Variational Kalman Filtering on Parallel Computers

Harri Auvinen, Tuomo Kauranne and Heikki Haario
Department of Mathematics
Lappeenranta University of Technology
tuomo.kauranne@lut.fi

31 October 2006

Outline

⇒ Benefits of Kalman Filtering (KF)

⇒ Feasible Approximations to KF

⇒ The Structure of Variational Kalman Filter (VKF) Computations

⇒ Properties of the Variational Kalman Filter

⇒ Serial and Parallel Complexity of VKF

⇒ Computational Results with the Lorenz'95 Model

⇒ Conclusions

Benefits of Kalman Filtering (KF)

The formulation of the general data assimilation (state estimation) problem for discrete time steps $t = 1, 2, 3, ..., n$ contains an evolution or prediction equation and an observation equation:

$$\mathbf{x}(t) = \mathcal{M}_t(\mathbf{x}(t-1)) + \mathbf{E}_t \quad (1)$$

$$\mathbf{y}(t) = \mathcal{K}_t(\mathbf{x}(t)) + \mathbf{e}_t, \quad (2)$$

where $\mathcal{M}_t$ is a model of the evolution operator, $\mathbf{x}_t$ is the state of the process at time t and $\mathbf{E}_t$ is a vector valued stochastic process.

The second equation connects the state estimate $\mathbf{x}(t)$ to the measurements $\mathbf{y}(t)$, with their associated observation errors $\mathbf{e}_t$, by way of an observation operator $\mathcal{K}_t$.

Benefits of Kalman Filtering (KF)

$\Rightarrow$ The optimal linear estimator to the data assimilation problem is the Extended Kalman Filter (EKF).

$\Rightarrow$ EKF can compensate for model bias

$\Rightarrow$ EKF analyses are discontinuous, but can be smoothed out afterwards

$\Rightarrow$ The analysis error covariance matrix can be used for assessing predictability

$\Rightarrow$ Its singular vectors can be used as initial perturbations to Ensemble Forecasting, if the time interval over which they have been computed is appropriate

Feasible Approximations to KF

$\Rightarrow$ EKF requires that in addition to evolving the model state with the nonlinear forecast model, all the columns of the analysis error covariance matrix be evolved back and forth in time, with the adjoint model and the tangent linear model in succession.

$\Rightarrow$ This is prohibitively expensive, and we must find feasible approximations to EKF.

Feasible Approximations to KF

$\Rightarrow$ 4D-Var is the special case of EKF where the model is perfect and the analysis error covariance matrix is static

$\Rightarrow$ In Reduced Rank Kalman Filtering (RRKF), the covariance matrix is constantly projected onto a fixed low dimensional subspace and evolved only there

$\Rightarrow$ In Ensemble Kalman Filtering (EnKF), a set of random perturbations is evolved to form a statistical sample of the span of the analysis error covariance matrix

Design Principles of the Variational Kalman Filter (VKF)

$\Rightarrow$ VKF - like EKF - behaves like a continuous data assimilation method

$\Rightarrow$ It is robust against model error

$\Rightarrow$ It produces a good estimate to the analysis error covariance matrix

$\Rightarrow$ VKF is highly parallel

Kalman Filter algorithm (1/2)

Let $\mathbf{x}_{est}(t-1)$ be an estimate of state $\mathbf{x}(t-1)$ and $\mathbf{S}_{est}(t-1)$ be the corresponding error covariance matrix of the estimate. At time t the evolution operator is used to produce an *a priori* estimate $\mathbf{x}_a(t)$ and its covariance $\mathbf{S}_a(t)$:

$$\mathbf{x}_a(t) = \mathbf{M}_t \mathbf{x}_{est}(t-1) \quad (3)$$

$$\mathbf{S}_a(t) = \mathbf{M}_t \mathbf{S}_{est}(t-1)\mathbf{M}_t^{\mathrm{T}} + \mathbf{SE}_t, \quad (4)$$

where $\mathbf{SE}_t$ is the covariance of the prediction error $\mathbf{E}_t$.

Kalman Filter algorithm (2/2)

The next step is to combine $\mathbf{x}_a(t)$ with the observations $\mathbf{y}(t)$ made at time t to construct an updated estimate of the state and its covariance:

$$\mathbf{G}_t = \mathbf{S}_a(t)\mathbf{K}_t^{\mathrm{T}}(\mathbf{K}_t\mathbf{S}_a(t)\mathbf{K}_t^{\mathrm{T}} + \mathbf{Se}_t)^{-1} \quad (5)$$

$$\mathbf{x}_{est}(t) = \mathbf{x}_a(t) + \mathbf{G}_t(\mathbf{y}(t) - \mathbf{K}_t\mathbf{x}_a(t)) \quad (6)$$

$$\mathbf{S}_{est}(t) = \mathbf{S}_a(t) - \mathbf{G}_t\mathbf{K}_t\mathbf{S}_a(t), \quad (7)$$

where $\mathbf{G}_t$ is the Kalman gain matrix, which is functionally identical to the maximum *a posteriori* estimator.

In a more general case, when the evolution model and/or the observation model is non-linear, the Extended Kalman Filter (EKF) is required.

Extended Kalman Filter algorithm (1/3)

The filter uses the full non-linear evolution model equation (1) to produce an *a priori* estimate: $\mathbf{x}_a(t) = \mathcal{M}_t(\mathbf{x}_{est}(t-1))$.

In order to obtain the corresponding covariance $\mathbf{S}_a(t)$ of the *a priori* information, the prediction model is linearized about $\mathbf{x}_{est}(t-1)$:

$$\mathbf{M}_t = \frac{\partial \mathcal{M}_t(\mathbf{x}_{est}(t-1))}{\partial \mathbf{x}} \quad (8)$$

$$\mathbf{S}_a(t) = \mathbf{M}_t \mathbf{S}_{est}(t-1)\mathbf{M}_t^{\mathrm{T}} + \mathbf{S}\mathbf{E}_t. \quad (9)$$

The linearization $\mathbf{M}_t$ of the model $\mathcal{M}_t$ in equation (8) is computed as the tangent linear model, from which the adjoint model $\mathbf{M}_t^{\mathrm{T}}$ is obtained as its transpose.

Extended Kalman Filter algorithm (2/3)

The observation operator is linearized at the time of the observation about the a $priori$ estimate $\mathbf{x}_a(t)$ in order to obtain $\mathbf{K}_t$, which is then used for calculating the gain matrix:

$$\mathbf{K}_t = \frac{\partial \mathcal{K}_t(\mathbf{x}_a(t))}{\partial \mathbf{x}} \tag{10}$$

$$\mathbf{G}_t = \mathbf{S}_a(t)\mathbf{K}_t^{\mathrm{T}}(\mathbf{K}_t\mathbf{S}_a(t)\mathbf{K}_t^{\mathrm{T}} + \mathbf{Se}_t)^{-1}. \tag{11}$$

The evolution of the full covariance matrix expressed by the term $\mathbf{M}_t\mathbf{S}_{est}(t-1)\mathbf{M}_t^{\mathrm{T}}$ in equation (9) is a computationally $very$ expensive operation for large models.

LAPPEENRANTA
UNIVERSITY OF TECHNOLOGY

Extended Kalman Filter algorithm (3/3)

After this, the full non-linear observation operator is used to update $\mathbf{x}_a(t)$ and this is then used to produce a current state estimate and the corresponding error estimate:

$$\mathbf{x}_{est}(t) = \mathbf{x}_a(t) + \mathbf{G}_t(\mathbf{y}(t) - \mathcal{K}_t(\mathbf{x}_a(t))). \quad (12)$$

$$\mathbf{S}_{est}(t) = \mathbf{S}_a(t) - \mathbf{G}_t\mathbf{K}_t\mathbf{S}_a(t). \quad (13)$$

If the linearization of the observation operator at $\mathbf{x}_a(t)$ is not good enough to construct $\mathbf{x}_{est}(t)$, it will be necessary to carry out some iterations of the last four equations.

Variational Kalman Filter algorithm (1/5)

The VKF method uses the full non-linear prediction equation (1) to construct an *a priori* estimate from the previous state estimate:

$$\mathbf{x}_a(t) = \mathcal{M}_t(\mathbf{x}_{est}(t-1)). \tag{14}$$

The corresponding approximated covariance $\mathbf{S}_a(t)$ of the *a priori* information is available from the previous time step of VKF method.

In order to avoid the computation of the Kalman gain we perform a 3D-Var optimization with a Kalman equivalent cost function. As the result of the optimization, we get the state estimate $\mathbf{x}_{est}(t)$ for the present time t.

Variational Kalman Filter algorithm (2/5)

The error estimate $\mathbf{S}_{est}(t)$ of the state estimate is obtained from the formula:

$$\mathbf{S}_{est}(t) = 2(Hess(t))^{-1}, \tag{15}$$

where the inverse of the matrix $Hess(t)$ can be approximated by using the search directions of the optimization process. These directions are used also by the Quasi-Newton method used in the minimization for creating an approximation to the Hessian of the cost function.

Variational Kalman Filter algorithm (3/5)

The estimate of the analysis error covariance is updated from the previous estimate using the Kalman formula

$$\mathbf{S}_a(t) = \mathbf{M}_{t-1}(\mathbf{S}_{est}(t-1) + \mathbf{SP}_{t-1})\mathbf{M}_{t-1}^{\mathrm{T}} + \mathbf{SE}_t, \qquad (16)$$

where $\mathbf{SP}_t$ is a stochastic process representing the *variance of the matrix valued truncation error* incurred in approximating the analysis error covariance matrix $\mathbf{S}_a(t)$ by the approximate inverse Hessian $(Hess(t))^{-1}$ from the minimization.

We split $\mathbf{S}_{est}(t-1)$ into a static background covariance $\mathbf{B}$ that represents the long term mean of the covariance matrix and to a low rank transient component, kept in vector form, which is transformed by the adjoint and tangent linear models.

Variational Kalman Filter algorithm (4/5)

In practice, the inverse of the Hessian only is kept, and in vector form to boot. The required multiplications with the analysis error covariance matrix are carried out with the Sherman-Morrison-Woodbury formula.

During the optimization another task is done: the full non-linear evolution model equation (1) is used to transfer search directions to the next time step $t + 1$. These evolved search directions are then used to update the approximation of the present covariance $\mathbf{S}_a(t)$ in order to approximate $\mathbf{S}_a(t + 1)$.

The other vectors used to approximate $\mathbf{S}_a(t)$ should be transferred by the nonlinear evolution as well, if there is a long interval between observations. In our current experiments, it has proven sufficient just to use them as they stand and still retain the useful qualities of VKF intact.

Variational Kalman Filter algorithm (5/5)

In order to maintain an upper limit on the rank of the Hessian approximation used in the L-BFGS Quasi-Newton method, the updates are carried out by using the limited memory Broyden-Fletcher-Goldfarb-Shannon (L-BFGS) update formulas.

After a 3D-Var optimization we have an updated covariance $\hat{\mathbf{S}}_a(t)$ which is then used to produce:

$$\mathbf{S}_a(t+1) = \hat{\mathbf{S}}_a(t) + \mathbf{SE}_t. \tag{17}$$

At each iteration of the optimization method the evolved search directions and the evolved gradients of J are stored and the approximation of the $\hat{\mathbf{S}}_a(t)$ is updated as the optimization method updates it's own approximation of the inverse Hessian $(\nabla\nabla J)^{-1}$.

The Structure of VKF Computations

Taken step-by-step, the VKF algorithm looks as follows

1. Start from the prior $\mathbf{x}_a(t-1)$ to the state at time $t-1$, and a prior guess for the analysis error covariance matrix $\mathbf{S}_{est}(t-1)$.
2. Solve the corresponding 3D-Var problem at time $t-1$ to produce the analysis $\mathbf{x}_{est}(t-1)$, using the current estimate of the error covariance matrix $\mathbf{S}_{est}(t-1)$ as the metric
3. Update $\mathbf{S}_{est}(t-1)$ with the search directions of the minimization, using the L-BFGS inverse Hessian update formula, so as to come up with the estimate $\tilde{\mathbf{S}}_{est}(t-1)$. The inverse Hessian is kept in vector form and the L-BFGS formula is applied at every matrix vector multiply.

The Structure of VKF Computations

4. Evolve the state estimate $\mathbf{x}_{est}(t-1)$ to the next observation time t with the nonlinear model $\mathcal{M}_{t-1}$, storing the tangent linear and adjoint models $\mathbf{M}_{t-1}$ and $\mathbf{M}_{t-1}^{\mathrm{T}}$, respectively,on the way
5. Carry out a local 4D-Var minimization starting at time $t-2$ with the state from the 3D-Var analyses as the observations to locally smoothen the analysis trajectory
6. Evolve some or all of the vectors forming the estimate $\tilde{\mathbf{S}}_{est}(t-1)$ to the observation time t, possibly dropping the oldest ones, with the nonlinear model $\mathcal{M}_{t-1}$

The Structure of VKF Computations

7. If the vectors spanning the range of the estimate to the analysis error covariance matrix above have lost too much of their orthogonality in the course of their evolution, reorthogonalize the basis using the Gram-Schmidt algorithm.
8. Transform this basis with the linear transformation $\mathbf{M}_{t-1}\tilde{\mathbf{S}}_{est}(t-1)\mathbf{M}_{t-1}^{\mathrm{T}}$ to produce a prior estimate to the error covariance matrix $\mathbf{S}_a(t)$ at time t. Drop vectors that do not shrink significantly in size in this reverse-time transformation (i.e. do not grow in forward time) and add the noise term $\mathbf{SE}_t$.
9. Loop from step 2 until end of assimilation window
10. As an outer loop, perform a 4D-Var assimilation over the entire assimilation window, using the sequence of analyzes $\mathbf{x}_{est}(t)$ as the observations
11. Iterate the whole sequence if needed

The Structure of VKF Computations

⇒ The tenth step results in a continuous analysis trajectory that has been processed by the VKF analogue to the Fixed Lag Kalman Smoother FLKS.

⇒ It comprises 4D-Var with direct observations of the state, as obtained from previous 3D-Var steps. We have called this method Variational Kalman Smoothing (VKS).

⇒ Smoothing by VKS does not change the analysis at the final time, but it does improve the accuracy of the analyzes during all intermediate steps, as has been verified in experiments.

⇒ Short local VKS sweeps, as described in the fifth step, have proven beneficial to forecast skill in experiments

⇒ VKS is particularly attractive for reanalysis.

The Serial Complexity of VKF Computations

⇒ Looking at the complexities and summing over all the steps, we see that the dominant serial cost in VKF consist of the advection of the covariance vectors back and forth, of the 3D-Var steps, the local 4D-Var steps, and of the 4D-Var minimization at the end.

⇒ Altogether, every vector of the inverse of the error covariance matrix stored requires three model or adjoint model integrations (in fact, once with each of the three models: the nonlinear one, the tangent linear one and the adjoint one).

⇒ If we were to keep the rank of the error covariance matrix at just the number of most recent search directions, the overall complexity of the VKF algorithm would amount just to two subsequent 4D-Var minimizations over the "one-and-a-half"assimilation window.

The Serial Complexity of VKF Computations

⇒ This is so, because the 3D-Var minimizations can be seen to be part of a single model integration with the 4D-Var algorithm in any case, and because 4D-Var requires as many model and adjoint model integrations as it takes steps to converge.

⇒ If it proves desirable to maintain a higher rank approximation to the error covariance matrix, the total serial complexity of VKF is multiplied by this rank, divided by a typical number of steps needed by 4D-Var to converge.

⇒ An educated guess would put the total serial complexity of VKF at two to five times the complexity of 4D-Var, and growing linearly with the resolution, just like the complexity of 4D-Var does.

The Parallel Complexity of VKF Computations

Let us now turn our sights on the parallel computational complexity of VKF and review parallelisation opportunities at each step

1. No computations yet
2. Standard 3D-Var at every observation time. No gain from parallelism.
3. Matrix update with a fixed number of vectors of the size of model resolution. Complexity is linear in spatial resolution and independent of the time step. No easy gains from parallelism.
4. A single model run to the next time step, with the standard tangent linear and adjoint coefficient stores on the way. No parallelism.
5. A short 4D-Var minimization with identity as observation operator.

6. A number of forward model runs with a fixed number (a few, possibly a few dozen) of independent initial states. All these are independent, and can be carried out in parallel.
7. Standard linear algebra with a fixed number of vectors of the size of model resolution. Complexity is linear in spatial resolution and independent of the time step. No parallelism.
8. A number of adjoint and subsequent tangent linear model runs, once back and forth for every vector, with a fixed number (a few, possibly a few dozen) of independent initial states, plus a sparse matrix vector product for each. Just as in step four, these can be fully parallelized, since we keep the approximate Hessian in vector form.
9. The steps above are taken over the entire assimilation window, instead of just between subsequent observation times
10. Standard 4D-Var over the entire assimilation window

The Parallel Complexity of VKF Computations

⇒ Many supercomputers in current operational weather forecasting centres, such as the IBM Cluster at ECMWF, or the 70 Tflops Cray Hood just ordered by CSC to Finland, or the 100 Teraflops Hood ordered by NERSC, have a clustered structure.

⇒ These parallel computers have dozens of Teraflops of computing power, and the operational weather models have been parallellized in a scalable fashion.

⇒ However, parallel supercomputers based on commodity processors have a significant bottleneck in their inter-cluster communications.

⇒ Because of Amdahls's and Hockney's laws, this limits the bisection bandwidth of the machines so badly that operational models are often run within a single cluster of processors only. (The Earth Simulator is a notable exception!)

The Parallel Complexity of VKF Computations

⇒ VKF has an ideal structure to fit itself neatly onto such a parallel architecture, with potentially close to linear speedup. This follows from the independence of the evolution steps of all the transient vectors used in approximating the analysis error covariance matrix.

⇒ Looking at the parallel complexities of the individual steps, and summing over all the steps, we get a surprising result: the parallel complexity of VKF is equivalent to just three model runs and local VKS sweeps, apart from the final 4D-Var smoothing step.

⇒ Moreover, the last 4D-Var iteration does not change the analysis at the final time step. This means that if we are to launch the next forecast from it we can postpone the 4D-Var to be carried out afterwards, outside the operational cycle, for archival and reanalysis purposes.

The Parallel Complexity of VKF Computations

⇒ We arrive therefore at a rather striking conclusion: the parallel complexity of VKF in the operational cycle is just three model runs, one with each model: the nonlinear one, the adjoint one and the tangent linear one, and a single, rapidly converging simplified 4D-Var.

⇒ Parallel complexity of VKF is also independent of the rank of the error covariance approximation, as long as this remains modest. If the covariance matrix is kept in vector form, all matrix vector products with it are fully parallelisable.

⇒ **VKF is therefore potentially faster than even the standard 4D-Var on a sufficiently powerful - yet realistic - parallel computer.**

Simulated assimilation results in Lorenz'95 case

We are very grateful to Mike Fisher and Martin Leutbecher of ECMWF for providing us with their codes for the Lorenz'95 model and the weak constraint 4D-Var and EKF data assimilation algorithms for it.

The assimilation results presented below are generated using the simple non-linear model introduced by Lorenz[1] in 1995. The model is small and represents simplified mid-latitude atmospheric dynamics of a single variable.

[1] E. N. Lorenz: Predictability: A problem partly solved. *Proc. seminar on Predictability, Vol. 1, ECMWF, Reading, Berkshire, UK*, 1-18, (1995).

[2] E. N. Lorenz and K. A. Emanuel: Optimal sites for supplementary weather observations: Simulations with a small model. *J. Atmos. Sci.*, **55**, 399-414, (1998).

Lorenz'95 model and parameters

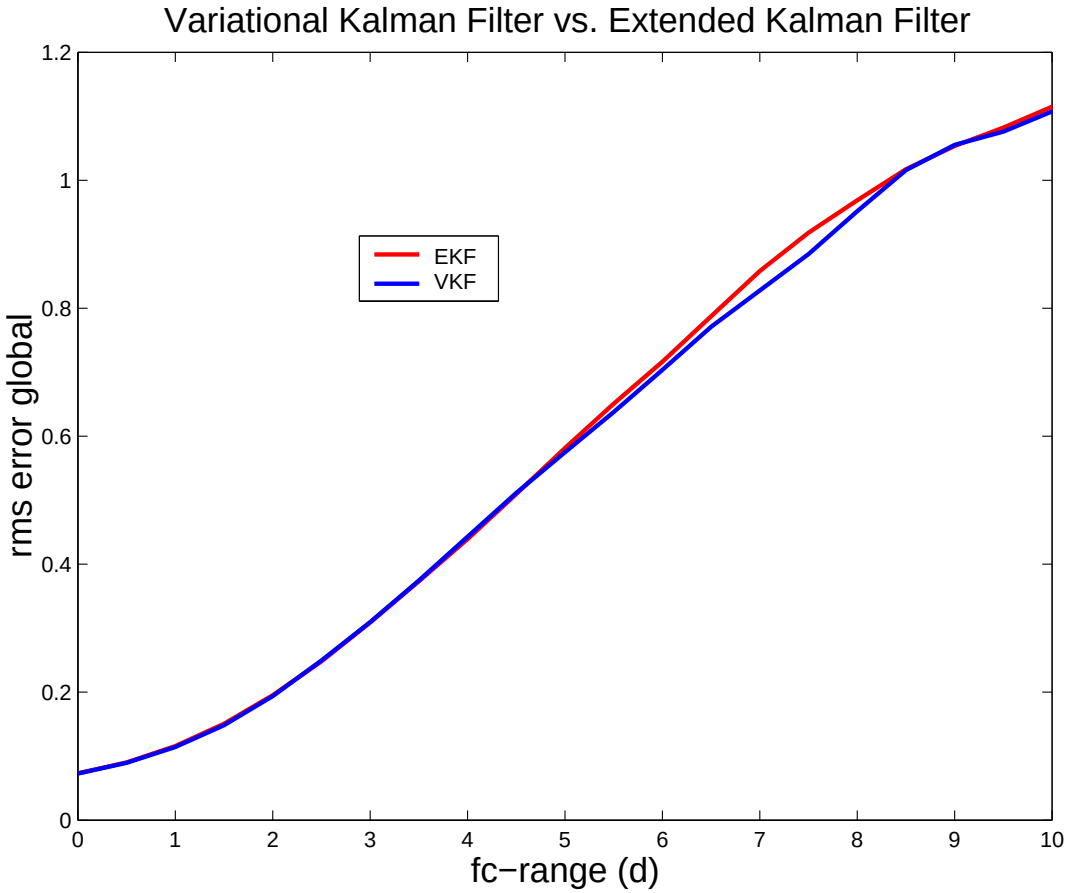
The model consist of a set of coupled ordinary differential equations

$$\frac{\partial c_i}{\partial t} = c_{i-2}c_{i-1} + c_{i-1}c_{i+1} - c_i + F, \tag{18}$$

where $i = 1, 2, ..., n$ and F is a constant. The number of grid points is controlled by the number n. The domain is set to be cyclic, so that $c_{-1} = c_{n-1}, c_0 = c_n$ and $c_{n+1} = c_1$.

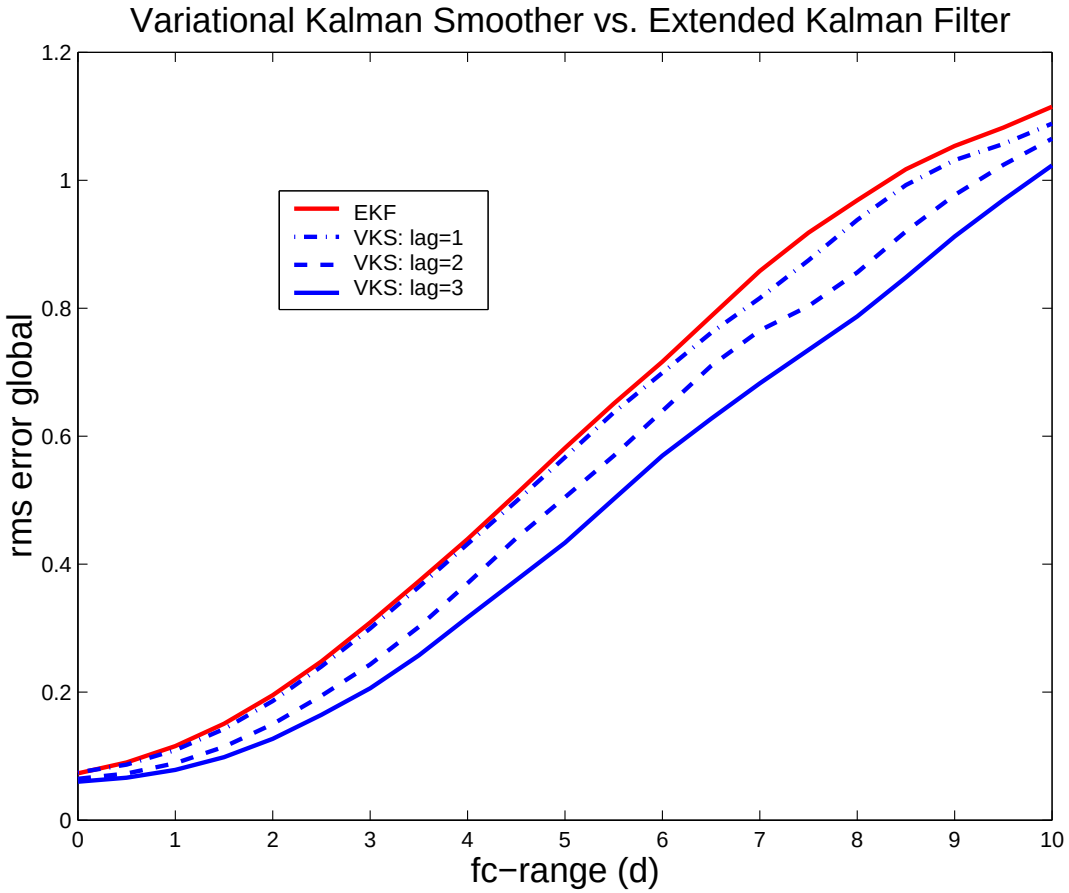
The simulations presented in this section follow Lorenz and Emmanuel[2], so we select $F = 8$ and take a unit time interval to represent 5 days. The number of the grid points was set to $n = 40$. The time integration of the model was performed using a fourth order Runge-Kutta method.

Simulated assimilation results without model bias



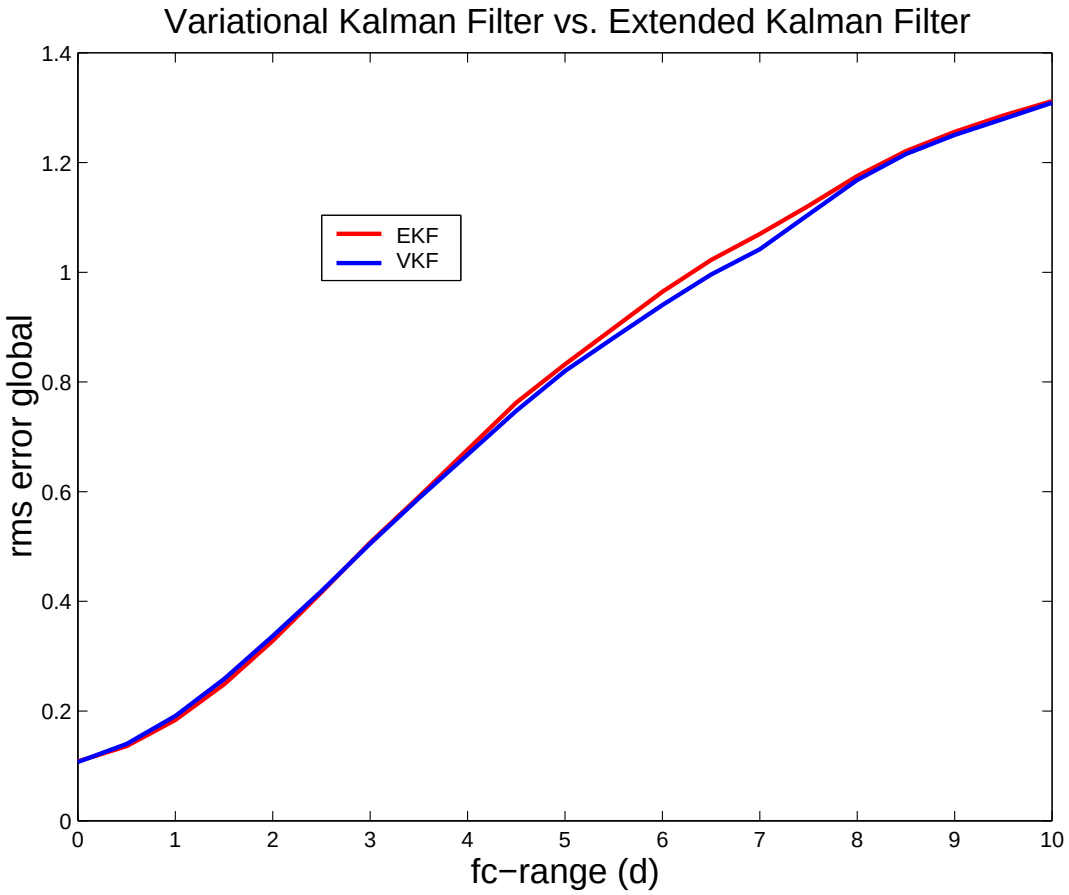
LAPPEENRANTA
UNIVERSITY OF TECHNOLOGY

Simulated assimilation results without model bias



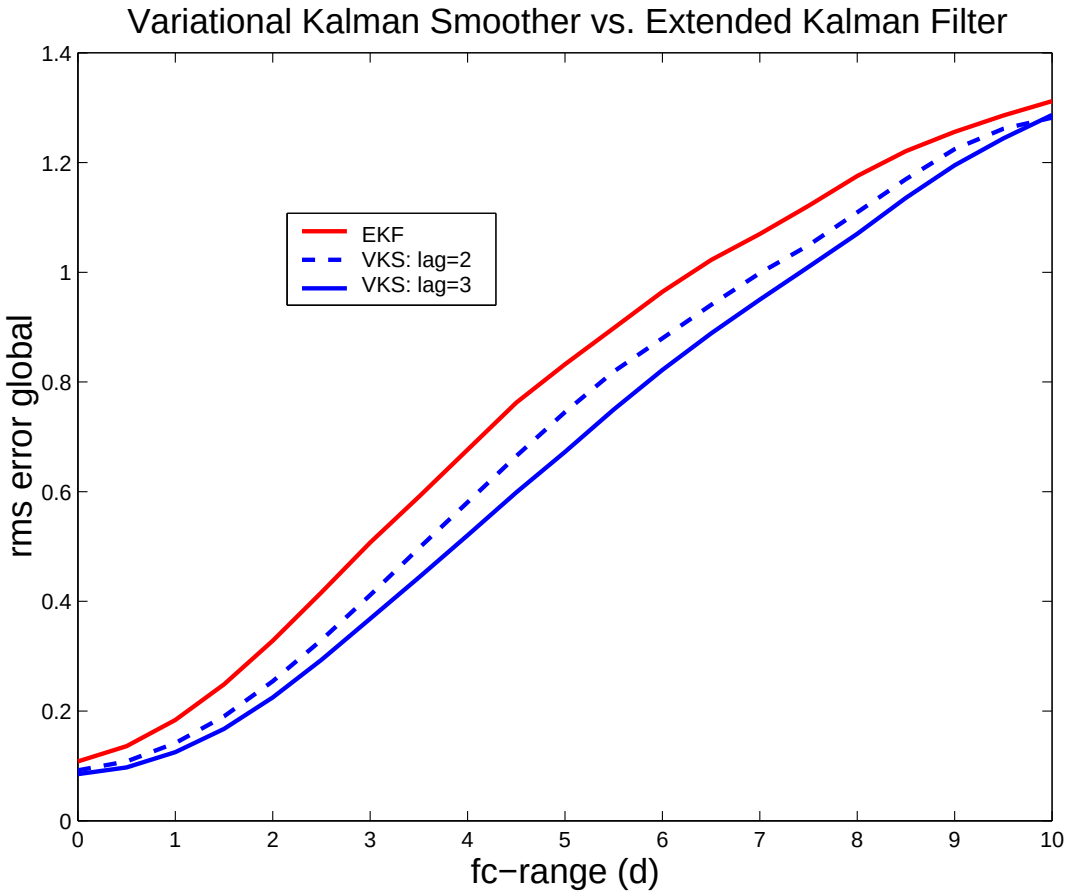
LAPPEENRANTA
UNIVERSITY OF TECHNOLOGY

Simulated assimilation results with model bias (1/2)



LAPPEENRANTA
UNIVERSITY OF TECHNOLOGY

Simulated assimilation results with model bias (2/2)



LAPPEENRANTA
UNIVERSITY OF TECHNOLOGY

Conclusions

⇒ The Variational Kalman Filter (VKF) method is as good a data assimilation method as EKF in the Lorenz'95 benchmark

⇒ It is robust against model error

⇒ It outperforms EKF, both in accuracy and in computational complexity, in retrospective analysis

⇒ It has a serial complexity comparable to that of 4D-Var

⇒ It has a parallel complexity comparable to that of three subsequent model runs with a parallelized model plus a sequence of local 4D-Var steps, and has the potential to outperform even the standard 4D-Var in wall-clock time on a large parallel supercomputer

⇒ In principle, linear speedup with such a parallel complexity looks attainable on current parallel supercomputers.