+1}^{est} \leftarrow x_{k+1}^{p} + G_{k+1}\left(y_{k+1} - H_{k+1}^{TL} x_{k+1}^{p}\right)$;
  $C_{k+1}^{est} \leftarrow C_{k+1}^{p} - G_{k+1} H_{k+1}^{TL} C_{k+1}^{p}$;
end

Algorithm 1: Extended Kalman Filter

In the given formulation of the EKF, we use terms M_k^{TL}, M_k^{AD}, H_{k+1}^{TL}, and H_{k+1}^{AD} to account for linearization process needed to reduce generally nonlinear transition and observation models to the linear case. The index TL denotes Jacobian matrix of the corresponding operator; here, the

DOI: 10.1002/fld

acronym TL is an abbreviation for *tangent linear code*. The index AD stays for transpose Jacobian of the operator to which the index is applied, and AD is a shortening for *adjoint code*. The reason to have these special names for Jacobian and transpose Jacobian matrices is to emphasize that as the dimension of the state space can be large, the linearized models (e.g., the Jacobian and the transpose) are actually implemented at the code level and never appear in explicit full-storage representation.

We note that in EKF the predicted covariance C^p_{k+1} is an n-by-n matrix and the Kalman gain G_{k+1} is computed by inversion of an m-by-m matrix and multiplication of an n-by-n, n-by-m, and m-by-m matrices (see lines 3–4 of Algorithm 1). That means that when either n or m (or both) becomes large, the algorithm is likely to run into memory issues and has high computational cost.

An approach to alleviate the problem is to use approximations provided by unconstrained optimization as it was performed for instance in [1, 9, 10] or [11]. Let us recall the L-BFGS method, which we apply in order to approximate the problematic matrices from the EKF formulation. In addition, we use the stabilizing correction from [17] needed to avoid problems due to the approximation. The stabilization guarantees that the approximate matrices preserve positive semi-definiteness. In addition, it allows to decrease the CPU cost and the storage required to properly capture the subspace spanned by the matrices being approximated.

Consider a positive semi-definite linear operator A and a sequence of vector pairs (s_k, y_k) mapped by this operator, that is, $As_k = y_k$. It is known that operator A and its inverse can be approximated by the following recurrent formulae [19–21]:

$$A \approx B_{k+1} = B_k + \frac{y_k y_k^T}{y_k^T s_k} - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k}, \tag{3}$$

$$A^{-1} \approx H_{k+1} = \left(I - \frac{s_k y_k^T}{y_k^T s_k}\right) H_k \left(I - \frac{y_k s_k^T}{y_k^T s_k}\right) + \frac{s_k s_k^T}{y_k^T s_k}. \tag{4}$$

Both formulae are usually initialized by a scaled identity matrix γI with a reasonable choice of γ. Approximations described by (3) and (4) can be employed to derive a quasi-Newton optimization method such that the mapped pairs (s_k, y_k) are defined by its iteration history. This constitutes the main idea behind BFGS unconstrained optimization scheme as described in [16]. In addition, the iteration history can be discarded at some point effectively restarting the recursion in (3) and (4), which leads to the limited-memory version of BFGS often referred to as L-BFGS.

Considering Algorithm 1 we can avoid the problematic matrix operations on lines 3–4 by applying L-BFGS method to the following auxiliary optimization task (note that all matrices in the cost function as follows could be defined as subroutines on the code level so their explicit storage is never required):

$$f(x) = x^T \left(H^{TL}_{k+1} C^p_{k+1} H^{AD}_{k+1} + C_{\eta_{k+1}}\right) x - \left(y_{k+1} - H^{TL}_{k+1} x^p_{k+1}\right)^T x, \tag{5}$$

$$f(x) \to \min.$$

The problem (5) is quadratic and therefore its Hessian and minimizing point are given by formulae (6) and (7), respectively.

$$A = H^{TL}_{k+1} C^p_{k+1} H^{AD}_{k+1} + C_{\eta_{k+1}}, \tag{6}$$

$$x_{min} = A^{-1}\left(y_{k+1} - H^{TL}_{k+1} x^p_{k+1}\right). \tag{7}$$

Hence, applying L-BFGS minimization to (5), we obtain approximate values for (6) and (7).

In the same way, an auxiliary optimization task can be formulated in order to obtain an approximation of the updated estimate covariance C^{est}_{k+1} defined on line 6 of Algorithm 1. This approximation together with the approximations of (5) and (6) can be directly applied to formulate a low-memory approximation of the EKF, which is outlined in the algorithmic block 2.

DOI: 10.1002/fld

Input : x_k^{est}, $\mathcal{M}_k$, M_k^{TL}, M_k^{AD}, y_{k+1}, H_{k+1}^{TL}, H_{k+1}^{AD}, $\hat{C}_k^{est}$, C_{ϵ_k}, $C_{\eta_{k+1}}$
Output: x_{k+1}^{est}, $\hat{C}_{k+1}^{est}$

1 **foreach** *time instance* k **do**
  /*Compute prediction of the state */
2 $x_{k+1}^p \leftarrow \mathcal{M}_k\left(x_k^{est}\right)$;
  /*Define approximate covariance operator of the predicted state */
3 **function** $PriorCovariance(x) \rightarrow C_{k+1}^p x$**:**
4 $C_{k+1}^p x \leftarrow M_k^{TL} \hat{C}_k^{est} M_k^{AD} x + C_{\epsilon_k} x$;
5 **return** $C_{k+1}^p x$;
6 **end**
  /*Compute corrected estimate and covariance */
7 **function** $AuxiliaryCostFunction1(x) \rightarrow y$**:**
    /*Define cost function as suggested by (5) */
8 $y \leftarrow H_{k+1}^{AD} x$;
9 $y \leftarrow PriorCovariance(y)$;
10 $y \leftarrow x^T\left(H_{k+1}^{TL} y + C_{\eta_{k+1}} x\right) - \left(y_{k+1} - H_{k+1}^{TL} x_{k+1}^p\right)^T x$;
11 **end**
  /*Subroutine LBFGS below takes a quadratic cost function and yields its minimizing point x_{min} and operator $\widehat{A^{-1}}$ approximating the inverse Hessian. For details on implementation of LBFGS refer to [21] */
12 $(x_{min}, \widehat{A^{-1}}) \leftarrow$LBFGS(*AuxiliaryCostFunction1*);
  /*Compute corrected estimate */
13 $x_{k+1}^{est} \leftarrow x_{k+1}^p + PriorCovariance\left(H_{k+1}^{AD} x_{min}\right)$;
  /*Compute approximate covariance matrix $\hat{C}_{k+1}^{est}$ of the state estimate x_{k+1}^{est} */
14 $b \leftarrow$random();
15 **function** $AuxiliaryCostFunction2(x) \rightarrow y$**:**
16 $aux \leftarrow PriorCovariance(x)$;
17 $y \leftarrow H_{k+1}^{TL} aux$;
18 $y \leftarrow \widehat{A^{-1}}(y)$;
19 $y \leftarrow H_{k+1}^{AD} y$;
20 $y \leftarrow PriorCovariance(y)$;
21 $y \leftarrow x^T(aux - y) - b^T x$;
22 **return** y;
23 **end**
  /*For brevity of notations, below we assume that LBFGS yields forward Hessian */
24 $(\sim, \hat{C}_{k+1}^{est}) \leftarrow$LBFGS (AuxiliaryCostFunction2);
25 **end**

Algorithm 2: Approximate L-BFGS Extended Kalman Filter

The given algorithm is called L-BFGS EKF. It was proposed by Auvinen *et al*. in [1] and then was adopted for variational formulation of the Kalman filter in [9]. Unfortunately, this approach is vulnerable to accumulating round-off errors. This becomes more clear when considering the matrix $\widehat{A^{-1}}$ obtained on line 12 of the algorithm. This matrix approximates the inverse of (6), which can

DOI: 10.1002/fld

be seen as operator reverting perturbed projection of prior covariance C^p_{k+1} onto the observation space. Therefore, matrix $\widehat{A^{-1}}$ should be at least positive semi-definite to maintain the physical properties of its exact counterpart. The latter is guaranteed by the properties of the BFGS update formula as described in [21]. However, when $\widehat{A^{-1}}$ is plugged into the expression for updated estimate covariance C^{est}_{k+1}, the resulting matrix may become indefinite, which makes the corresponding cost function (as defined on lines 15–23 of Algorithm 2) non-convex and collapses the L-BFGS convergence with exact line search as it was originally proposed in [1]. The algorithm is thus vulnerable to numerical errors, as experimentally demonstrated in [17].

The described problem is circumvented by stabilizing correction that can be embedded into the L-BFGS EKF formulae as suggested in [17]. The correction ensures that the cost functions in Algorithm 2 have positive semi-definite Hessians. We recall the following lemma.

Lemma 1
Consider a symmetric positive semi-definite matrix C of dimension n-by-n. Let H be an arbitrary matrix of dimension m-by-n and R be a positive semi-definite matrix of dimension m-by-m. Finally, assume that B is a symmetric m-by-m matrix and I is identity m-by-m matrix. Then the following matrix is positive semi-definite:

$$C - CH^T \left(2I - B\left(HCH^T + R\right)\right) BHC. \tag{8}$$

The proof of this lemma can be found in [17]. We can use this result to introduce corrections into the L-BFGS EKF algorithm. Ultimately, we replace the cost function from the second optimization task (lines 15–23 in algorithmic box 2) with the following:

$$f(x) = x^T \left(C^p_{k+1} - C^p_{k+1} H^{AD}_{k+1} \left(2I - \widehat{A^{-1}}A\right) \widehat{A^{-1}} H^{TL}_{k+1} C^p_{k+1}\right) x - b^T x, \tag{9}$$

$$f(x) \to \min$$

where $A = H^{TL}_{k+1} C^p_{k+1} H^{AD}_{k+1} + C_{\eta_{k+1}}$ as in (6). The BFGS update (3) guarantees that $\widehat{A^{-1}}$ is symmetric (and even positive semi-definite although we do not use it here). Hence, by Lemma 1, the optimization task (9) is convex and L-BFGS minimization can be safely applied. For more details, see [17]. Note that by plugging A^{-1} into the place of $\widehat{A^{-1}}$ in (9) we get back to the formula for C^{est}_{k+1} from the standard EKF.

In the next section, we introduce the parallel filtering task and describe how it can be efficiently solved using the methods based on the former considerations.

3. PARALLEL FILTERING DATA ASSIMILATION TASK AND WEAK-CONSTRAINT 4D-VAR

3.1. Parallel filtering task

Let us recall the dynamical system with observed behavior defined by coupled equations (1) and (2). Assume that $x_k \in \mathbb{R}^n$ is the state vector of the system and $y_k \in \mathbb{R}^m$ is the observation. We define combined state and combined observation as follows:

$$X_k = \left(x_{k-p+1}, x_{k-p+2}, \ldots, x_k\right), \tag{10}$$

$$Y_{k+1} = \left(y_{k-p+2}, y_{k-p+3}, \ldots, y_{k+1}\right). \tag{11}$$

Therefore, the new state space is modeled by $\mathbb{R}^{np}$ and the new observation space is described by $\mathbb{R}^{mp}$ linear spaces both composed of combined states and observations, respectively.

DOI: 10.1002/fld

To define the parallel filtering task, we need to extend transition operator from (1) onto combined state space $\mathbb{R}^{np}$. This can be performed as suggested by the following:

$$\bar{\mathcal{M}}_k = \left(\mathcal{M}_{k-p+1}\left(x_{k-p+1}\right), \mathcal{M}_{k-p+2}\left(x_{k-p+2}\right), \ldots, \mathcal{M}_k\left(x_k\right)\right). \tag{12}$$

The advantageous property of operator (12) is its ability to independently invoke simulations forecasts initialized by the individual components x_{k-i} ($i = p-1, \ldots, 0$) of the combined state X_k, which enables possibility for natural parallelization (i.e., this is what makes the parallel task 'parallel').

The new definition for observation mapping is very similar to that of the transition operator 12:

$$\bar{\mathcal{H}}_{k+1} = \left(\mathcal{H}_{k-p+2}\left(x_{k-p+2}\right), \mathcal{H}_{k-p+3}\left(x_{k-p+3}\right), \ldots, \mathcal{H}_{k+1}\left(x_{k+1}\right)\right). \tag{13}$$

Naturally, the argument regarding the parallelization properties of (12) can be directly applied to (13). Hence, both $\bar{\mathcal{M}}_k$ and $\bar{\mathcal{H}}_{k+1}$ can be implemented by p independent workers, with the cost of single forecast/observation assigned to each worker.

Finally, we need to redefine prediction error covariance C_{ϵ_k} and observation error covariance $C_{\eta_{k+1}}$:

$$\bar{C}_{\epsilon_k} = \begin{pmatrix} C_{\epsilon_{k-p+1}} & & & \\ & C_{\epsilon_{k-p+2}} & & \\ & & \cdots & \\ & & & C_{\epsilon_k} \end{pmatrix}, \tag{14}$$

$$\bar{C}_{\eta_{k+1}} = \begin{pmatrix} C_{\eta_{k-p+2}} & & & \\ & C_{\eta_{k-p+3}} & & \\ & & \cdots & \\ & & & C_{\eta_{k+1}} \end{pmatrix}. \tag{15}$$

We should mention that the block-diagonal matrices (14) and (15) are not the only possible way to extend error covariances to the combined spaces $\mathbb{R}^{np}$ and $\mathbb{R}^{mp}$. For instance, the off-block elements can be nonzero, which enables possibility to account for cross-time correlations for the errors within data assimilation window.

Obviously, the parallel filtering task can be thought of as a particular case of (1) and (2); hence, the Kalman filter can be in principle directly applied. However, because each observation is ultimately repeated p times during data assimilation, the parallel filtering task solved by the Kalman filter turns out to be equivalent to the fixed-lag Kalman smoother as formulated in [22] and thus provides only suboptimal estimate.

The main drawback behind the data assimilation task defined by (12) and (13) is that it can become really large scale, especially if dimension n of the original non-combined state was already large. Therefore, we need to use low-memory approximations introduced earlier in the previous sections. In this paper, we elaborate the L-BFGS EKF modified by the stabilizing correction. As was already discussed, the L-BFGS EKF is able to provide low-memory approximation for the full EKF, whereas the stabilizing correction makes it resistant to the round-off errors. In addition, in [22], it was shown that the filtering problems (12) and (13) inherit stability properties of the filter, and therefore, the stabilizing correction that under certain assumption guarantees stability of the filter [17] will also assure stability of the series of estimates for our parallel filtering task. Hereinafter, for brevity, we will use term 'parallel filter' when referring to the procedure of solving the parallel filtering task by means of stabilized L-BFGS EKF.

We wish to particularly emphasize that the estimate of the combined state X_k^{est} and its covariance matrix $\bar{C}_k^{est}$ are essentially computed by a usual step of the Kalman filter (one has of course to implement this step using L-BFGS approximation and the stabilizing correction to address the issues raised by dimension of the combined state space) with the model and observation operators

DOI: 10.1002/fld

defined by (12) and (13) and with the error covariance matrices from (14) and (15). Hence, provided the formalism we have introduced, a single step of the Kalman filter produces estimates of several consequent states of the original model and calculates the covariance matrix of their joint distribution $\bar{C}_k^{est}$.

In the following subsection, we discuss an approach that comes very close to this idea. It does not directly compute the covariance matrix $\bar{C}_k^{est}$, but instead makes certain assumptions regarding its structure and leaves its exact implementation as a hand-tuned parameter.

3.2. Weak-constraint 4D-VAR

The data assimilation algorithm we review in this section is called weak-constraint 4D-VAR. This method employs framework, which in the essence is very similar to our parallel filter. The weak-constraint 4D-VAR is a gradient-based approach, that is, it requires presence and takes advantage of tangent linear and adjoint codes. The algorithm has been originally suggested in [23] and then extended in [24] and [25] so that its convergence was improved by means of smart preconditioning schemes applied to the likelihood function. Notwithstanding the similarities between this method and our parallel filter discussed earlier, the assumptions made in the derivations of the algorithms are rather different so that both methods are having their own strong and weak sides. Taking into account that the methods come so close that the similarities may become confusing, we wish to review the main concepts of derivation of the weak-constraint 4D-VAR. In addition, in the concluding part of this section, we briefly contrast the main differences between the algorithms.

A natural background for derivation of the Kalman filter relies on the Bayes theorem. We will present the weak-constraint 4D-VAR following the same scheme. However, unlike the classical formulations of the Kalman filter, the weak-constraint 4D-VAR exploits the concept of assimilation window with several sub-windows.

We begin derivation of the weak-constraint 4D-VAR by considering an assimilation window of p consequent states $\{x_{k-i}\}_{i=p-1}^{i=0}$ of a given dynamical system and a sequence of observations $\{y_{k-i+1}\}_{i=p-1}^{i=0}$. Hereinafter, for formulation of the weak-constraint 4D-VAR, we will use the concept of combined state X_k and combined observation Y_{k+1} introduced in (10) and (11).

Our task is to compute a posterior estimate of X_{k+1} given Y_{k+1}. Recalling the Bayes theorem, we restate the problem by the following product of likelihood and prior probability density functions. To obtain a posterior estimate with highest probability, this product should be maximized with respect to combined state variable X_{k+1}:

$$p\left(X_{k+1}|Y_{k+1}\right) \propto p\left(Y_{k+1}|X_{k+1}\right) p\left(X_{k+1}\right). \tag{16}$$

The final formulation of the weak-constraint 4D-VAR is obtained by making few extra assumptions about the distributions in (16). First, we assume that an arbitrary j-th observation relies only on the corresponding j-th state and the observations are jointly independent from each other given the corresponding states (the latter is equivalent to time-independent measurement noise in the observations). Given this supposition, the likelihood $p\left(Y_{k+1}|X_{k+1}\right)$ can be expanded into a product of likelihoods as follows:

$$p\left(Y_{k+1}|X_{k+1}\right) = \prod_{j=0}^{p-1} p(y_{k-j+1}|x_{k-j+1}). \tag{17}$$

Our second assumption is that an arbitrary state x_j depends only on its preceding ancestor x_{j-1}, that is, the state evolution process forms a Markov chain. Therefore, the prior of the state sequence $p\left(X_{k+1}\right)$ can be factorized into the following product:

$$p\left(X_{k+1}\right) = p\left(x_{k-p+2}\right) \prod_{j=0}^{p-2} p\left(x_{k-j+1}|x_{k-j}\right). \tag{18}$$

DOI: 10.1002/fld

Collecting equations (16), (17), and (18), we obtain the following reformulation of the likelihood function:

$$p\left(X_{k+1}|Y_{k+1}\right) \propto p\left(x_{k-p+2}\right) \prod_{j=0}^{p-1} p(y_{k-j+1}|x_{k-j+1}) \prod_{j=0}^{p-2} p\left(x_{k-j+1}|x_{k-j}\right). \tag{19}$$

It remains to define the density functions in (19). In order to do that, we have to rely on few further assumptions. Firstly, for the distribution of the state vector x_{k-p+2}, we assume a Gaussian prior with mean x_b^{k-p+2} and covariance matrix B_{k-p+2}. Therefore,

$$p\left(x_{k-p+2}\right) \propto \exp\left(-\left(x_{k-p+2}-x_b^{k-p+2}\right)^T B_{k-p+2}^{-1}\left(x_{k-p+2}-x_b^{k-p+2}\right)\right). \tag{20}$$

In the 4D-VAR framework, x_b^{k-p+2} is often called *background state* and B_{k-p+2} is called *background covariance matrix*. In practice, x_b^{k-p+2} is often taken to be the prediction initialized by one of the recent estimates of the model state.

The likelihoods $p(y_{k-j+1}|x_{k-j+1})$ that essentially represent the measurement errors are also assumed to be Gaussian, with error covariance matrices R_{k-j+1}:

$$p(y_{k-j+1}|x_{k-j+1}) \propto \prod_{j=0}^{p-1} \exp\left(-d_{k-j+1}^T R_{k-j+1}^{-1} d_{k-j+1}\right), \tag{21}$$

where $d_{k-j+1} = \mathcal{H}_{k-j+1}\left(x_{k-j+1}\right) - y_{k-j+1}$ determines departure of the observation.

Finally, the densities of prediction distributions $p(x_{k-j+1}|x_{k-j})$ are defined by again resorting to the Gaussian assumption, which is now made regarding the model prediction discrepancies $\epsilon_{k-j} = x_{k-j+1} - \mathcal{M}_{k-j}(x_{k-j})$ assuming the mean model error $\bar{\epsilon}_{k-j}$ and the corresponding covariance matrix Q_{k-j} that are both subjects for hand tuning. Hence,

$$p(x_{k-j+1}|x_{k-j}) \propto \prod_{j=0}^{p-2} \exp\left(-\left(\epsilon_{k-j}-\bar{\epsilon}_{k-j}\right)^T Q_{k-j}^{-1}\left(\epsilon_{k-j}-\bar{\epsilon}_{k-j}\right)\right). \tag{22}$$

Substituting (20), (21), and (22) into (19) and taking the negative logarithm of the result, we obtain the weak-constraint 4D-VAR cost function, which should be minimized with respect to X_{k+1} (in the following, we ignore the constant terms because they do not affect the optimization):

$$\begin{aligned}\mathcal{L}_1\left(X_{k+1}\right) = &\left(x_{k-p+2}-x_b^{k-p+2}\right)^T B_{k-p+2}^{-1}\left(x_{k-p+2}-x_b^{k-p+2}\right)\\ &+\sum_{j=0}^{p-1} d_{k-j+1}^T R_{k-j+1}^{-1} d_{k-j+1} +\\ &\sum_{j=0}^{p-2}\left(\epsilon_{k-j}-\bar{\epsilon}_{k-j}\right)^T Q_{k-j}^{-1}\left(\epsilon_{k-j}-\bar{\epsilon}_{k-j}\right).\end{aligned} \tag{23}$$

3.3. Relation between the parallel filter and the weak-constraint 4D-VAR

In the final part of this section, we briefly consider the relation between the weak-constraint 4D-VAR and the parallel filtering task introduced earlier. In order to do that, we refer to the variational formulation of the Kalman filter from [9]. Hence, retaining the notation from Section 3.1, we can restate the estimation problem with equivalent minimization task:

$$\begin{aligned}\mathcal{L}_2\left(X_{k+1}\right) = &\left(X_{k+1}-X_{k+1}^p\right)^T\left(\bar{C}_{k+1}^p\right)^{-1}\left(X_{k+1}-X_{k+1}^p\right)\\ &+\left(Y_{k+1}-\bar{\mathcal{H}}_{k+1}^{TL}\left(X_{k+1}^p\right)\right)^T \bar{C}_{\eta_{k+1}}^{-1}\left(Y_{k+1}-\bar{\mathcal{H}}_{k+1}^{TL}\left(X_{k+1}^p\right)\right),\end{aligned} \tag{24}$$

DOI: 10.1002/fld

where by $\bar{\mathcal{H}}^{TL}_{k+1}$, we denote the linearization of the observation operator (i.e., not only Jacobean matrix but also the corresponding constant shift as required by the Taylor formula).

Finally, assuming block-diagonal structures for covariance matrices $\bar{C}^{p}_{k+1}$ and $\bar{C}_{\eta_{k+1}}$ with blocks C^{p}_{k-j+1} and $C_{\eta_{k-j+1}}$, respectively, and $j = p-1, \ldots, 0$, we arrive at reformulation of (24), which is very similar to the cost function of the weak-constraint 4D-VAR defined by (23):

$$\begin{aligned}\mathcal{L}_2\left(X_{k+1}\right) = &\left(x_{k-p+2} - x^{p}_{k-p+2}\right)^{T}\left(C^{p}_{k-p+2}\right)^{-1}\left(x_{k-p+2} - x^{p}_{k-p+2}\right)\\ &+\sum_{j=0}^{p-1} d^{T}_{k-j+1} C^{-1}_{\eta_{k-j+1}} d_{k-j+1} + \\ &\sum_{j=0}^{p-2}\left(x_{k-j+1} - x^{p}_{k-j+1}\right)^{T}\left(C^{p}_{k-j+1}\right)^{-1}\left(x_{k-j+1} - x^{p}_{k-j+1}\right).\end{aligned} \tag{25}$$

Let us now consider how (25) relates to (23). For simplicity, we assume that the observation operator is linear. This assumption may be relaxed and justifications later remain correct, although in this case, one has to consider linearization of the cost function (23) as suggested by Trémolet in [24]. It is easy to observe that the first two terms in (25) are similar to that of (23). Moreover, in accordance with equation (1), $x^{p}_{k-j+1} = \mathcal{M}_{k-j}\left(x_{k-j}\right) + \epsilon_{k-j}$. Hence, substituting this value into (25) makes it equivalent to (23) with $\bar{\epsilon}_{k-j} = 0$ (which does not play any significant role because the systematic bias of the model can always be 'embedded' into the model itself).

We need to notice however that minimization of (25) is not equivalent to minimization of (23). Indeed, during minimization of (25), the values of the forecasts x^{p}_{k-j+1} remain *fixed*, whilst the cost function (23) uses transition equation (1) to allow their variation during optimization. Therefore, minimization of (25) ultimately aligns the estimate with the observation and the forecast, whereas minimization of (23) aligns the estimate with the observation and minimizes the discrepancy of the model at the same time. These arguments are summarized by the following lemma.

Lemma 3.3.1
Consider the parallel Kalman filter defined using the notation introduced in Section 3.1. Assume that covariance matrix $\bar{C}^{p}_{k+1}$ of combined forecast and covariance matrix $\bar{C}_{\eta_{k+1}}$ of the observation errors are block diagonal (i.e., the forecasts and the observations are not correlated in time). Then, the estimate of the weak-constraint 4D-VAR is equivalent to that of the parallel filter if the latter is computed by minimizing (25) with $x^{p}_{k-j+1} = \mathcal{M}_{k-j}\left(x_{k-j}\right)$.

Proof
The proof of this lemma immediately follows from the argumentation provided in this section. □

4. QUASI-GEOSTROPHIC MODEL

In this section, we briefly introduce two-layer QG-model, which was used as a toy case for our numerical benchmarks. This model is nonlinear and is proved to be chaotic [18]. In addition, it can be made large scale by adjusting resolution of its spatial grid. However, even for high-resolution cases, the QG-model remains cheap to run and can be executed with no need for special hardware.

The two-layer QG-model resides on a cylindrical surface divided into two interacting vertical layers each having depth parameter H_1 and H_2. Schematic layout of model geometry is provided in Figure 1. The model describes behavior of quasi-geostrophic wind motion and simulates stream function, which acts as the wind speed field potential and is rigorously defined by the following:

$$\nabla\psi_i = (v_i, -u_i),\ i = 1, 2.$$

Here, u_i and v_i are horizontal and vertical components of the wind speed, respectively. Subscript i denotes the model's layer for which parameter is defined. Value $i = 1$ corresponds to the top layer, and $i = 2$ corresponds to the bottom layer of the model.

DOI: 10.1002/fld

The formal mathematical definition of the model is given by the following system of PDEs:

$$\frac{D_1 q_1}{Dt} = 0, \quad \frac{D_2 q_2}{Dt} = 0 \tag{26}$$

$$q_1 = \nabla^2 \psi_1 - F_1 (\psi_1 - \psi_2) + \beta y \tag{27}$$

$$q_2 = \nabla^2 \psi_2 - F_2 (\psi_2 - \psi_1) + \beta y + R_s. \tag{28}$$

Here, the coupled equations (27) and (28) denote potential vorticity operator. Operator defines relation between the stream function ψ_i and potential vorticity q_i. It can be proved that under certain circumstances this relation is a one-to-one correspondence [26]. Operator $\frac{D_i \cdot}{Dt}$ in (26) denotes substantial derivative corresponding to speed field (u_i, v_i), that is, $\frac{D_i \cdot}{Dt} = \frac{\partial \cdot}{\partial t} + u_i \frac{\partial \cdot}{\partial x} + v_i \frac{\partial \cdot}{\partial y}$.

Equations (26)–(28) are presented in non-dimensional form and relate to physical parameters through the standard procedure of non-dimensionalization, which we provide as follows for completeness:

$$F_1 = \frac{f_0^2 L^2}{\acute{g} H_1}, F_2 = \frac{f_0^2 L^2}{\acute{g} H_2},$$
$$\acute{g} = g \frac{\Delta\theta}{\bar{\theta}}, R_s = \frac{S(x, y)}{\eta H_2},$$
$$\beta = \beta_0 \frac{L}{U}.$$

In the non-dimensionalization relations earlier, f_0 is the Coriolis parameter, $\Delta\theta$ and $\bar{\theta}$ are change and mean values of potential temperature across the layer interface, L and U are main length and velocity scales, $S(x, y)$ is topography surface that affects the bottom layer of the model, β_0 is northward gradient of the Coriolis parameter, and $\eta = \frac{U}{f_0 L}$ is the Rossby number of the system. Table I summarizes the values, which we used in order to set up the model for our numerical benchmarks that are discussed in more detail in the next section.

In the experiments that we run, topography component $S(x, y)$ was modeled by a Gaussian hill. The layer depths H_1 and H_2 were used to introduce a bias into the model used for data assimilation. Their values are also clarified in the next section.

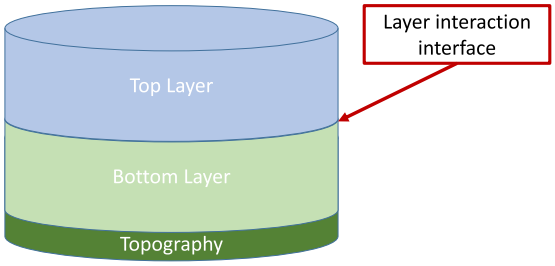

Figure 1. Benchmark results: 10 observations in use.

Table I. Quasi-geostrophic model settings used in the truth and in the biased runs.

Parameter name	Notation	Value
Coriolis parameter	f_0	10^{-4}
Length scale	L	10^6
Velocity scale	U	10
Potential temperature change/mean ratio	$\Delta\theta/\bar{\theta}$	0.1
Northward gradient of Coriolis parameter	β_0	$1.5 \cdot 10^{-11}$

DOI: 10.1002/fld

5. EXPERIMENTS

This section is devoted to numerical experiments designed to test the estimation performance of the parallel filter introduced in Section 3.1. Because the dimension of the data assimilation tasks in our benchmarks still allows the use of the EKF, we employ it as the reference method. We wish to test the parallel filter with the weak-constraint 4D-VAR, as well as against several ensemble methods, as they provide another natural way to parallelization. We should mention that the ensemble filters do not use the information given by the tangent linear and adjoint codes. On the other hand, this makes their implementation technically simpler because the tangent linear and adjoint codes usually require meticulous and time-consuming programming.

The rest of this section is organized as follows: we firstly introduce the general setting of our data assimilation model and define the metrics to asses performance of the algorithms; then, we review configuration of the filters; finally, we discuss the particular benchmarks and present results obtained from the runs.

5.1. General data assimilation settings

We wanted to make the settings of our benchmark closer to the real case; therefore, we introduced a bias into transition operator. This was carried out the following way: the QG-model was executed twice with different parameter values. The evolution trajectory of one run was employed to generate observed data, and the other run was used for data assimilation. Hereinafter, we will refer to the instance of the QG-model used to collect observations as to the truth run, whereas the instance employed in data assimilation will be called the biased run. For both runs, a single execution of the model was set to evolve the input state for 1 h of the model time. Table II summarizes parameters that we used to initialize these runs. Parameters of the QG-model that are not listed in the table did not differ between the truth and the biased model instances (the values for them can be found in the previous section).

In order to be able to analyze convergence of the methods, it was needed to specify a concept of distance between the state vectors from the truth and the biased runs. Let x_{truth} be a state vector from the truth run and x_{est} be the corresponding estimate computed by data assimilation. Then, the standard technique to measure the distance between these vectors is provided by root mean square relative error (RMS error), which is defined as follows:

$$R = \frac{\|x_{\text{truth}} - x_{est}\|}{\sqrt{n}}, \tag{29}$$

where n is dimension of the state space. If the truth run and the biased run have different resolutions (as in our benchmarks), the term x_{est} should be interpolated onto the spatial grid employed by the truth model and then formula (29) can be applied normally.

Convergence of the data assimilation process was analyzed by assessing its ability to drive estimates towards trajectory of the truth run beginning from biased initial guess. In order to simulate the initial bias, both the truth and the biased runs were initialized by the same starting state and then executed for 2 weeks of the model time. The final state of the biased run obtained at the end of the 2-week spin-up period was then used as the biased initial guess for data assimilation.

Recall observed dynamical systems (1) and (2). For our data assimilation benchmarks, the transition operator $\mathcal{M}_k$ was defined by six consequent executions of the biased run, which corresponds to

Table II. Quasi-geostrophic-model settings used in the truth and in the biased runs.

Parameter name	Truth run	Biased run
Horizontal resolution	80	40
Vertical resolution	40	20
Top layer depth	6000	5500
Bottom layer depth	4000	4500

DOI: 10.1002/fld

6-h evolution in the model time. Observation operator $\mathcal{H}_k$ was modeled by simple linear mapping that firstly interpolates the biased run state onto the higher dimensional grid employed by the truth run and then randomly selects a few components from the result of the interpolation. The chosen components are combined into a vector and then treated as the observation.

Finally, for covariance matrices of the noise terms ϵ_k and η_{k+1} from equations (1) and (2), and for covariance matrix C_0^{est} of the initial guess, we assumed the following values:

$$C_{\epsilon_k} = \begin{pmatrix} 0.1^2 \cdot I, 0.005 \cdot I \\ 0.005 \cdot I, 0.1^2 \cdot I \end{pmatrix},\ C_{\eta_{k+1}} = 0.5^2 \cdot I,\ C_0^{est} = I, \tag{30}$$

where I is an n-by-n identity matrix and n denotes dimension of the state space as before.

All our benchmarks cover 200 data assimilation steps (i.e., $200 \cdot 6 = 1200$ h or 50 days of model time). Performance of the methods is studied by considering RMS curves of the estimates against the artificial truth following the definition in (29).

5.2. *Configuration of the filters*

Let us firstly discuss parameters that are specific to the parallel Kalman filter. In all our experiments, the window length (parameter p in equalities (12) and (13)) was set to three. Secondly, combined covariance matrices $\bar{C}_{\epsilon_k}$ and $\bar{C}_{\eta_{k+1}}$ did not have nonzero cross-time correlation blocks. Hence, these matrices were block-diagonal as defined in (14) and (15). It is possible that the parallel filter may perform slightly better if the model and observation error cross-time correlations are taken into account. However, this configuration has not been considered in this paper because of limitations of our present implementation of the parallel filter.

As discussed earlier in Section 3.1, the parallel filtering task is complicated by dimension of the related combined state space. Hence, we use stabilized L-BFGS approximation of the EKF to circumvent this complication. Table III summarizes parameters used in L-BFGS minimization step of the filter (see Section 2 for further details). All of the benchmarks share the same configuration of the minimizer.

In Table III, the gradient and function tolerance values identify optimizer's stopping criteria that are based on Euclidean difference between the consequent cost function values and the corresponding consequent gradients obtained during iterations of the optimizer.

The ensemble-based methods that we included into our test cases were the following: the classical approach by Evensen discussed in [3] (EnKF), the variational approach introduced in [9] (nicknamed as VEnKF), and a more recent method from [11] based on the 'randomize-then-optimize' ideas (abbreviated as RTOEnKF). The methods that are based on L-BFGS minimization (VEnKF from [9] and RTOEnKF from [11]) all use the values summarized in Table III. In fact, the number of L-BFGS iterations in Table III is an overkill as in our experience, all the methods converge even for as few as 20 iterations and with the storage capacity of only 20 vector mapped pairs (see (10) for details). However, we wanted to guarantee that all the methods fully demonstrate their potential and that no local divergence could occur.

The EnKF and RTOEnKF are able to benefit from localization schemes. These schemes attempt to account for spurious correlations that may appear (because of the sampling error) between the state elements, which are physically distant from each other. In our experiments, we remove such

Table III. Limited-memory Broyden–Fletcher–Goldfarb–Shanno unconstrained minimizer settings.

Upper iteration limit	20
Maximal iteration history storage	20
Gradient tolerance	10^{-10}
Function value tolerance	10^{-10}
Upper boundary for single step size	10^{10}
Initial Hessian approximation	I

DOI: 10.1002/fld

nonphysical correlations using Gaspari–Cohn correlation cut-off scheme [27]. The VEnKF in the form it was given in [9] does not allow straightforward localization so we do not use it with this method.

All ensemble filters in our benchmarks use 80 ensemble members. Some of them could converge with less, but in this study, we were trying not to tune the methods in order to optimize their performance. Hence, some of the results later may be improved to certain extent if such optimizations are applied. In general, we were trying to set 'overkilling' values everywhere where possible.

The weak-constraint 4D-VAR was having the same length of the assimilation window as the parallel filter (i.e., three sub-windows). Here, we do not present the settings used by the optimizer employed by this method as the corresponding minimization technique is beyond the scope of this paper. The details regarding this technique as well as the settings that we use in our benchmarks can be found in [28].

Finally, we should notice that for the parallel Kalman filter and for the weak-constraint 4D-VAR, we attempt to assess their peak performance, that is, we analyze the quality of their estimates at the middle of the data assimilation window. In other words, if $X_k = (x_{k-2}, x_{k-1}, x_k)$ is the combined state of the system at time instance k, we only analyze the quality of the estimate x_{k-1} as both the parallel filter and weak-constraint 4D-VAR are capable to utilize maximal amount of data at this point.

5.3. *Benchmarks configuration and results*

In our benchmarks, we vary the number of observation sources beginning at some small initial amount and then gradually increasing it.

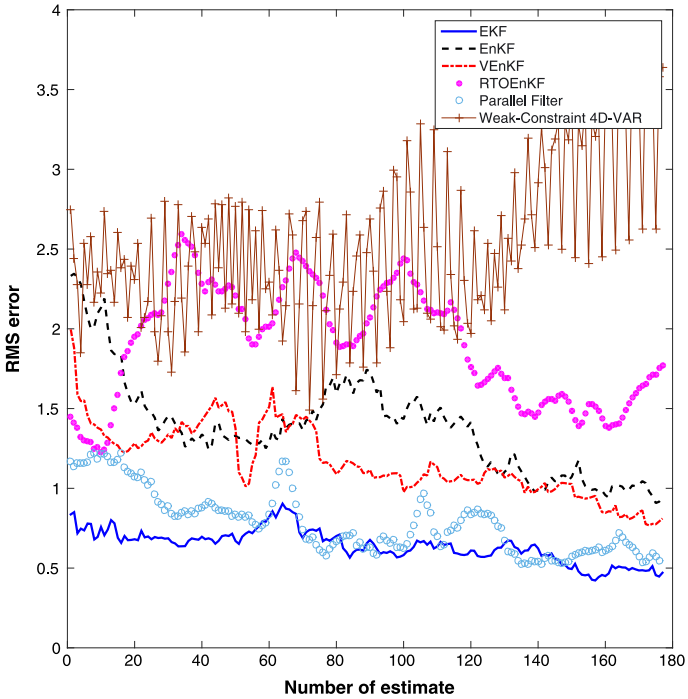

Figure 2. Benchmark results with 20 observations (the first 25 data assimilation steps of premature convergence are left out). EKF, extended Kalman filter; EnKF, ensemble Kalman filter; VEnKF, variational approach introduced in [9]; RTOEnKF, randomize then optimize ideas; RMS error, root mean square relative error.

DOI: 10.1002/fld

Our first benchmark was using only 20 observation sources. The RMS curves obtained from the data assimilation are shown in Figure 2. It can be clearly seen that the parallel filter demonstrates the best performance in terms of RMS error and is only inferior to the EKF (which is natural). In addition, one can notice that in some cases the estimates of parallel filter are as good or even slightly better compared with the estimates computed by the EKF. This can be naturally explained by the ability of the parallel filter to utilize observed information not only at the current time instant but also from the past and future.

In the second benchmark (Figure 3), we employ 200 observations. In this case, the parallel filter is still superior to the others (except the EKF). The ensemble methods demonstrate practically similar performance, and the weak-constraint 4D-VAR converges slightly worse, which might be due to the lack of tuning in the model discrepancy covariance matrices Q_{k-j} from (23).

Finally, in the third benchmark (Figure 4), we model a case with abundant observation sources, which were 500 in this experiment. The results are in general similar to what we obtained in the case of 20 and 200 observations, but now we acquire significant improvement in convergence of the weak-constraint 4D-VAR. Concluding this section, we must again emphasize that we did not carry out any special tuning of the methods, which might be the reason the weak-constraint 4D-VAR did not perform as well as expected in comparison with the other algorithms. Indeed, recalling (23), we note again that the cost function of the weak-constraint 4D-VAR requires hand-picked values for the background covariance matrices B_{k-p+2} and for the model discrepancy covariance matrices Q_{k-j}. While this may require a tedious case-dependent study, we acknowledge that it, if properly carried out, can greatly improve the performance. On the contrary, with parallel Kalman filter, the covariance data are generated as part of the normal filtering process and thus require no manual adjustments after fixing the model and observation error covariances. In our view, this advantageous

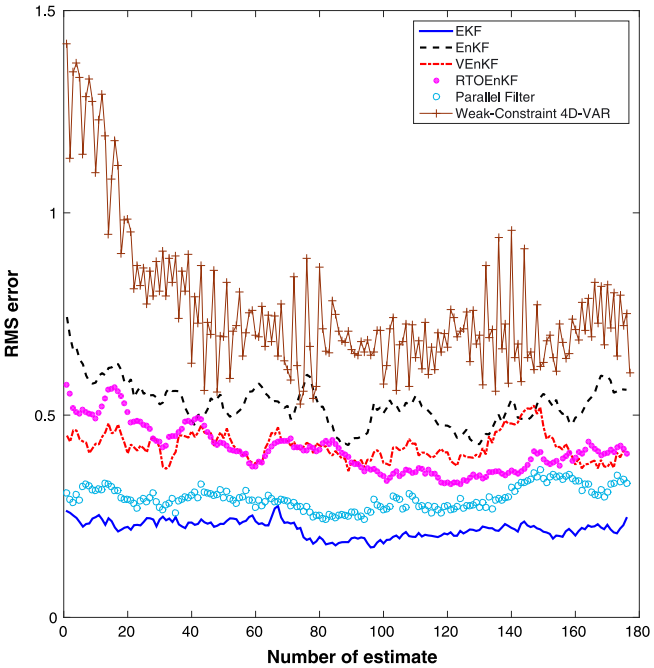

Figure 3. Benchmark results with 200 observations (the first 25 data assimilation steps of premature convergence are left out). EKF, extended Kalman filter; EnKF, ensemble Kalman filter; VEnKF, variational approach introduced in [9]; RTOEnKF, randomize then optimize ideas; RMS error, root mean square relative error.

DOI: 10.1002/fld

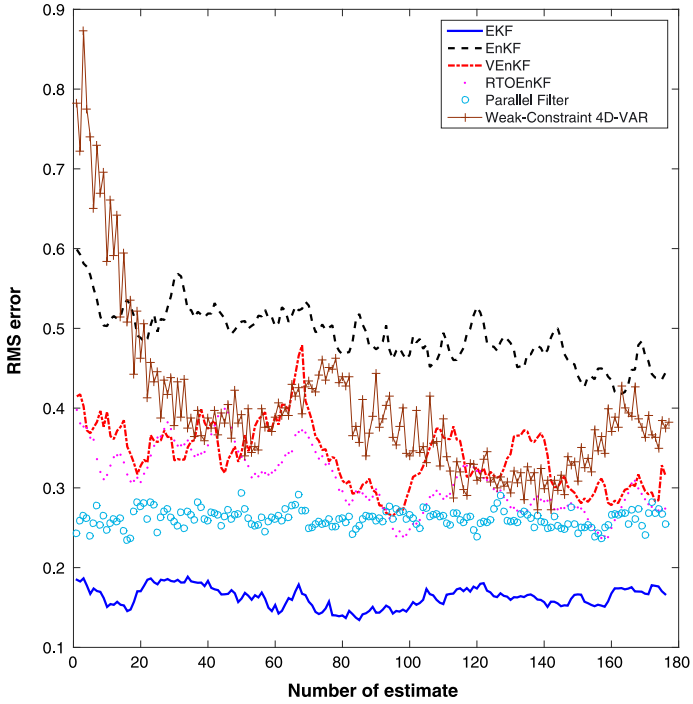

Figure 4. Benchmark results with 500 observations (the first 25 data assimilation steps of premature convergence are left out). EKF, extended Kalman filter; EnKF, ensemble Kalman filter; VEnKF, variational approach introduced in [9]; RTOEnKF, randomize then optimize ideas; RMS error, root mean square relative error.

property of the parallel filter may make its application prospective when particulars of the underlying model are not well known, and therefore, no smart tuning of the algorithm can be easily carried out.

6. CONCLUSIONS

In this paper, we formulated parallel filtering task and employed the stabilized correction for L-BFGS EKF from [17], which allowed to solve this task efficiently. Our numerical experiments demonstrated notable improvement of the estimates computed by the parallel filter over that of a number of ensemble-based methods. In addition, the parallel filter has performed clearly better than the weak-constraint 4D-VAR, which can be derived using very similar ideas, but employs a more complicated 4D-VAR optimization. On the contrary, the parallel Kalman filter can be defined in terms of a simple quadratic minimization that follows the 3D-VAR framework. This property makes the parallel filter advantageous when the underlying dynamics model is provided as a 'black box', and therefore, the abilities for smart hand-tuned adjustments of the filter are limited. On the other hand, the weak-constraint 4D-VAR provides more 'tuning knobs' such as background covariance matrices and model discrepancy covariance matrices. These 'knobs' can be particularly effective when the underlying model is well known, and it is possible to make a good guess regarding the values of the otherwise unknown parameters. In addition, it should be mentioned, that when implemented on a parallel computer, our approach has no extra computation outlay, because the prediction step of the filter is implemented by independent model runs. At the same time, it shows notable improvement over traditional ensemble methods demonstrating ability to efficiently benefit from presence of the gradient codes.

DOI: 10.1002/fld

REFERENCES

1. Auvinen H, Bardsley JM, Haario H, Kauranne T. Large-scale Kalman filtering using the limited memory BFGS method. *Electronic Transactions on Numerical Analysis* 2009; **35**:217–233.
2. Simon D. *Optimal State Estimation. Kalman, H_∞, and Nonlinear Approaches*. Wiley-Interscience: Hoboken, 2006.
3. Evensen G. Sequential data assimilation with a non-linear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research* 1994; **99**(C5):10,143–10,162.
4. Houtekamer PL, He B, Herschel LM. Parallel implementation of an ensemble Kalman filter. *Monthly Weather Review* 2014; **142**:1163–1182.
5. Dee DP. Simplification of the Kalman filter for meteorological data assimilation. *Quarterly Journal of the Royal Meteorological Society* 1990; **117**(498):365–384.
6. Cane MA, Kaplan A, Miller RN, Tang B, Hackert EC, Busalacchi AJ. Mapping tropical Pacific sea level: data assimilation via reduced state Kalman filter. *Journal of Geophysical Research* 1996; **101**(C10):22599–22617.
7. Voutilainen A, Pyhälahti T, Kallio KY, Haario H, Kaipio JP. A filtering approach for estimating lake water quality from remote sensing data. *International Journal of Applied Earth Observation and Geoinformation* 2007; **9**(1):50–64.
8. Fisher M. *Development of a Simplified Kalman Filter. ECMWF Technical Memorandum 260*. ECMWF, 1998.
9. Auvinen H, Bardsley JM, Haario H, Kauranne T. The variational Kalman filter and an efficient implementation using limited memory BFGS. *International Journal on Numerical methods in Fluids* 2009; **64**:314–335.
10. Bardsley JM, Parker A, Solonen A, Howard M. Krylov space approximate Kalman filtering. *Numerical Linear Algebra with Applications* 2011; **20**(2):171–184.
11. Bardsley JM, Solonen A, Parker A, Haario H, Howard M. An ensemble Kalman filter using the conjugate gradient sampler. *International Journal for Uncertainty Quantification* 2013; **3**(4):357–370.
12. Sacher W, Bartello P. Sampling errors in ensemble Kalman filtering. Part I: theory. *Monthly Weather Review* 2008; **136**:3035–3049.
13. Deng X, Heinkenschloss M. *A parallel-in-time gradient-type method for discrete time optimal control problems. Preprint, Department of Computational and Applied Mathematics*. Rice University, 2016. Available from:http://www.caam.rice.edu/$\sim$heinken [Available in 24 June 2016].
14. Rao V, Sandu A, A time-parallel approach to strong-constraint four-dimensional variational data assimilation. *Technical Report*, Virginia Polytechnic Institute and State University, 2015.
15. Niño ED, Sandu A, Deng X. A parallel implementation of the ensemble Kalman filter based on modified Cholesky decomposition. *Technical Report*, 2016.
16. Nocedal J, Wright SJ. *Numerical Optimization, Chapter Limited-memory BFGS*. Springer-Verlag: New York, 1999.
17. Bibov A, Haario H, Solonen A. Stabilized BFGS approximate Kalman filter. *Inverse Problems and Imaging* 2015; **9**(4):1003–1024.
18. Fandry CB, Leslie LM. A two-layer quasi-geostrophic model of summer trough formation in the Australian subtropical easterlies. *Journal of the Atmospheric Sciences* 1984; **41**:807–818.
19. Broyden CG. The convergence of a class of double-rank minimization algorithms 1. General considerations. *IMA Journal of Applied Mathematics* 1970; **6**(1):76–90.
20. Goldfarb D. A family of variable-metric methods derived by variational means. *Mathematics of Computation* 1970; **21**:23–26.
21. Dennis JE, Schnabel RB. *A New Derivation of Symmetric Positive Definite Secant Updates*. University of Colorado at Boulder, Department of Computer Science, 1980.
22. Moore JB. Discrete-time fixed-lag smoothing algorithms. *Automatica* 1973; **9**:163–173.
23. Zupanski D. A general weak constraint applicable to operational 4DVAR data assimilation systems. *Monthly Weather Review* 1996; **125**:2274–2292.
24. Trémolet Y. Accounting for an imperfect model in 4D-VAR. *Quarterly Journal of the Royal Meteorological Society* 2006; **132**(621):2483–2504.
25. Fisher M, Gratton S. *Preconditioning Saddle-point Formulation of the Variational Data Assimilation*, 2014.
26. Pedlosky J. *Geophysical Fluid Dynamics, Chapter Geostrophic Motion*. Springer-Verlag: New York, 1987.
27. Gaspari G, Cohn SE. Construction of correlation functions in two and three dimensions. *Quarterly Journal of the Royal Meteorological Society* 1999; **125**(554):723–757.
28. Fisher M. *An Investigation of Model Error in a Quasi-geostrophic, Weak-constraint, 4D-VAR Analysis System*. Oral presentation: ECMWF, 2009.

DOI: 10.1002/fld

Publication III

Amour, I., Mussa, Z., Bibov, A., and Kauranne, T. (2013) Using ensemble data assimilation to forecast hydrological flumes. *Nonlinear Processes in Geophysics*, 20, pp. 955-964.

Nonlin. Processes Geophys., 20, 955–964, 2013
www.nonlin-processes-geophys.net/20/955/2013/
doi:10.5194/npg-20-955-2013

Nonlinear Processes in Geophysics
Open Access

Using ensemble data assimilation to forecast hydrological flumes

I. Amour, Z. Mussa, A. Bibov, and T. Kauranne

Lappeenranta University of Technology, Lappeenranta, Finland

Correspondence to: I. Amour (idrissa.amour@lut.fi)

Received: 29 May 2013 – Revised: 23 September 2013 – Accepted: 9 October 2013 – Published: 8 November 2013

Abstract. Data assimilation, commonly used in weather forecasting, means combining a mathematical forecast of a target dynamical system with simultaneous measurements from that system in an optimal fashion. We demonstrate the benefits obtainable from data assimilation with a dam break flume simulation in which a shallow-water equation model is complemented with wave meter measurements. Data assimilation is conducted with a Variational Ensemble Kalman Filter (VEnKF) algorithm. The resulting dynamical analysis of the flume displays turbulent behavior, features prominent hydraulic jumps and avoids many numerical artifacts present in a pure simulation.

1 Introduction

Hydrological flumes and other phenomena related to rivers, estuaries, canals, and other water bodies that lend themselves to a one- or two-dimensional description have received somewhat less attention in computational fluid dynamics (CFD) research than flows in industrial processes, or fully three-dimensional flows in general. However, there is one notable exception, namely, weather forecasting.

Weather models are complex codes that have received a lot of attention from the CFD community since the 1950s. Many techniques employed in weather forecasting are naturally amenable to other hydrostatic flows, such as river, estuary, and canal flows, but have yet to be tried in these applications. One of the most prominent of such techniques is data assimilation.

In the current paper, we introduce data assimilation of wave meter data into a river model that was originally presented by Martin and Gorelick (2005). Data assimilation is a process that optimally combines model forecasts with observations. In weather forecasting, data assimilation is used to generate the initial conditions for an ensuing forecast, but also to continuously correct a forecast towards observations, whenever these are available in the course of the forecast.

Turbulence, which is an irregular motion found in fluids (liquids and gases) when passing objects or streamlines of itself passing one another (Goldstein, 1938), is the main reason that makes the CFD model ambitious to assimilate, as we hope for more information from the available measurements to shape the behavior of the model estimates.

In the paper of Martin and Gorelick (2005), the authors present a shallow-water model of a dam break experiment conducted in a laboratory. The goal of the model is to simulate the behavior of the flume in the case that a dam suddenly breaks. Results from numerical simulations are compared with measurements by wave meters and pressure transducers that record water height during the experiment.

In the current article, we take the model and the measurements into account simultaneously in a process of continuous data assimilation. This results in a more realistic representation of the behavior of the flume during the experiment, in the sense that the resulting flume displays turbulent behavior, features prominent hydraulic jumps and avoids many numerical artifacts present in a pure simulation. As our data assimilation algorithm, we have chosen a recent version of the Ensemble Kalman Filter, the Variational Ensemble Kalman Filter (VEnKF) introduced by Solonen et al. (2012). Ensemble Kalman filters have the distinct advantage over other data assimilation methods that they can be implemented as "wrappers" on top of existing CFD codes without necessitating modification of the models themselves. Moreover, they conduct data assimilation continuously, so that their results can be compared with direct numerical simulations.

This paper is organized as follows. In the second section we discuss previous attempts at numerically simulating river flow. Section three describes the MODFreeSurf2D shallow-water code used in the data assimilation experiments introduced in Martin and Gorelick (2005). Section four discusses

data assimilation and describes how it was implemented in the current research effort. Section five describes the dam break experiment and the way data assimilation was modified to accommodate the model and the data in this case. Section six describes the results of numerical tests, both with and without data assimilation. Section seven concludes the paper.

2 Numerical simulation of river flows

Simulation of river flows presents challenges to computational schemes due to the complex geometry and meandering path of the flow. Different schemes have been suggested and applied to try to capture as much information about the river as possible. A two-dimensional finite-element solution has been used to simulate an 11 km long reach of River Culm in Devon, UK, modeled by two-dimensional depth-averaged Reynolds equations (Bates and Anderson, 1993). The numerical model simulation finds an error of $\pm 2\,\%$ in continuity, although the mass conservation is still adequate.

The same method has been applied to flow simulations in Aliparast (2009). The governing equations of the model are 2-D shallow-water equations, where the stress term is ignored because of the influence of bottom roughness caused by turbulent shear stress between grids (Yoon and Kang, 2004). The numerical model has been validated by an example of an oblique hydraulic jump for which an analytical solution is available (Aliparast, 2009). The numerical model has been tested with a dam break case in a converging-diverging flume (Bellos et al., 1991). The flume is 20.7 m long and 1.40 m wide, it has 5 stations used to record the water depth, and a dam is located 8.5 m from the closed end. The results of the dam break case simulation match well with the experimental results (Aliparast, 2009). However, in station 4, which is 13.5 m downstream from the dam, water depth is underestimated.

The finite volume method is a further widely-used numerical scheme. It has been applied for a shallow water model in Heniche et al. (2000), Zhang and Wu (2011) and Ying et al. (2009).

A recent application on the dam break case is presented by Baghlani (2011), where a robust flux vector splitting (FVS) scheme is applied. FVS has been frequently applied in solving similar compressible flow problems (e.g. in Baghlani, 2011; Erpicum et al., 2010; Toro and Vazquez-Cendon, 2012). Two well-known FVS methods are that of Steger and Warming and that of Van Leer (Drikakis and Tsangaris, 1993). Steger and Warming's FVS exploits the homogeneous property of the Euler equation and splits the fluxes into positive and negative parts with respect to the propagation (Drikakis and Tsangaris, 1993). Van Leer's FVS constructs the fluxes as a function of the local Mach number (Drikakis and Tsangaris, 1993). The FSV proposed by Baghlani (2011) decomposes the flux vector into positive and negative components by means of Jacobian matrices of the flux vectors and a Liou–Steffen splitting for decomposing the pressure term. The FVS has been criticized for its expensive computational cost, as the eigensystem of equations must be evaluated at every time step (Baghlani, 2011).

One approach to studying flow behavior is to construct flume properties directly from measurements by regression. A method of surface analysis and velocity changes of this type has been presented (Barcena et al., 2012). It uses regression with respect to a collection of model scenarios to form a continuous function of hydrodynamic responses. The method has been successful in predicting estuarine free surface and velocity with significant savings in computational cost for a short- and medium-term simulation period. One advantage of this method is its ability to simulate a long-term hydrodynamic flow with a short computational time.

Pure 2-D simulations of hydrological flows suffer from several handicaps. Because the numerical flow is two-dimensional, it cannot capture the true three-dimensional flow, in particular turbulence: the numerical solution remains in perpetual hydrostatic balance. Even more importantly, the numerical time-stepping scheme implies that a flow front in front of a discontinuity, such as a flood wave, will only propagate one grid line per time step. The speed of this shock wave is therefore dependent on grid size and the numerical time step, and not on the correct physical speed. Finally, there is no way to connect the simulated flow to the true flow after the initial condition has been fixed. With data assimilation, we hope to address all three defects.

3 MODFreeSurf2D

MODFreeSurf2D is an open Matlab code that is designed to solve depth-averaged shallow-water equations (Martin and Gorelick, 2005). The code implements the semi-implicit, semi-Lagrangian time-stepping scheme of Casulli and Cheng (1992) and Casulli (1999), and uses a finite volume discretization. The scheme is stable and can simulate water/land boundaries (Martin and Gorelick, 2005; Casulli and Cheng, 1992).

3.1 Depth-Averaged Shallow-Water Equations (DASWE)

The governing equations in MODFreeSurf2D are the depth-averaged shallow-water equations as given in Martin and Gorelick (2005):

$$\frac{\partial U}{\partial t} + U\frac{\partial U}{\partial x} + V\frac{\partial U}{\partial y} = -g\frac{\partial \eta}{\partial x} + \varepsilon\left(\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2}\right) + \gamma_T\frac{(U_a - U)}{H} - g\frac{\sqrt{U^2+V^2}}{C_z^2}U + fV, \quad (1)$$

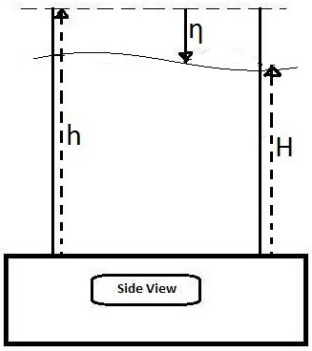

Fig. 1. ModFreeSurf2D variable definition (side view) which shows the relationship between free surface elevation η, total water depth H, and undisturbed water depth h (Martin and Gorelick, 2005).

$$\frac{\partial V}{\partial t}+U\frac{\partial V}{\partial x}+V\frac{\partial V}{\partial y}=-g\frac{\partial \eta}{\partial y}+\varepsilon\left(\frac{\partial^2 V}{\partial x^2}+\frac{\partial^2 V}{\partial y^2}\right)+\gamma_T\frac{(V_a-V)}{H}-g\frac{\sqrt{U^2+V^2}}{C_z^2}V-fU, \quad (2)$$

$$\frac{\partial \eta}{\partial t}+\frac{\partial (HU)}{\partial x}+\frac{\partial (HV)}{\partial y}=0, \quad (3)$$

where U is the depth-averaged x direction velocity component, V is the depth-averaged y direction velocity component, η is the free surface elevation, g is the gravitational constant, t is time, ε is the horizontal eddy viscosity, f is the Coriolis parameter, $H=h+\eta$ is the total water depth, γ_T is the wind stress coefficient, C_z is the Chezy coefficient, and U_a and V_a are wind velocities. In the above, h is the undisturbed water depth. Figure 1 illustrates the variable definitions of MODFreeSurf2D.

Top friction and bottom friction boundaries are given by Eqs. (4) and (5), respectively.

$$\nu\frac{\partial U}{\partial z}=\gamma_T(U_a-U),\quad \nu\frac{\partial V}{\partial z}=\gamma_T(V_a-V) \quad (4)$$

$$\nu\frac{\partial U}{\partial z}=g\frac{\sqrt{U^2+V^2}}{C_z^2}U,\quad \nu\frac{\partial V}{\partial z}=g\frac{\sqrt{U^2+V^2}}{C_z^2}V, \quad (5)$$

where ν is the kinetic viscosity coefficient, and z indicates the vertical direction (Martin and Gorelick, 2005).

3.2 Numerical approximation

In MODFree2DSurf a combination of a semi-implicit, semi-Lagrangian time-stepping scheme and a finite-volume discretization is employed to numerically solve the hydrological shallow-water equations on a rectangular grid. This scheme provides a stable solution, even for a time step larger than the Courant–Friedrichs–Lewy (CFL) restriction defined by

$$\mathrm{CFL}=w\frac{\Delta t}{\Delta x_i}, \quad (6)$$

where w is the velocity component in the x_i direction, $i=1,2$, Δt is the time step size, and Δx_i is the cell dimension in the x_i direction of flow (Martin and Gorelick, 2005). The CFL relates fluid velocity and time step size to computational cell size, and requires that it should be smaller than 1 (Martin and Gorelick, 2005).

3.2.1 Semi-implicit representation

In this representation, the free surface elevation η and the horizontal velocity components U and V are the unknown variables at time $N+1$:

$$\begin{aligned}\eta_{i,j}^{N+1}=&\eta_{i,j}^{N}-\theta\frac{\Delta t}{\Delta x}\left(H_{i+1/2,j}^{N}U_{i+1/2,j}^{N+1}-H_{i-1/2,j}^{N}U_{i-1/2,j}^{N+1}\right)\\&-\theta\frac{\Delta t}{\Delta x}\left(H_{i,j+1/2}^{N}V_{i,j+1/2}^{N+1}-H_{i,j-1/2}^{N}V_{i,j-1/2}^{N+1}\right)\\&-(1-\theta)\frac{\Delta t}{\Delta x}\left(H_{i+1/2,j}^{N}U_{i+1/2,j}^{N}-H_{i-1/2,j}^{N}U_{i-1/2,j}^{N}\right)\\&-(1-\theta)\frac{\Delta t}{\Delta x}\left(H_{i,j+1/2}^{N}V_{i,j+1/2}^{N}-H_{i,j-1/2}^{N}V_{i,j-1/2}^{N}\right),\end{aligned} \quad (7)$$

$$\begin{aligned}U_{i+1/2,j}^{N+1}=&FU_{i+1/2,j}^{N}-(1-\theta)\frac{g\Delta t}{\Delta x}\left(\eta_{i+1,j}^{N}-\eta_{i,j}^{N}\right)\\&-\theta\frac{g\Delta t}{\Delta x}\left(\eta_{i+1,j}^{N+1}-\eta_{i,j}^{N+1}\right)+\Delta t\frac{\gamma_T(U_a-U_{i+1/2,j}^{N+1})}{H_{i+1/2,j}^{N}}\\&-g\Delta t\frac{\sqrt{(U_{i+1/2,j}^{N})^2+(V_{i+1/2,j}^{N})^2}}{Cz_{i+1/2,j}^{2}H_{i+1/2,j}^{N}}U_{i+1/2,j}^{N+1},\end{aligned} \quad (8)$$

$$\begin{aligned}V_{i,j+1/2}^{N+1}=&FV_{i,j+1/2}^{N}-(1-\theta)\frac{g\Delta t}{\Delta y}\left(\eta_{i,j+1}^{N}-\eta_{i,j}^{N}\right)\\&-\theta\frac{g\Delta t}{\Delta y}\left(\eta_{i,j+1}^{N+1}-\eta_{i,j}^{N+1}\right)+\Delta t\frac{\gamma_T(V_a-V_{i,j+1/2}^{N+1})}{H_{i,j+1/2}^{N}}\\&-g\Delta t\frac{\sqrt{(U_{i,j+1/2}^{N})^2+(V_{i,j+1/2}^{N})^2}}{Cz_{i,j+1/2}^{2}H_{i,j+1/2}^{N}}V_{i,j+1/2}^{N+1}.\end{aligned} \quad (9)$$

In the above equations, Δx is the computational volume length in the x direction, Δy is the computational volume length in the y direction, and Δt is the computational time step (Martin and Gorelick, 2005). The parameter θ dictates the degree of implicitness of the solution. Its value ranges between 0.5 and 1, where $\theta=0.5$ means that the approximation is centered in time and $\theta=1.0$ means that the approximation is completely implicit (Casulli and Cheng, 1992).

The operators FU and FV in Eqs. (8) and (9) contain the advective, viscous, and Coriolis components of the governing equations (Martin and Gorelick, 2005).

The value of the Chezy coefficient in Eq. (10) is given in terms of Manning's roughness coefficient Mn, which is assumed to be dimensionless (Martin and Gorelick, 2005). Other details of the discretization process can be found in Martin and Gorelick (2005):

$$C_{z_{i+1/2,j}} = \frac{\left(H_{i+1/2,j}\right)^{1/6}}{Mn_{i+1/2,j}}. \quad (10)$$

3.3 The boundary condition

The model itself identifies the location of water/land boundaries using the following equations (Martin and Gorelick, 2005):

$$H^{N+1}_{i+1/2,j} = \max\left(0, h_{i+1/2,j} + \eta^{N+1}_{i,j}, h_{i+1/2,j} + \eta^{N+1}_{i+1,j}\right), \quad (11)$$

$$H^{N+1}_{i,j+1/2} = \max\left(0, h_{i,j+1/2} + \eta^{N+1}_{i,j}, h_{i,j+1/2} + \eta^{N+1}_{i,j+1}\right). \quad (12)$$

Two types of horizontal boundary conditions have been set:
(i) The projection of the velocity normal to the domain boundary:

$$\frac{\partial U}{\partial t} + U_{\text{upw}} \frac{\partial U}{\partial n} = 0, \quad (13)$$

where U_{upw} is the upwinded normal direction velocity component, and n is the direction normal to the domain boundary (Martin and Gorelick, 2005).

(ii) To limit wave reflections at open boundaries, the following condition is imposed:

$$\frac{\partial \eta}{\partial t} + C_n \frac{\partial \eta}{\partial n} = 0, \quad (14)$$

where C_n is the propagation velocity from grid points around the boundary (Martin and Gorelick, 2005).

4 Data assimilation

4.1 Overview

Data assimilation aims to establish an optimal compromise between the prediction of a computational model and a set of observations. Both the model and the observations are assumed to be incorrect and contain some error. Heuristically, data assimilation takes some weighted average between these two estimates, with weights inversely proportional to the anticipated error in each (Daley, 1991).

Two forms of data assimilation have been common in weather forecasting. In "lumped" data assimilation, the goal is to produce a single initial state for the system to be simulated, from which a subsequent forecast can be launched. In "lumped" data assimilation it has been possible to impose the model state equation – a set of conservation laws – exactly on that initial state, especially when so-called variational assimilation has been used (Le Dimet and Talagrand, 1986; Lewis and Derber, 1985; Courtier and Talagrand, 1987). However, variational data assimilation implicitly contains the same defects that the model it is based on contains – for example some model bias. Bias threatens to throw the model trajectory away from the true physical flow, which is tracked by observations even when they contain some noise.

In continuous data assimilation, on the other hand, the solution of the state equation is constantly "nudged" towards observations (Lorenc, 1986). This means that the state equation is only approximately true and that several conservation properties may get lost, but the model trajectory is likely to always stay close to the true observed trajectory.

The Extended Kalman Filter (EKF) (Kalman, 1960) combines the best properties of both lumped and continuous data assimilation methods. Unfortunately, the computational complexity of the classical EKF is prohibitively high for high-dimensional numerical models such as those appearing in geophysical simulations. However, this situation has started to change with the emergence of Ensemble Kalman filters (EnKF), (Evensen, 1994). Ensemble Kalman filters are stochastic approximations of the Extended Kalman Filter that purport to preserve many of its good properties. In practice, the degree to which this is achieved depends crucially on the particular EnKF variant chosen.

Some variants of Ensemble Kalman filters draw their inspiration from the same Bayesian paradigm as the original Kalman Filter does. A prominent example of these methods is the maximum likelihood estimation filter (MLEF) introduced in Zupanski (2004). MLEF solves a Bayesian minimization with an ensemble of forecasts and uses the Limited Memory BFGS method to minimize a cost function that measures the distance of observations from the forecast. However, it generates a single ensemble of forecasts in the beginning of the forecast and uses it for all the minimzations, unlike the Variational Ensemble Kalman Filter (VEnKF) that will be introduced below. As will become evident from the convergence behavior of VEnKF, re-sampling the ensemble very frequently dramatically improves the convergence of the method and the stability of the corresponding Kalman filter.

4.2 VEnKF

The Variational Ensemble Kalman Filter (VEnKF) is a stochastic approximation of the EKF. In this paper, we restrict ourselves to a brief discussion of the main ideas behind the VEnKF. A more detailed discussion can be found in Solonen et al. (2012). We begin by introducing the following coupled system of stochastic dynamic equations:

$$s_{k+1} = \mathcal{M}_k\left(s_k\right) + \varepsilon_k, \quad (15)$$

$$o_{k+1} = \mathcal{H}_{k+1}\left(s_{k+1}\right) + \zeta_{k+1}, \quad (16)$$

where $\mathcal{M}_k$ is an N-dimensional transition operator used to forecast the model state at time instance $k+1$ given the model state at time instance k; $\mathcal{H}_{k+1}$ is an observation operator that maps the model state s_{k+1} to the observation o_{k+1}; ε_k and ζ_{k+1} are zero mean random terms that define prediction and observation error, respectively. Our task is to define an estimate for s_{k+1} given the operators $\mathcal{M}_k$, $\mathcal{H}_{k+1}$, the observation o_{k+1}, and the covariance matrices of ε_k and ζ_{k+1}, hereafter defined as C_{ε_k} and $C_{\zeta_{k+1}}$. In many cases, ε_k and ζ_{k+1} are also assumed to be normally distributed, although this requirement can be relaxed.

The motivation behind the VEnKF is quite similar to that of the EnKF. Foremost, we compute a sample estimate for the prior covariance, where the samples are explicitly generated by Eq. (15). Thereafter, instead of following the EKF formulas as is done in EnKF, we replace them with a MAP (maximum a posteriori probability) estimate problem. By taking −log of the MAP cost function we arrive at an equivalent minimization problem as suggested in Auvinen et al. (2009):

$$l(s|o_{k+1}) = \frac{1}{2}\left(s - s^{\mathrm{p}}_{k+1}\right)^T \left[C^{\mathrm{p}}_{k+1}\right]^{-1}\left(s - s^{\mathrm{p}}_{k+1}\right) + \frac{1}{2}\left(o_{k+1} - \mathcal{H}_{k+1}(s)\right)^T C^{-1}_{\zeta_{k+1}}\left(o_{k+1} - \mathcal{H}_{k+1}(s)\right). \quad (17)$$

Here s^{p}_{k+1} is the predicted model state at time instance $k+1$ and C^{p}_{k+1} is the covariance matrix of the prediction. When transition and observation operators are linear, it can be proven (Simon, 2006) that a minimum variance unbiased estimator for s_{k+1}, hereafter denoted as s^{est}_{k+1}, minimizes Eq. (17), whereas the covariance matrix of this estimate, which we denote by C^{est}_{k+1}, is defined by the inverse Hessian of Eq. (17). This approach can also be expanded to non-linear cases.

Before giving a rigorous formulation of the VEnKF algorithm, we need to introduce some supporting notation. Let $\left\{e_{k,i}\right\}_{i=1}^{N}$ denote an ensemble of cardinality N sampled from the distribution of s^{est}_k. More precisely, $\forall i\ e_{k,i} \sim \mathcal{N}(s^{\mathrm{est}}_k, C^{\mathrm{est}}_k)$. In addition, we denote the mean of $\left\{e_{k,i}\right\}_{i=1}^{N}$ by $\bar{e}_k$ and introduce an M-element vector $X\left(e_{k,i}\right)$ defined as follows:

$$X\left(e_{k,i}\right) = \left(\left(e_{k,1} - \bar{e}_k\right), \ldots, \left(e_{k,N} - \bar{e}_k\right)\right)/\sqrt{N-1}.$$

The VEnKF algorithm now reads as follows:

i. Compute the model prediction: $s^{\mathrm{p}}_{k+1} = \mathcal{M}_k(s_k)$.

ii. Move the ensemble $\left\{e_{k,i}\right\}_{i=1}^{N}$ forward using Eq. (15):

$$\forall i\ \tilde{e}_{k+1,i} = \mathcal{M}_k\left(e_{k,i}\right).$$

iii. Define the sample estimate for the prior covariance:

$$C^{\mathrm{p}}_{k+1} = X\left(\tilde{e}_{k+1,i}\right)\left(X\left(\tilde{e}_{k+1,i}\right)\right)^T + C_{\varepsilon_k}.$$

iv. Assign s^{est}_{k+1} to the minimizer of the cost function (17) and C^{est}_{k+1} to an approximation of the inverse Hessian of Eq. (17).

v. Update the ensemble $\left\{\tilde{e}_{k+1,i}\right\}_{i=1}^{N}$ by sampling from $\mathcal{N}(s^{\mathrm{est}}_{k+1}, C^{\mathrm{est}}_{k+1})$.

The strength of the VEnKF is that it allows a memory-efficient representation of the prior covariance C^{p}_{k+1}. The latter is advantageous when the model state dimension is too big to allow the explicit storage of the covariance matrices. However, in order to implement steps (iv) and (v) of the presented algorithm, we need to evaluate the cost function defined by Eq. (17) and specify a low-memory approximation for its inverse Hessian. The first goal is achieved by leveraging the Sherman–Morrison–Woodbury formula to invert C^{p}_{k+1}. More precisely:

$$\left[C^{\mathrm{p}}_{k+1}\right]^{-1} = \left[C_{\varepsilon_k} + X\left(\tilde{e}_{k+1,i}\right)\left(X\left(\tilde{e}_{k+1,i}\right)\right)^T\right]^{-1} = C^{-1}_{\varepsilon_k} - C^{-1}_{\varepsilon_k} X\left(\tilde{e}_{k+1,i}\right)\left(I + \left(X\left(\tilde{e}_{k+1,i}\right)\right)^T C^{-1}_{\varepsilon_k} X\left(\tilde{e}_{k+1,i}\right)\right)^{-1} \times \left(X\left(\tilde{e}_{k+1,i}\right)\right)^T C^{-1}_{\varepsilon_k}. \quad (18)$$

Formulation (18) can be directly inserted into cost function (17), so it is not necessary to store the full matrices in order to evaluate it. Since the matrix C_{ε_k} is usually specified by a simplified (diagonal or block-diagonal) structure, and the ensemble size is assumed to be much smaller than the model state dimension, the inversion operations in Eq. (18) are feasible. Reverting back to the implementation of step (v) of the VEnKF algorithm, we suggest approximating the inverse Hessian of Eq. (17) by either the full rank low-memory representation employed by the L-BFGS unconstraint optimizer (Nocedal and Wright, 1999) or the reduced rank representation in Krylov space generated by conjugate gradient minimization of Eq. (17) (Bardsley et al., 2013).

The computational complexity of the VEnKF algorithm remains linear in the number of degrees of freedom of the model despite the fact that it solves a minimization problem with observations every time step. This follows from the fact that the minimization problem is very well-conditioned and converges in a small number of iterations. This number is likely to remain independent of the number of degrees of freedom, because the minimization in VEnKF is identical to that applied in three-dimensional variational data assimilation that has been observed to have this behavior in operational weather data assimilation. This good behavior comes from the fact that in the current application scenario the assimilation window is just one time step long, and therefore the Hessian matrix remains diagonally dominated and the initial guess very good, in the same manner as in the minimization applied in implicit time-stepping schemes. The convergence history of the residual of minimization indeed shows fast linear convergence, as seen in Fig. 2.

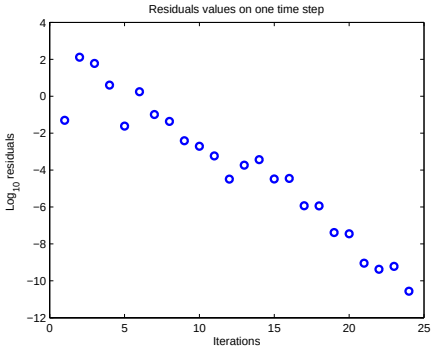

Fig. 2. Sample residuals plotted for one time step.

5 Dam break experiments

One of the applications published in Martin and Gorelick (2005) is a dam break experiment. The experiment consists of a flume 21.2 m long and 1.4 m wide. The flume is closed at one end and open at the other end. It has a curved constriction 5.0 m from the closed end that ends 4.7 m from the open end. A dam is located 8.5 m from the closed end, with an opening of width 0.6 m. The flume has a slope of 0.002 with water at a height of 0.15 m behind the dam (Martin and Gorelick, 2005).

Wave meters and pressure transducers were located at 8 locations, as seen in Fig. 3; however, the water depth measurements were only given in seven locations. The recorded water depths last for 62 s after the dam is broken. With the dam position chosen as the origin, the wave meters are located at places $x = -8.5$, -4.5, and -0.0 m, and the pressure transducers are placed at $x = +0.0$, +2.5, +5.0, +7.5, and +10.0 m (Martin and Gorelick, 2005). The computational time step used in the experiment is $\Delta t = 0.103$ s and the grid dimensions are $\Delta x = 0.05$ m and $\Delta y = 0.125$ m. With this grid cell size, the geometry is sliced into 30×171 grid cells. Finally, simulated water heights at the seven locations were compared with heights measured by the wave meters and pressure transducers.

5.1 VEnKF parameters

The state vector for the assimilation is defined as the vector of heights at the center of a grid point. The complete state vector comprises the free surface elevation η and the horizontal velocities u in the x direction and v in the y direction for the entire domain, i.e., $s = [\eta \;\; u \;\; v]^T$. The model has, therefore, altogether 16 000 spatial degrees of freedom. With the interpolations described in Sect. 5.3, the ensembles are sampled in every time step of the assimilation. The observation error and the model error covariance matrices are both assumed

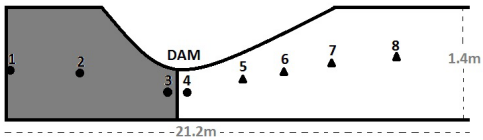

Fig. 3. Plan view flume layout for the dam break experiment of Bellos et al. (1992). Wave meter locations are displayed with circles. Pressure transducer locations are displayed with triangles (Martin and Gorelick, 2005).

to be diagonal. The observation operator $\mathcal{H}$ in Eq. (17) is a linear operator that maps the state vector to the observation space corresponding to all grid points covered by the interpolated data, but restricted to the water height values only.

5.2 Experiment 1: VEnKF application to dam break with synthetic data

The aim of this experiment is to examine both qualitative and quantitative characteristics of the VEnKF method to the dam break experiment. The data set has been generated by adding normally distributed noise with mean 0 and a variance of 5×10^{-2} from the solution of the model simulation. To make the experiment more realistic, data has been picked in 8 positions corresponding to wave meter locations defined in Martin and Gorelick (2005). More precisely, the time interval between the data in all locations was fixed, but chosen randomly for every location. This setup emulates the fact that wave meters do not necessarily collect information at the same time.

5.2.1 Results for experiment 1

In Figs. 4 and 5, the matching between the data, VEnKF estimates (50 ensembles) and the model simulation, here referred to as the truth, is shown for all 8 wave meter locations. The target of the current study is not really data assimilation for the purpose of a subsequent forecast with VEnKF, as would be the case in an atmospheric dynamics context, but instead qualitatively better hind-casting of a catastrophic event such as a dam breaking down, with an ensemble-based approach. In our case, the length of each forecast is therefore just one computational time step at a time with interpolated observations. This close match between the model and observations over one time step also results in very compact ensemble spreads, as can be seen in Fig. 6.

Figure 7 shows a plot of root mean square error (RMSE) for the entire duration of the simulation. The error first increases, then decreases steadily. Such error behavior can be explained by the fact that at the begining of the dam break, a steady initial state quickly breaks into a turbulent flume that then peters out to a drib as the water eventually runs out. In the forecast skill plot shown in Fig. 8, we have plotted the forecast skill from a time about 4 s after the dam break for

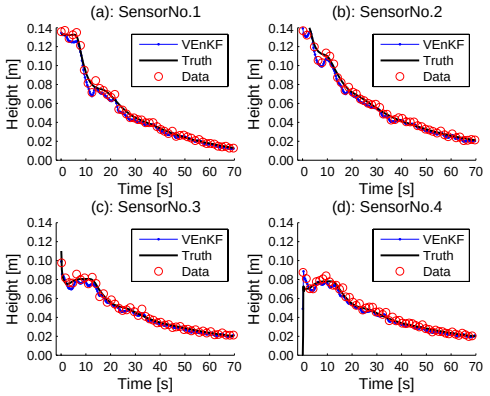

Fig. 4. Experiment 1: Comparison of VEnKF estimates, true water depth and data of the dam break experiment for the first four sensors at the upstream end.

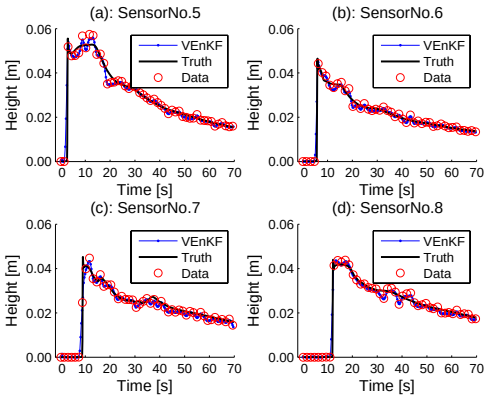

Fig. 5. Experiment 1: Comparison of VEnKF estimates, true water depth and data of the dam break experiment for the last four sensors.

a period of 10 s from the begining of the experiment, as this period is the best one to display the error before water has started to run out from the flume.

5.3 Experiment 2: VEnKF for dam break experiment with real data

The published data set (see Martin and Gorelick, 2005) of measurements is used to assimilate water heights. The high sparsity of the set of measurements is challenging for data assimilation. Observations come at an average rate of 1.6 observations in one or more locations per time step and at a maximum of 5 locations per time step. They feature only measurements of water height. This means that the number of observations in relation to the dimension of state space is approximately $1/100\,000$, since the computational time step is 0.1 s. For this reason, the observations are interpolated in time by a spline scheme and in space by a Gaussian mask to make them dense enough for the VEnKF assimilation scheme. This means we interpolate the observations in time and extrapolate each of them in space by a Gaussian kernel. After this, the ratio of the number of observations to system dimension improves to $1/50$.

5.3.1 Shore boundary definition for the VEnKF

As can be seen from the MAP estimate problem (17), the VEnKF does not account for additional prior knowledge beyond the observations. This means that in the case of problems on bounded domains, there is no way to include information about the boundaries in the Kalman filter analysis. If the prediction model automatically maintains the boundaries in accordance with defined constraints, one can simply

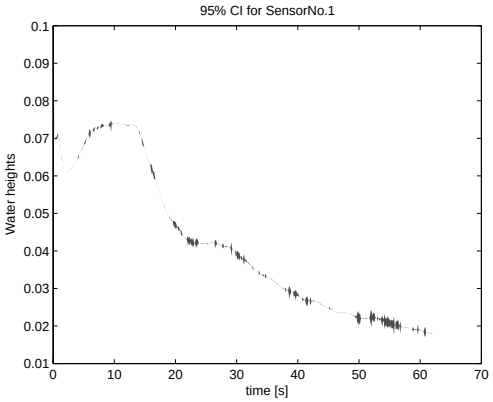

Fig. 6. 95 % confidence region is shown for the location of Sensor No. 1.

reduce the data assimilation analysis to the inner part of the model domain. However, this approach is complicated when the boundaries change over time.

In our experiments, we use a strategy that allows us to account more flexibly for evolving boundaries, albeit without guaranteeing that the boundaries will be preserved exactly as required by the model constraints. Information about the boundaries is included in the model uncertainty description, i.e., in the model error covariance C_{ε_k} (see Eq. 15). This changes the analytical representation of the boundaries to a probabilistic description. Thus, there is no absolute certainty about where the boundaries are located, but there is more confidence about the evolution of boundaries than that of the model.

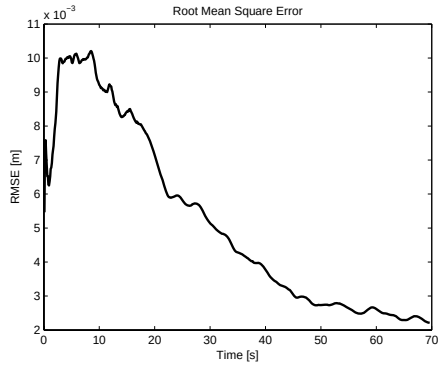

Fig. 7. The RMSE plot for the entire time of assimilation.

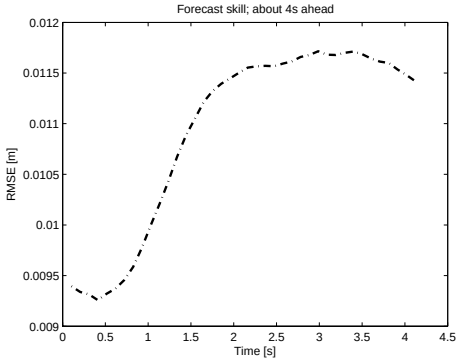

Fig. 8. The forecast skill plot for about 4 s period of forecast

In the dam break case studied here, we have prior knowledge about the shoreline and there will be no water in places where there is a riverbank. Therefore, we define the model covariance C_{ε_k} such that the state elements that are confined to the riverbank have variances much smaller than the variances assigned to the rest of the state. This strategy shifts the responsibility of maintaining the boundaries to the data assimilation analysis.

6 Model applications with real data

6.1 Results without data assimilation

The results of direct model simulations with MOD-FreeSurf2D show that the simulated water depth matches well with the measured depth only for the three measurement locations above the dam (at the upstream end), as can be seen in Fig. 9a–c. The results are less accurate for other locations below the dam due to the emergence of super-critical flows in the downstream end. The downstream end is also characterized by turbulent flow, and the model only tracks the height of water, but not the turbulent fine structure of the flow. In Fig. 9d as well as Fig. 10a and b in particular, we can see the discontinuity of the flow at the beginning of the dam opening.

6.2 Results with data assimilation

The model error and the observation error covariance matrices were set to $\mathbf{C}_{\varepsilon_k^p} = (0.0011)^2\mathbf{I}$ and $\mathbf{C}_{\zeta_k} = (0.001)^2\mathbf{I}$, respectively. We use the initial estimate of the state x_0^{est} equals the initial height of water and the initial covariance estimate $C_0^{\text{est}} = \mathbf{I}$.

Measurements are incorporated into the model, and the assimilation is done with an ensemble size of 75 members. The number of LBFGS iterations and stored vectors is set to 25. When the dam is removed, a strong flood wave is generated

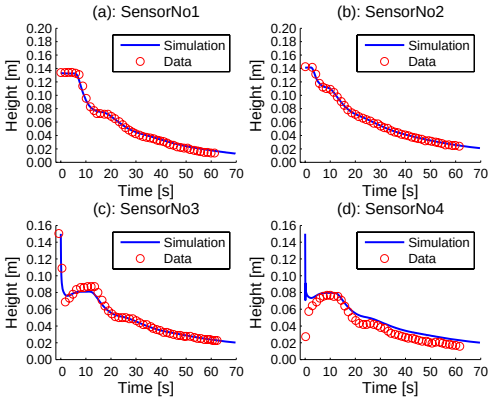

Fig. 9. Comparison of simulated water depth and measured depth of the dam break experiment for the first four sensors at the upstream end.

and propagated downstream from the flume. The variation of water depth with time is compared with experimental data as given by the seven sensors; see Figs. 11 and 12.

From the results it can be seen that at the location immediately after the dam (location 4), the original simulation did not capture the behavior of the flow at the beginning of the simulation. However, VEnKF is able to approximate the water height and the structure of the flow. The same situation was observed in locations 5, 6 and 8, where the VEnKF result captures well the most prominent features of the flow.

At all seven sensors located at the upstream and downstream ends with available measurements, these measurements agree well with the VEnKF results. This demonstrates the capability and accuracy of the VEnKF for predicting dam break flows for rivers and streams.

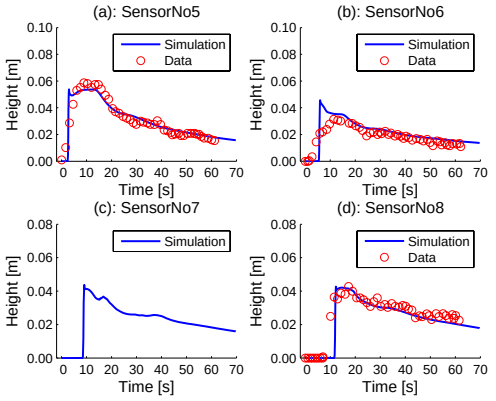

Fig. 10. Comparison of simulated water depth and measured depth of the dam break experiment for the last four sensors at the downstream end. Sensor No. 7 did not have measurements.

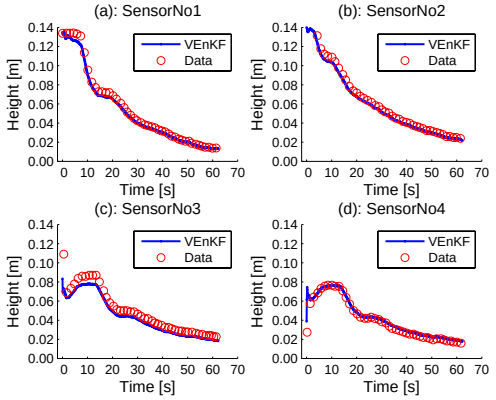

Fig. 11. Experiment 2: Comparison of VEnKF results and measured depth of the dam break experiment for the first four sensors at the upstream end.

It is worth pointing out the time series of water heights at sensor 7 that did not provide any measurements because of a sensor malfunction. If we compare the simulated curve in Fig. 10c with direct simulation to that of Fig. 12c with data assimilation, we see that the latter contains similar fine scale oscillations due to small waves as the sensors with observations, but that these oscillations are missing in Fig. 10c.

This demonstrates that the qualitative improvements towards a more realistic representation of the flume are not restricted to sites with observations, but are indeed spread throughout the computational domain. This can be seen in more detail in the accompanying videos that represent the

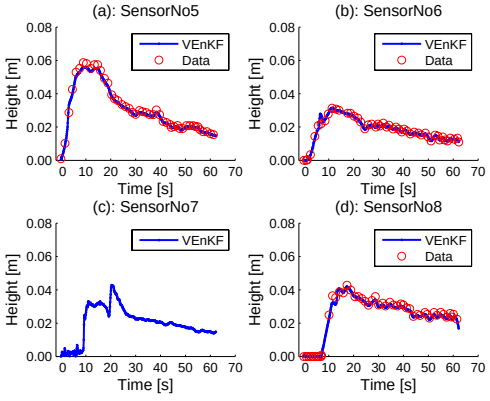

Fig. 12. Experiment 2: Comparison of VEnKF results and measured depth of the dam break experiment for the last four sensors at the downstream end. Sensor No. 7 did not have measurements.

pure simulation and the flume obtained with data assimilation.

7 Discussion and conclusions

The use of data assimilation to complement forecasts made by mathematical models with observational data is a growing trend in scientific computing. This trend is likely to continue, since the computer capacity accessible to researchers is increasing rapidly, and many different kinds of automatic measurement devices are becoming available that provide large numbers of measurements from target systems.

In the foregoing sections, we have demonstrated some of the benefits that data assimilation can bring to hydrological modeling. The resulting analysis of a dam break flume behaves in a more realistic manner than the corresponding computer simulation alone. It displays turbulent behavior such as the real flow, features prominent hydraulic jumps, and avoids several numerical artifacts.

Another benefit of data assimilation is a proper statistical treatment of flume simulations. Traditional mathematical models are deterministic, whereas in computer simulation we can only approximate a real physical phenomenon in a statistical sense. For this reason, the adoption of a version of Kalman filtering, the Variational Ensemble Kalman Filter (VEnKF), adds value to the simulation, as it automatically incorporates information about the expected error covariance of the analysis of the flume into the approximate error covariance matrix that it computes in the course of data assimilation. Continuous data assimilation therefore addresses qualitative defects in flow simulations and correctly interprets simulated numerical values as samples from a distribution of possible physical values, not as true physical values.

Acknowledgements. Thanks to Lappeenranta University of Technology and the World Bank Project funding through the University of Dar es salaam and Dar es salaam University College of Education, without these funding this work would be impossible. Our thanks also go to the anonymous reviewers for their insightful review of our work.

Edited by: R. Buizza
Reviewed by: two anonymous referees

References

Aliparast, M.: Two-dimensional finite volume method for dam-break flow simulation, Int. J. Sediment Res., 24, 99–107, 2009.

Auvinen, H., Bardsley, J., Haario, H., and Kauranne, T.: The variational Kalman filter and an efficient implementation using limited memory BFGS, Int. J. Numer. Meth. Fl., 64, 314–335, 2009.

Baghlani, A.: Simulation of dam-break problem by a robust flux-vector splitting approach in Cartesian grid, Scientia Iranica, 18, 1061–1068, 2011.

Barcena, J. F., Garcia, A., Garcia, J., Alvares, C., and Revilla, J. A.: Surface analysis of free surface and velocity to changes in river flow and tidal amplitude on a shallow mesotidal estuary: An application in Suances Estuary (Nothern Spain), J. Hydrol., 420–421, 301–318, 2012.

Bardsley, J., Solonen, A., Parker, A., Haario, H., and Howard, M.: An Ensemble Kalman Filter Using the Conjugate Gradient Sampler, Int. J. Uncert. Quant., 3, 357–370, doi:10.1615/Int.J.UncertaintyQuantification.2012003889, 2013.

Bates, P. and Anderson, M.: A two-dimensional finite-element model for river flow inundation, P. Roy. Soc. Lond. A. Mat., 440, 481–491, doi:10.1098/rspa.1993.0029, 1993.

Bellos, C., Soulis, J., and Sakkas, J.: Computation of two-dimensional dam-break induced flows, Adv. Water Resour., 14, 31–41, 1991.

Casulli, V.: A Semi-implicit finite difference method for non hydrostatic, free-surface flows, Int. J. Numer. Meth. Fl., 30, 425–440, 1999.

Casulli, V. and Cheng, R.: Semi-implicit finite difference methods for three-dimensional shallow water flow, Int. J. Numer. Meth. Fl., 15, 629–648, doi:10.1002/fld.1650150602, 1992.

Courtier, P. and Talagrand, O.: Variational assimilation of meteorological observations with the adjoint vorticity equation, Part 2: Numerical results, Q. J. Roy. Meteor. Soc., 113, 1329–1368, 1987.

Daley, R.: Atmospheric Data Analysis, Cambridge University Press, 1st Edn., 1991.

Drikakis, D. and Tsangaris, S.: On the solution of the compressible Navier-Stokes equations using improved flux vector splitting methods, Appl. Math. Model., 17, 282–297, 1993.

Erpicum, S., Dewals, B., Archambeau, P., and Pirotton, M.: Dam break flow computation based on an efficient flux vector splitting, J. Comput. Appl. Math., 234, 2143–2151, 2010.

Evensen, G.: Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte-Carlo methods to forecast error statistics, J. Geophys. Res., 99, 10143–10162, 1994.

Goldstein, S.: Modern developments in fluid mechanics, Oxford Univ. Press, 1938.

Heniche, M., Secretan, Y., Boudreau., P., and Leclerc, M.: Two-dimensional finite volume model for dam-break simulation, Adv. Water Resour., 23, 359–372, 2000.

Kalman, R.: A new approach to linear filtering and prediction problems. Transaction of the ASME, J. Basic Eng.-T. ASME, 82, 35–45, 1960.

Le Dimet, F.-X. and Talagrand, O.: Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects, Tellus, 38A, 97–110, 1986.

Lewis, J. M. and Derber, J. C.: The use of adjoint equations to solve a variational adjustment problem with advective constraints, Tellus, 37A, 309–322, 1985.

Lorenc, A.: Analysis methods for numerical weather prediction, Q. J. Roy. Meteor. Soc., 112, 1177–1194, 1986.

Martin, N. and Gorelick, S. M.: MODFreeSurf2D: A MATLAB surface fluid flow model for rivers and streams, Comput. Geosci., 31, 926–946, 2005.

Nocedal, J. and Wright, S.: Numerical Optimization, chap. Limited-Memory BFGS, Springer-Verlag, New York, 224–227, 1999.

Simon, D.: Optimal state estimation, Kalman, H_∞, and nonlinear approaches, Wiley-Interscience, Hoboken, 2006.

Solonen, A., Haario, H., Hakkarainen, J., Auvinen, H., Amour, I., and Kauranne, T.: Variational ensemble Kalman filtering using limited memory BFGS, Electron T. Numer. Ana., 39, 271–285, 2012.

Toro, E. and Vazquez-Cendon, M.: Flux splitting schemes for the Euler equations, Comput. Fluids, 70, 1–12, 2012.

Ying, X., Jorgeson, J., and Wang, S.: Modeling Dam-Break flows using Finite Volume method on unstructured grid, Eng. Appl. Comput. Fluid Mech., 3, 184–194, 2009.

Yoon, T. and Kang, S.: Finite volume model for two-dimensional shallow water flows on unstructured grids, J. Hydraul. Eng.-ASCE, 130, 678–688, 2004.

Zhang, M. and Wu, W.: A two dimensional hydrodynamic and sediment transport model for dam break based on finite volume method with quadtree grid, Appl. Ocean Res., 33, 297–308, 2011.

Zupanski, M.: Maximum Likelihood Ensemble Filter: Theoretical Aspects, Mon. Weather Rev., 133, 1710–1726, 2004.

Publication IV

Solonen, A., Hakkarainen, J., Ilin, A., Abbas, M., and Bibov, A. (2014) Estimating model error covariance matrix parameters in extended Kalman filtering. *Nonlinear Processes in Geophysics*, 21, pp. 919-927.

Nonlin. Processes Geophys., 21, 919–927, 2014
www.nonlin-processes-geophys.net/21/919/2014/
doi:10.5194/npg-21-919-2014

Nonlinear Processes in Geophysics
Open Access

Estimating model error covariance matrix parameters in extended Kalman filtering

A. Solonen[1,2], J. Hakkarainen[2], A. Ilin[3], M. Abbas[3], and A. Bibov[1]

[1]Lappeenranta University of Technology, Lappeenranta, Finland
[2]Finnish Meteorological Institute, Helsinki, Finland
[3]Aalto University, Helsinki, Finland

Correspondence to: A. Solonen (antti.solonen@gmail.com)

Received: 8 August 2013 – Revised: 19 May 2014 – Accepted: 29 July 2014 – Published: 1 September 2014

Abstract. The extended Kalman filter (EKF) is a popular state estimation method for nonlinear dynamical models. The model error covariance matrix is often seen as a tuning parameter in EKF, which is often simply postulated by the user. In this paper, we study the filter likelihood technique for estimating the parameters of the model error covariance matrix. The approach is based on computing the likelihood of the covariance matrix parameters using the filtering output. We show that (a) the importance of the model error covariance matrix calibration depends on the quality of the observations, and that (b) the estimation approach yields a well-tuned EKF in terms of the accuracy of the state estimates and model predictions. For our numerical experiments, we use the two-layer quasi-geostrophic model that is often used as a benchmark model for numerical weather prediction.

1 Introduction

In state estimation, or data assimilation, the goal is to estimate the dynamically changing state of the model, given incomplete and noisy observations. The estimation is usually carried out sequentially: the model prediction made from the previous state is updated with the new observations that become available. State estimation methods need a description of the error that the forward model makes in an assimilation step: otherwise the erroneous model prediction is overweighted when it is combined with new observations, potentially leading to divergence of the method. From the perspective of data assimilation, the model error representation can be viewed as a tunable quantity that has an effect on the performance of the method.

The extended Kalman filter (EKF) is a popular nonlinear data assimilation method. It is a nonlinear extension of the Kalman filter (KF; Kalman, 1960), where the forward model and observation operators are linear, and the model and observation errors are assumed to be additive and normally distributed, which yields direct matrix formulas for updating the model state with observations. In EKF, the nonlinear forward model and the observation operator are linearized, and the KF formulas are applied. The goal in this paper is to study a technique for estimating a parameterized version of a model error covariance matrix.

The model error tuning problem can be viewed as a parameter estimation problem in state space models. One way to estimate static parameters in dynamical state space models is to compute the likelihood of the parameters by "integrating out" the uncertain model state using a filtering technique such as EKF. This "filter likelihood" technique is a well-known tool for parameter estimation in stochastic differential equation (SDE) models (Singer, 2002) and time series analysis (Durbin and Koopman, 2001). In Hakkarainen et al. (2012), the approach was used to estimate parameters of chaotic models. As noted in Hakkarainen et al. (2012), the same approach can be used for estimating the parameters of a model error covariance matrix. In this paper, we study this possibility further. The technique needs a parametric representation of the model error covariance matrix, which can range from something very simple (e.g., diagonal) to complicated representations that take into account, for instance, spatial correlation structures.

The presented approach can be thought of as a generalization of the online error covariance parameter estimation method of Dee (1995), where the model error covariance

matrix parameters are estimated at each assimilation step using a single batch of observations that become available at that step. In the approach presented here, data from several assimilation steps are gathered in the likelihood.

We test the method with the 1600-dimensional two-layer quasi-geostrophic (QG) benchmark model (Pedlosky, 1987) with synthetic data. We show that (a) the importance of the model error covariance matrix calibration depends on the quality of the observations, and that (b) the estimation approach yields a well-tuned EKF in terms of root mean squared (rms) errors of the state estimates and model predictions.

In such a synthetic case, the truth is known, so the true model error can be studied by computing differences between the truth and the forecast model. However, we observe that the true model error is not optimal with respect to the performance of the filter. This happens because filtering methods make certain assumptions and approximations, and the effect of these can be compensated for by appropriately choosing the model error term in the filter. This issue is discussed further in Hakkarainen et al. (2013).

2 Likelihood via filtering

We start this section by introducing how the parameter likelihood can be computed via filtering methods. We introduce briefly the general formulas first, and then proceed to the specific case when EKF is applied. For more details about state estimation theory in the general setting, and about parameter estimation within state space models, refer to, e.g., Särkkä (2013).

Let us consider the following general state space model at time step k with unknown parameters $\boldsymbol{\theta}$:

$$\boldsymbol{x}_k \sim p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{\theta}) \tag{1}$$

$$\boldsymbol{y}_k \sim p(\boldsymbol{y}_k|\boldsymbol{x}_k) \tag{2}$$

$$\boldsymbol{\theta} \sim p(\boldsymbol{\theta}). \tag{3}$$

In addition to the unknown dynamically changing model state $\boldsymbol{x}_k$, we thus have unknown static parameters $\boldsymbol{\theta}$, from which we might have some prior information $p(\boldsymbol{\theta})$. The goal in parameter estimation, in Bayesian terms, is to find the posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n})$ of the parameters, given a fixed data set $\boldsymbol{y}_{1:n}$. According to the Bayes formula, the posterior is proportional to the product of the likelihood and the prior: $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n}) \propto p(\boldsymbol{y}_{1:n}|\boldsymbol{\theta})\, p(\boldsymbol{\theta})$. Here, the notation $\boldsymbol{y}_{1:n} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n\}$ means all observations for n time steps. In the prior term $p(\boldsymbol{\theta})$, we can include things that we know about $\boldsymbol{\theta}$ before collecting any data, such as physical bounds for the parameter values. Here, we assume that the prior is given, and concentrate on the computation of the likelihood $p(\boldsymbol{y}_{1:n}|\boldsymbol{\theta})$, which is nontrivial in the case of a state space model.

The general state space model notation given above can be somewhat unfamiliar to readers in the atmospheric sciences community, and some clarification may be useful. The first equation basically contains the probabilistic model for propagating the state forward: it gives the probability density for the state $\boldsymbol{x}_k$, given the value for the previous state $\boldsymbol{x}_{k-1}$ and the model parameters $\boldsymbol{\theta}$. The second equation contains the observation model; it gives the probability density for observing the value $\boldsymbol{y}_k$, given a value for the current state $\boldsymbol{x}_k$. Presentation of filtering theory often starts from some assumptions of the forms of these densities (such as Gaussian). For the purpose of parameter estimation, it is instructive to develop the likelihood first in a general state space model setup.

Filtering methods (particle filters, Kalman filters, etc.) estimate the dynamically changing model state sequentially. They give the marginal distribution of the state, given the measurements obtained until the current time k. For a given value for $\boldsymbol{\theta}$, filtering methods thus target $p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}, \boldsymbol{\theta})$. Filters work by iterating two steps: prediction and update. In the prediction step, the current distribution of the state is evolved with the dynamical model to the next time step. In the general notation, the predictive distribution is given by the integral

$$p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1}, \boldsymbol{\theta}) = \int p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, \boldsymbol{\theta})\, p(\boldsymbol{x}_{k-1}|\boldsymbol{y}_{1:k-1}, \boldsymbol{\theta})\, \mathrm{d}\boldsymbol{x}_{k-1}, \tag{4}$$

which is known as the Chapman–Kolmogorov equation. When the new observation $\boldsymbol{y}_k$ is obtained, the model state is updated using the Bayes rule with the predictive distribution $p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1}, \boldsymbol{\theta})$ as the prior:

$$p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k}, \boldsymbol{\theta}) \propto p(\boldsymbol{y}_k|\boldsymbol{x}_k, \boldsymbol{\theta})\, p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1}, \boldsymbol{\theta}). \tag{5}$$

This posterior is used inside the integral (Eq. 4) to obtain the prior for the next time step.

Using the marginal state posteriors obtained in the filtering method, it is also possible to compute the predictive distribution of the next observation. For observation $\boldsymbol{y}_k$, the predictive distribution, given all previous observations, can be written as

$$p(\boldsymbol{y}_k|\boldsymbol{y}_{1:k-1}, \boldsymbol{\theta}) = \int p(\boldsymbol{y}_k|\boldsymbol{x}_k, \boldsymbol{\theta})\, p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1}, \boldsymbol{\theta})\, \mathrm{d}\boldsymbol{x}_k. \tag{6}$$

The term $p(\boldsymbol{x}_k|\boldsymbol{y}_{1:k-1}, \boldsymbol{\theta})$ in the integral is the predictive distribution given by Eq. (4).

Let us now proceed to the original task of estimating static parameters $\boldsymbol{\theta}$ from observations $\boldsymbol{y}_{1:n}$, i.e., computing the posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n}) \propto p(\boldsymbol{y}_{1:n}|\boldsymbol{\theta})\, p(\boldsymbol{\theta})$. Applying the chain rule for joint probability, we obtain

$$p(\boldsymbol{y}_{1:n}|\boldsymbol{\theta}) = p(\boldsymbol{y}_1|\boldsymbol{\theta}) \prod_{k=2}^{n} p(\boldsymbol{y}_k|\boldsymbol{y}_{1:k-1}, \boldsymbol{\theta}). \tag{7}$$

The likelihood of the whole data set $\boldsymbol{y}_{1:n}$ can thus be calculated as the product of the predictive distributions of the

individual observations. In the filtering context, the predictive distributions $p(\boldsymbol{y}_k|\boldsymbol{y}_{1:k-1}, \boldsymbol{\theta})$ are calculated based on the marginal posterior of the states; see Eq. (6).

The integrals required to construct the likelihood above are usually computationally intractable. In this paper, we use EKF as the filtering method to compute an approximation of the likelihood. We thus now write the state space model in a more familiar form:

$$\boldsymbol{x}_k = \mathcal{M}(\boldsymbol{x}_{k-1}) + \boldsymbol{E}_k(\boldsymbol{\theta}) \tag{8}$$

$$\boldsymbol{y}_k = \mathcal{K}(\boldsymbol{x}_k) + \boldsymbol{e}_k, \tag{9}$$

where $\mathcal{M}$ is the forward model and $\mathcal{K}$ is the observation operator. Note that the unknown parameters $\boldsymbol{\theta}$ now appear in the model error $\boldsymbol{E}_k$. In Kalman filtering, it is assumed that model and observation errors are zero mean Gaussian, and that the state and the model error are uncorrelated. Let us assume that the covariance matrix of the model error is parametrized – $\boldsymbol{E}_k(\boldsymbol{\theta}) \sim N(\mathbf{0}, \mathbf{Q}_k(\boldsymbol{\theta}))$ – and that the observation error covariance matrix is known – $\boldsymbol{e}_k \sim N(\mathbf{0}, \mathbf{R}_k)$. Now we can apply EKF, and we get the following expression for the predictive distributions needed in the filter likelihood Eq. (7):

$$\boldsymbol{y}_k|\boldsymbol{y}_{1:k-1}, \boldsymbol{\theta} \sim N\left(\mathcal{K}\left(\boldsymbol{x}_k^{\mathrm{p}}\right), \mathbf{C}_k^{y}\right), \tag{10}$$

where $\boldsymbol{x}_k^{\mathrm{p}}$ is the predicted state from the previous time and $\mathbf{C}_k^{y} = \mathbf{K}_k \mathbf{C}_k^{\mathrm{p}} \mathbf{K}_k^T + \mathbf{R}_k$ is the covariance matrix of the predictive distribution, containing both the model prediction covariance matrix $\mathbf{C}_k^{\mathrm{p}}$ and the observation error covariance matrix $\mathbf{R}_k$. The matrix $\mathbf{K}_k$ is the linearized observation model. In EKF, the prediction covariance matrix is computed as $\mathbf{C}_k^{\mathrm{p}} = \mathbf{M}_k \mathbf{C}_{k-1}^{\mathrm{est}} \mathbf{M}_k^T + \mathbf{Q}_k(\boldsymbol{\theta})$, where $\mathbf{M}_k$ is the linearized forward model, $\mathbf{C}_{k-1}^{\mathrm{est}}$ is the covariance estimate of the previous time step and $\mathbf{Q}_k(\boldsymbol{\theta})$ is the parameterized model error covariance matrix.

Now, applying the general formula (Eq. 7) to the EKF case, the likelihood of observing data $\boldsymbol{y}_{1:n}$ given parameters $\boldsymbol{\theta}$ can be written as

$$p(\boldsymbol{y}_{1:n}|\boldsymbol{\theta}) \propto \exp\left(-\frac{1}{2}\sum_{k=1}^{n}\left[\boldsymbol{r}_k^T \left(\mathbf{C}_k^{y}\right)^{-1} \boldsymbol{r}_k + \log|\mathbf{C}_k^{y}|\right]\right), \tag{11}$$

where $\boldsymbol{r}_k = \boldsymbol{y}_k - \mathcal{K}(\boldsymbol{x}_k^{\mathrm{p}})$ are the prediction residuals and $|\cdot|$ denotes the matrix determinant. The normalization "constants" of the likelihood terms depend on $\boldsymbol{\theta}$ through the covariances $\mathbf{C}_k^{\mathrm{p}}$, and the determinant term therefore needs to be included. Note that the above likelihood is only an approximation to the true likelihood (Eq. 7), and the accuracy of this approximation depends on how well the EKF assumptions (linear model used in error propagation, model error assumed independent of the state) are met. In practice, using the EKF likelihood in parameter estimation often yields good results; see, for instance, Singer (2002), Hakkarainen et al. (2012), Mbalawata et al. (2013) and the numerical examples of this paper.

In Dee (1995), model error covariance parameters were estimated online at each assimilation time step, using only the observations that become available at that specific step. In our notation, this would correspond to having just one term in the exponent of Eq. (11) instead of a sum; that is, the approach presented here can be thought of as a generalization of the approach in Dee (1995), where data from several assimilation steps are gathered in the likelihood.

Note that we could also include some parameters in the forward model, and we would have $\mathcal{M}(\boldsymbol{x}_k, \boldsymbol{\theta})$ instead of $\mathcal{M}(\boldsymbol{x}_k)$. In this paper, we focus only on the model error parameters, but the same technique also applies to model parameters, as demonstrated in Hakkarainen et al. (2012). In principle, we could also assume a model error with a nonzero mean and estimate the parameterized mean of the model error as well, and possibly correct for systematic bias in the model, but this possibility is not pursued further here.

This method assumes that there is a parametric representation of the model error covariance $\mathbf{Q}_k(\boldsymbol{\theta})$ available. In the examples presented in this paper, the model error is assumed to be static over time; we have $\mathbf{Q}_k(\boldsymbol{\theta}) = \mathbf{Q}(\boldsymbol{\theta})$ for all time steps k.

3 Numerical experiments with the two-layer quasi-geostrophic model

3.1 Model description

The two-layer quasi-geostrophic model simulates atmospheric flow for the geostrophic (slow) wind motions. This model can be used as a benchmark for data assimilation in numerical weather prediction (NWP) systems, as it supports some features common to operational weather models, such as baroclinic instability. At the same time, the QG model has a relatively low computational complexity, and requires no special hardware to run. The geometrical domain of the model is specified by a cylindrical surface vertically divided into two "atmospheric" layers that can interact through the interface between them. The model also accounts for an orographic component that defines the surface irregularities affecting the bottom layer of the model. When the geometrical layout of the two-layer QG model is mapped onto a plane, it appears as shown in Fig. 1. In the figure, parameters U_1 and U_2 denote mean zonal flows in the top and the bottom atmospheric layers, respectively.

The model operates in terms of potential vorticity and stream function, where the latter one is analogous to pressure. The assumption of quasi-geostrophic motion leads to a coupled system of partial differential equations (PDEs) describing a conservation law for potential vorticity, given as

$$\frac{\mathcal{D}_1 q_1}{\mathcal{D}t} = 0, \ \frac{\mathcal{D}_2 q_2}{\mathcal{D}t} = 0, \tag{12}$$

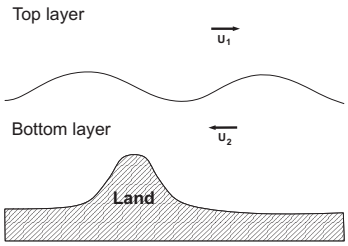

Figure 1. Geometrical layout of the two-layer quasi-geostrophic model.

where $\mathcal{D}_i$ denote the substantial derivatives for latitudinal wind u_i and longitudinal wind v_i, defined as $\frac{\mathcal{D}_i \cdot}{\mathcal{D}t} = \frac{\partial \cdot}{\partial t} + u_i \frac{\partial \cdot}{\partial x} + v_i \frac{\partial \cdot}{\partial y}$, q_i denote the potential vorticity functions, and index i specifies the top atmospheric layer ($i = 1$) and the bottom layer ($i = 2$). Interaction between the layers, as well as the relation between the potential vorticity q_i and the stream function ψ_i, is modeled by the following system of PDEs:

$$q_1 = \nabla^2 \psi_1 - F_1 (\psi_1 - \psi_2) + \beta y, \tag{13}$$

$$q_2 = \nabla^2 \psi_2 - F_2 (\psi_2 - \psi_1) + \beta y + R_s. \tag{14}$$

Here, R_s and β denote the dimensionless orography component and the northward gradient of the Coriolis parameter, which we hereafter denote as f_0. The relations between the model physical attributes and dimensionless parameters that appear in Eqs. (13)–(14) are as follows:

$$F_1 = \frac{f_0^2 L^2}{g' D_1},\ F_2 = \frac{f_0^2 L^2}{g' D_2},\ g' = g \frac{\Delta\theta}{\overline{\theta}},$$
$$R_s = \frac{S(x, y)}{\eta D_2},\ \beta = \beta_0 \frac{L}{U},$$

where D_1 and D_2 are the layer depths, $\Delta\theta$ defines the potential temperature change across the layer interface, $\overline{\theta}$ is the mean potential temperature, g is the acceleration of gravity, $\eta = \frac{U}{f_0 L}$ is the Rossby number associated with the defined system, and $S(x, y)$ and β_0 are dimensional representations of $R_s(x, y)$ and β, respectively.

The system of Eqs. (12)–(14) defines the two-layer quasi-geostrophic model. The state of the model, and thus the target of the estimation, is the stream function ψ_i. For the numerical solution of the system, we consider potential vorticity functions q_1 and q_2 to be known, and invert the spatial Eqs. (13) and (14) for ψ_i. More precisely, we apply ∇^2 to Eq. (13), and subtract F_1 times (Eq. 14) and F_2 times (Eq. 13) from the result, which yields the following equation:

$$\nabla^2 \left[\nabla^2 \psi_1\right] - (F_1 + F_2) \left[\nabla^2 \psi_1\right]$$
$$= \nabla^2 q_1 - F_2 (q_1 - \beta y) - F_1 (q_2 - \beta y - R_s). \tag{15}$$

Equation (15) can be treated as a non-homogeneous Helmholtz equation with negative parameter $-(F_1 + F_2)$ and unknown $\nabla^2 \psi_1$. Once $\nabla^2 \psi_1$ is solved, the stream function for the top atmospheric layer is determined by a Poisson equation. The stream function for the bottom layer can be found by plugging the obtained value for ψ_1 into Eqs. (13) and (14) and by solving the equations for ψ_2. The potential vorticity functions q_i are evolved over time by a numerical advection procedure that models the conservation Eq. (12). For more details on the QG model, refer to Fisher et al. (2011).

3.2 Experiment setup and results

We study the model error parameter estimation problem with the QG model described above. In our experiments, we run the model with a 20×40 grid in each layer, and the dimension of the state vector is thus 1600. To generate data, we run the model with a 1 h time step with layer depths $D_1 = 6000$ and $D_2 = 4000$. Data is generated at every 6th step (the assimilation step is thus 6 h) by adding random noise with a given standard deviation σ_y to a given number of randomly chosen grid points. For the EKF estimation, bias is introduced to the forward model by using the wrong layer depths $\tilde{D}_1 = 5500$ and $\tilde{D}_2 = 4500$. To illustrate the model and the observations, a snapshot of a single step of the EKF estimation is given in Fig. 2.

We apply two different parameterizations for $\mathbf{Q}(\boldsymbol{\theta})$, a simple diagonal parameterization and a more complicated one that includes horizontal and vertical correlations. First, we simply study how important the model error term is in terms of EKF accuracy, with various observation settings. We then test the filter likelihood computation for the two selected $\mathbf{Q}(\boldsymbol{\theta})$ matrices. As a validation metric, we use the rms error of the state estimates and the model predictions. The likelihood values and the validation metrics are computed using separate "training data" and validation data.

The filter likelihood approach attempts to find a $\mathbf{Q}(\boldsymbol{\theta})$ so that the model predictions fit the observations with the correct accuracy (forecast error + measurement error), and we therefore expect this approach to yield reasonably good forecast error estimates as well, provided that the EKF assumptions are met. In Solonen and Järvinen (2013), a similar estimation technique was used to estimate the parameters of a small-scale ensemble prediction system (EPS), and there the approach produced a good representation of the forecast uncertainty. In order to verify the realism of the forecast error covariance matrix in this setup, we compare the squared mean variance of the forecast error covariance matrix against the true rms forecast error.

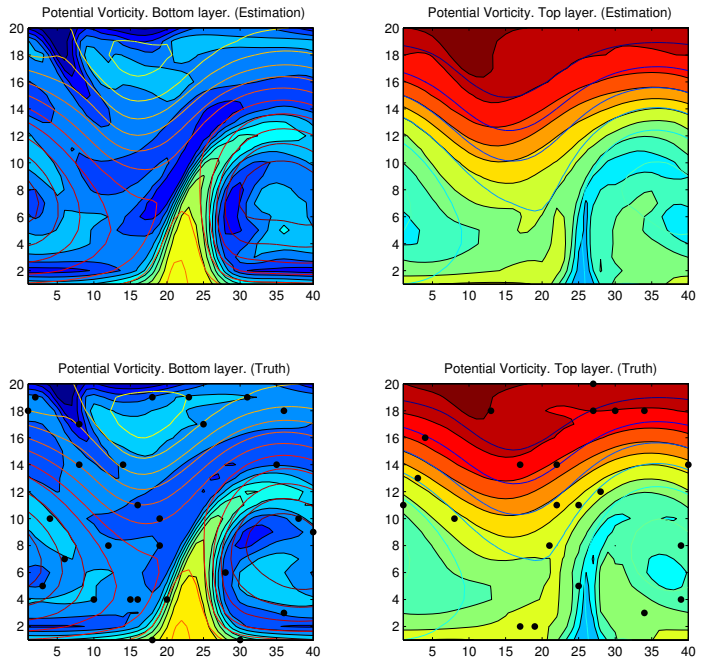

Figure 2. The true state (top-row panels) and the EKF estimate (bottom-row panels) for the bottom layer (left-column panels) and for the top layer (right-column panels) in a 20×40 grid for each layer using 50 randomly chosen observation locations (black dots). Filled contours describe the potential vorticity, and line contours describe the stream function.

3.2.1 The effect of observations to model error calibration

First, we study the relevance of the model error covariance matrix calibration by running EKF with various observation settings and various levels for the model error. We vary the number of observations used for the estimation at each assimilation step (20, 50 and 100), and the observation error standard deviation ($\sigma_y = 0.1$ and $\sigma_y = 1$). As the model error covariance matrix, we use the simplest possible parameterization, $\mathbf{Q} = \lambda \mathbf{I}$. For each observation setting, we run EKF with different values for the model error variance λ.

The results are shown in Fig. 3. One can see that with a large enough number of accurate enough observations (left panel, red curve), the model error calibration has very little effect on the accuracy of the filter; λ can vary many orders of magnitude without any significant difference in the rms errors of the state estimates. Reducing the number of observations (left panel, black and green curves) makes the calibration of the model error slightly more meaningful, and there seems to be an optimum value for λ that yields the most accurate EKF. Still, one can see that the benefit of optimizing λ is limited in this case; with large values for λ, EKF still converges, and the state estimates are almost as accurate as with the optimized λ.

When the observation error standard deviation is increased from $\sigma_y = 0.1$ to $\sigma_y = 1$ (right panel), the situation changes. Now the model error variance has a clearer impact on the accuracy of the filter, and substantial improvements in the filter can be achieved by correctly choosing the model error covariance parameters.

We conclude that the relevance of the model error calibration depends on the quality of the observations. If we have a large number of accurate observations available, the model error might not matter much. On the other hand, if the information of the observations is limited, the model error has to be tuned accurately to make the filter work properly. From the Bayesian perspective, this result is natural: if the observations do not identify the state properly, the prior has to be tuned carefully to make the estimation work.

3.2.2 Simple model error covariance matrix

To test the likelihood calculation, we first test the simple diagonal parameterization $\mathbf{Q} = \lambda \mathbf{I}$. We compute the likelihood from a training period of 100 assimilation steps, and compute the rms errors of the state estimates using a separate validation period of 100 steps. At each assimilation step, we use 100 observations with noise standard deviation $\sigma_y = 1$.

In Fig. 4, we plot the negative log-likelihood values of the training period to rms errors of the validation period. There is a clear correlation between the likelihood and the validation rms error, maximizing the likelihood results in optimal filter accuracy.

3.2.3 More complicated model error covariance matrix

Next, we perform the same experiment as above, but use a slightly more complicated covariance matrix parameterization. We use a Gaussian covariance function with three parameters, and for each layer, we define the covariance matrix elements as

$$\mathbf{Q}_{ij} = \begin{cases} \tau^2 + \sigma^2 \exp\left(-\frac{d(x_i, x_j)^2}{2\alpha^2}\right) & \text{when } i = j \\ \sigma^2 \exp\left(-\frac{d(x_i, x_j)^2}{2\alpha^2}\right) & \text{when } i \neq j, \end{cases} \tag{16}$$

where $d(x_i, x_j)$ denotes the distance between two grid points, $\sigma^2 > 0$ is the variance parameter, $\alpha > 0$ is the correlation length, and $\tau^2 > 0$ is a small positive nugget term that ensures that the matrix is positive definite. In addition, we estimate the vertical correlation $\rho \in [0, 1]$ between the two layers. We thus have four parameters altogether: $\boldsymbol{\theta} = (\tau^2, \sigma^2, \alpha, \rho)$.

We test randomly chosen parameter values uniformly in the intervals $\sigma^2 \in [0,\ 0.05]$, $\alpha \in [0,\ 10]$, $\rho \in [0,\ 1]$ and $\tau^2 \in [0,\ 0.01]$. For each parameter combination, we compute the likelihood values and rms errors for validation. The results are shown in Fig. 5 (left panel). Again, we see a clear correlation between the likelihood values and the rms errors: maximizing the likelihood results in an accurate EKF. To validate the results further, we compute the rms errors of forecasts launched from the EKF state estimates with different model error covariance matrix parameters. The results are shown in Fig. 5 (right panel). One can see that the parameters with a high likelihood also validate well in terms of forecast skill.

In a synthetic case, such as here, we know the truth behind the observations, and we can compute "samples" of the model error by running the truth model and the biased forecast model, starting from the same initial value and collecting the differences. This allows us to estimate the "true" model error covariance matrix. We estimated the covariance from 2000 samples of the error in two ways: first, we computed the full matrix directly with the empirical covariance estimation formula, and then estimated the parameters of the covariance function (Eq. 16) based on the samples. We plugged these matrices into EKF and computed the rms errors for the validation period. The results obtained in this way are shown in the left part of Fig. 5. Surprisingly, these matrices do not validate that well in terms of filter accuracy. We believe that the reason is that EKF is an approximative filter – it uses linearizations and assumes, for instance, that the model error and state are independent – and the imperfections in the method can to some extent be compensated for by calibrating the model error covariance matrix.

3.2.4 Verification of the forecast error covariance matrix

In order to verify the quality of the Kalman filter forecast error covariance matrix $\mathbf{C}_k^{\mathrm{p}} = \mathbf{M}_k \mathbf{C}_{k-1}^{\mathrm{est}} \mathbf{M}_k^T + \mathbf{Q}_k(\boldsymbol{\theta})$, the following two metrics are considered:

$$m_1(k) = \sqrt{\overline{\left(\boldsymbol{x}_k^{\mathrm{p}} - \boldsymbol{x}_k^{\mathrm{true}}\right)^2}}, \tag{17}$$

$$m_2(k) = \sqrt{\overline{\left(\sigma_k^{\mathrm{p}}\right)^2}}. \tag{18}$$

The first metric is the true rms forecast error, where the forecast $\boldsymbol{x}_k^{\mathrm{p}}$ is calculated from the previous Kalman filter estimate. The second metric is the squared mean variance of the forecast error covariance matrix. The mean in both cases is calculated over the 1600-dimensional state space.

In Fig. 6, we plot these two metrics using five different parameter combinations and the "true" model error covariance matrix (obtained via samples of the model error; see Sect. 3.2.3). For the parameter combinations, we selected the points that give the best and worst cost function values, and the points that correspond to the three quartile points of the cost function values (indicated by Q1, median and Q3 in Fig. 6).

From Fig. 6, we can observe that, using the best cost function point, the two metrics give a similar mean error level, showing that – on average – the Kalman filter forecast error (co)variance is realistic. This observation is also valid for the other points for which the cost function value is close to a minimum (grey lines in Fig. 6). For the other parameter combinations, we observe that the estimated and true forecast errors do not match: the forecast error is overestimated, and the difference grows gradually when going towards the worst parameter combination. The "true" model error covariance matrix, on the other hand, underestimates the forecast (co)variance, which is anticipated, as it does not take into account the imperfections of the EKF method discussed earlier.

The validation here was done using the more complicated model error parameterization (Eq. 16). We note that if metrics m_1 and m_2 are calculated using the simple diagonal covariance matrix $\mathbf{Q} = \lambda \mathbf{I}$, too low (high) λ values give on average too low (high) forecast error (co)variance, as expected. Near the optimum, the level of the forecast error covariance matrix is realistic (not shown).

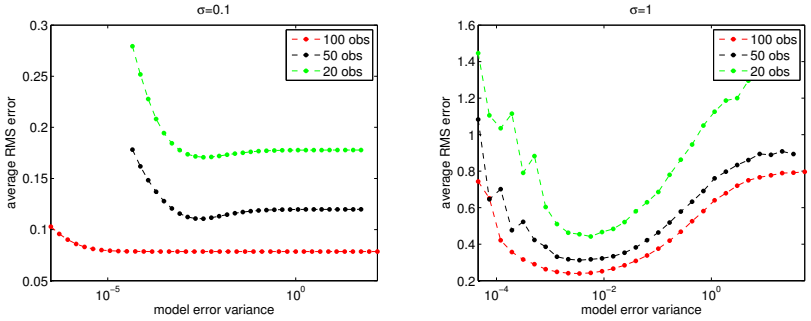

Figure 3. Model error variances λ vs. average rms errors of the state estimates with a varying number of observations and observation errors $\sigma_y = 0.1$ (left panel) and $\sigma_y = 1$ (right panel).

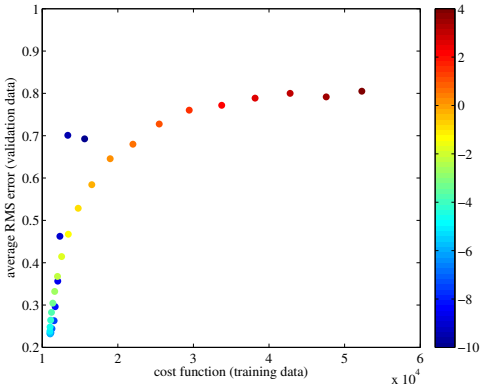

Figure 4. Negative log likelihood of the training data vs. the average rms error of the filter with the validation data. Colors indicate the logarithm of the model error variance λ.

4 Discussion and conclusions

In this paper, we consider the problem of calibrating the model error covariance matrix $\mathbf{Q}$ in extended Kalman filtering (EKF). The matrix $\mathbf{Q}$ is commonly seen as a tuning parameter in EKF, and is often postulated by the user in an ad hoc manner. We study a technique for objectively estimating the parameters $\boldsymbol{\theta}$ of a parametric version of the matrix, $\mathbf{Q}(\boldsymbol{\theta})$, based on indirect and noisy observations of the model state. The approach is based on approximating the likelihood of the parameters $\boldsymbol{\theta}$ using the EKF output. This "filter likelihood" method is tested with the two-layer quasi-geostrophic model that is often used as a benchmark case in numerical weather prediction studies.

One of our findings is that the relevance of the calibration of $\mathbf{Q}$ depends on the quality of the observations. The less information the observations contain about the model state, the more carefully the prior, a part of which $\mathbf{Q}$ is, needs to be tuned. On the other hand, if we have enough accurate observations, accurate optimization of $\mathbf{Q}$ might not be that beneficial. Secondly, we conclude that the filter likelihood approach works well in our test cases; maximizing the likelihood results in accurate EKF in terms of the rms errors of the state estimates and model predictions. In addition, the points that give a high likelihood validate well in terms of the quality of the forecast error estimates.

Our experiments in this paper are synthetic in the sense that we generate observations with the "true model" and introduce bias into the model that is used for estimation. In such a case, one can estimate the "true" model error by running predictions with the truth model and the biased forecast model, and collecting the differences between the predictions. Our experiments suggest that the model error obtained in this way is not optimal in terms of filter accuracy. A reason might be that the model error can be used to account for approximations and assumptions made in the filtering method. The consequence is that each filtering method should be tuned separately: the $\mathbf{Q}$ that works best in EKF might not be optimal for other filtering methods.

In this paper, the focus was on the extended Kalman filter. However, similar model error parameters appear in many other data assimilation methods as well, like, for instance, in the weak-constraint 4D-Var (Fisher et al., 2005) and ensemble Kalman filters (Evensen, 2007). In many so-called ensemble square root Kalman filters (Tippett et al., 2003), the model error is neglected, but covariance inflation techniques are used to account for the resulting underestimation of the uncertainty. We note that the parameters related to covariance inflation can also be tuned with the presented approach, as demonstrated in Hakkarainen et al. (2013). A problem with some ensemble methods is that they contain random perturbations, which can complicate the optimization process,

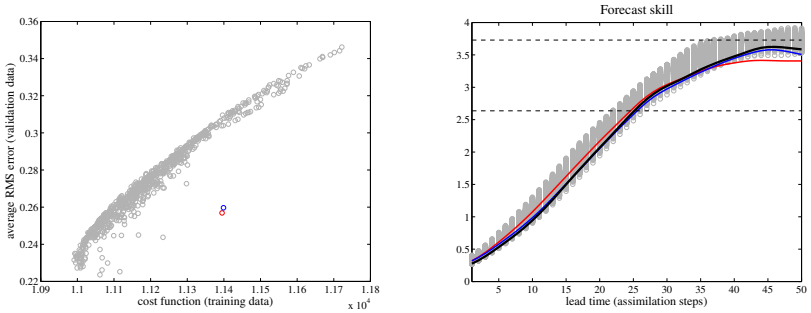

Figure 5. Left panel: negative log likelihood of the training data vs. the average rms error of the filter with the validation data. Right panel: the average forecast rms errors with different covariance matrix parameters. Red and blue colors indicate the results acquired with the full and parametrized "true" model error covariance matrices, respectively. The black line indicates the forecast skill acquired with the parameters that give the smallest negative log likelihood. Dashed vertical lines indicate climatological forecast skill and error saturation level.

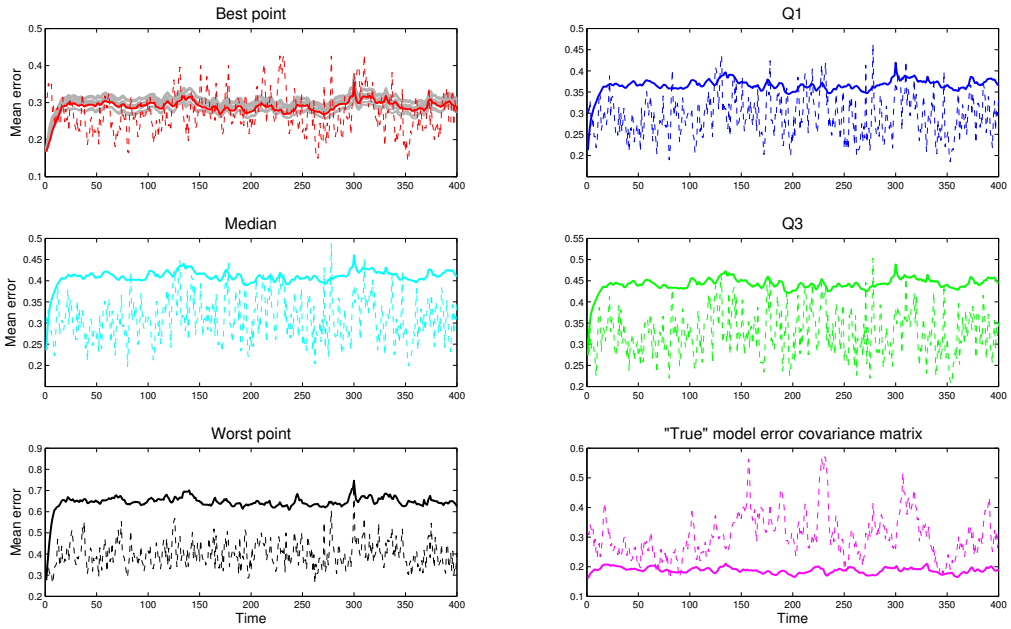

Figure 6. The true forecast error vs. the Kalman filter forecast error, calculated using different parameter combinations, and the "true" model error covariance matrix obtained from samples of model error. Dashed lines indicate the true rms forecast error and solid lines the squared mean variance of the forecast error covariance matrix. In the first subplot, the latter metric is also calculated from points that give the ten lowest cost function values (grey lines). The subplots Q1, median and Q3 give the results for parameter values that correspond to the three quartiles of the cost function values. Note the difference in scale in each subplot.

since the likelihood is stochastic, as noted in Hakkarainen et al. (2012) and Dowd (2011).

Here, we simply confirm that the likelihood approach works for calibrating the model error, and do not consider algorithms for maximizing or exploring the likelihood surface. A suitable method is case dependent. For simple models, standard optimization or, for instance, Markov chain Monte Carlo algorithms, are available. If the model is computationally expensive, one needs an efficient method to explore the surface with as few likelihood evaluations as possible. For instance, methods that apply empirical approximations (or "emulators") of the likelihood surface seem promising here; see, e.g., Rasmussen (2003). These topics are left for future research.

Acknowledgements. The research has been supported by the Academy of Finland (NOVAC and INQUIRE projects) and the Centre of Excellence in Inverse Problems Research. The authors would like to thank the editor and the two anonymous reviewers for their insightful comments that helped to improve the manuscript.

Edited by: Z. Toth
Reviewed by: two anonymous referees

References

Dee, D. P.: On-line estimation of error covariance parameters for atmospheric data assimilation, Mon. Weather Rev., 123, 1128–1145, doi:10.1175/1520-0493(1995)123<1128:OLEOEC>2.0.CO;2, 1995.

Dowd, M.: Estimating parameters for a stochastic dynamic marine ecological system, Environmetrics, 22, 501–515, doi:10.1002/env.1083, 2011.

Durbin, J. and Koopman, S. J.: Time series analysis by state space methods, Oxford University Press, New York, 2001.

Evensen, G.: Data assimilation: The ensemble Kalman filter, Springer, Berlin, 2007.

Fisher, M., Leutbecher, M., and Kelly, G.: On the equivalence between Kalman smoothing and weak-constraint four-dimensional variational data assimilation, Q. J. Roy. Meteorol. Soc., 131, 3235–3246, doi:10.1256/qj.04.142, 2005.

Fisher, M., Tremolet, Y., Auvinen, H., Tan, D., and Poli, P.: Weak-constraint and long window 4DVAR, Technical Report 655, European Centre for Medium-Range Weather Forecasts, Reading, UK, 2011.

Hakkarainen, J., Ilin, A., Solonen, A., Laine, M., Haario, H., Tamminen, J., Oja, E., and Järvinen, H.: On closure parameter estimation in chaotic systems, Nonlin. Processes Geophys., 19, 127–143, doi:10.5194/npg-19-127-2012, 2012.

Hakkarainen, J., Solonen, A., Ilin, A., Susiluoto, J., Laine, M., Haario, H., and Järvinen, H.: The dilemma on the uniqueness of weather and climate model closure parameters, Tellus A, 65, 20147, doi:10.3402/tellusa.v65i0.20147, 2013.

Kalman, R. E.: A New Approach to Linear Filtering and Prediction Problems, J. Basic Eng., 82, 35–45, 1960.

Mbalawata, I., Särkkä, S., and Haario, H.: Parameter estimation in stochastic differential equations with Markov chain Monte Carlo and non-linear Kalman filtering, Comput. Stat., 28, 1195–1223, doi:10.1007/s00180-012-0352-y, 2013.

Pedlosky, J.: Geophysical Fluid Dynamics, 2nd Edn., Springer-Verlag, New York, 1987.

Rasmussen, C. E.: Gaussian processes to speed up hybrid Monte Carlo for expensive Bayesian integrals, Bayesian Stat., 7, 651–659, 2003.

Särkkä, S.: Bayesian Filtering and Smoothing, Cambridge University Press, 2013.

Singer, H.: Parameter estimation of nonlinear stochastic differential equations: Simulated maximum likelihood versus extended Kalman filter and Itô–Taylor expansion, J. Comput. Graph. Stat., 11, 972–995, doi:10.1198/106186002808, 2002.

Solonen, A. and Järvinen, H.: An approach for tuning ensemble prediction systems, Tellus A, 65, 20594, doi:10.3402/tellusa.v65i0.20594, 2013.

Tippett, M., Anderson, J., Bishop, G., Hamill, T., and Whitaker, J.: Ensemble Square Root Filter, Mon. Weather Rev., 131, 1485–1490, doi:10.1175/1520-0493(2003)131<1485:ESRF>2.0.CO;2, 2003.

ACTA UNIVERSITATIS LAPPEENRANTAENSIS

706. LEVCHUK, IRINA. Titanium dioxide based nanomaterials for photocatalytic water treatment. 2016. Diss.

707. AMOUR, IDRISSA. Variational ensemble kalman filtering applied to data assimilation problems in computational fluid dynamics. 2016. Diss.

708. SHESTAKOVA, MARINA. Ultrasound-assisted electrochemical treatment of wastewaters containing organic pollutants by using novel Ti/Ta2O5-SnO2 electrodes. 2016. Diss.

709. OLEKSIIENKO, OLGA. Physico-chemical properties of sol-gel synthesized titanosilicates for the uptake of radionuclides from aqueous solutions. 2016. Diss.

710. PATALA, SAMULI. Advancing sustainability-oriented innovations in industrial markets. 2016. Diss.

711. KUORIKOSKI, TERO. Kohti resonoivaa urheilujohtamista – Tavoitteen muodostuminen urheilun kentässä. 2016. Diss.

712. LAHTELA, VILLE. Improving the properties of solid Scots pine (Pinus sylvestris) wood by using modification technology and agents. 2016. Diss.

713. NEVARANTA, NIKO. Online time and frequency domain identification of a resonating mechanical system in electric drives. 2016. Diss.

714. FANG, CHAO. Study on system design and key technologies of case closure welding for ITER correction coil. 2016. Diss.

715. GARCÍA PÉREZ, MANUEL. Modeling the effects of unsteady flow patterns on the fireside ash fouling in tube arrays of kraft and coal-fired boilers.

716. KATTAINEN, JARI. Heterarkkisen verkostoyhteistyön johtamistarpeet verkoston muotoutumisvaiheessa. 2016. Diss.

717. HASAN, MEHDI. Purification of aqueous electrolyte solutions by air-cooled natural freezing. 2016. Diss.

718. KNUTAS, ANTTI. Increasing beneficial interactions in a computer-supported collaborative environment. 2016. Diss.

719. OVASKA, SAMI-SEPPO. Oil and grease barrier properties of converted dispersion-coated paperboards. 2016. Diss.

720. MAROCHKIN, VLADISLAV. Novel solutions for improving solid-state photon detector performance and manufacturing. 2016. Diss.

721. SERMYAGINA, EKATERINA. Modelling of torrefaction and hydrothermal carbonization and heat integration of torrefaction with a CHP plant. 2016. Diss.

722. KOTISALO, KAISA. Assessment of process safety performance in Seveso establishments. 2016. Diss.

723. LAINE, IGOR. Institution-based view of entrepreneurial internationalization. 2016. Diss.

724. MONTECINOS, WERNER EDUARDO JARA. Axial flux permanent magnet machines – development of optimal design strategies. 2016. Diss.

725. MULTAHARJU, SIRPA. Managing sustainability-related risks in supply chains. 2016. Diss.

726. HANNONEN, JANNE. Application of an embedded control system for aging detection of power converter components. 2016. Diss.

727. PARKKILA, JANNE. Connecting video games as a solution for the growing video game markets. 2016. Diss.

728. RINKINEN, SATU. Clusters, innovation systems and ecosystems: Studies on innovation policy's concept evolution and approaches for regional renewal. 2016. Diss.

729. VANADZINA, EVGENIA. Capacity market in Russia: addressing the energy trilemma. 2016. Diss.

730. KUOKKANEN, ANNA. Understanding complex system change for a sustainable food system. 2016. Diss.

731. SAVOLAINEN, JYRKI. Analyzing the profitability of metal mining investments with system dynamic modeling and real option analysis. 2016. Diss.

732. LAMPINEN, MATTI. Development of hydrometallurgical reactor leaching for recovery of zinc and gold. 2016. Diss.

733. SUHOLA, TIMO. Asiakaslähtöisyys ja monialainen yhteistyö oppilashuollossa: oppilashuoltoprosessi systeemisenä palvelukokonaisuutena. 2017. Diss.

734. SPODNIAK, PETR. Long-term transmission rights in the Nordic electricity markets: An empirical appraisal of transmission risk management and hedging. 2017. Diss.

735. MONTONEN, JUHO. Integrated hub gear motor for heavy-duty off-road working machines – Interdisciplinary design. 2017. Diss.

736. ALMANASRAH, MOHAMMAD. Hot water extraction and membrane filtration processes in fractionation and recovery of value-added compounds from wood and plant residues. 2017. Diss.

737. TOIVANEN, JENNI. Systematic complaint data analysis in a supply chain network context to recognise the quality targets of welding production. 2017. Diss.

738. PATEL, GITESHKUMAR. Computational fluid dynamics analysis of steam condensation in nuclear power plant applications. 2017. Diss.

739. MATTHEWS, SAMI. Novel process development in post-forming of an extruded wood plastic composite sheet. 2017. Diss.

740. KÄHKÖNEN, TOMMI. Understanding and managing enterprise systems integration. 2017. Diss.

741. YLI-HUUMO, JESSE. The role of technical dept in software development. 2017. Diss.

742. LAYUS, PAVEL. Usability of the submerged arc welding (SAW) process for thick high strength steel plates for Arctic shipbuilding applications. 2017. Diss.

743. KHAN, RAKHSHANDA. The contribution of socially driven businesses and innovations to social sustainability. 2017. Diss.