A stable and accurate Variational Kalman Filter

Alexander Bibov, Heikki Haario, Harri Auvinen, John Bardsley, Tuomo Kauranne

Lappeenranta University of Technology / University of Montana

ECMWF Workshop on High-performance Computing in Meteorology
October 2, 2012
Overview

- Desirable properties of assimilation methods
- Variational methods and Ensemble Kalman filters
- Deterministic Kalman filters:
  - Classical Extended Kalman filter
  - Variational reformulation of the Kalman filter
- Quasi-Geostrophic model: integration and implementation
- Parallelization concerns
- Conclusions
Desirable properties of assimilation methods

- Assimilation methods should be accurate
  - No bias
- Precise
  - Use all available information in an optimal fashion
  - Provide for dynamic error covariances
- Parallelizable
- Simple
  - Tangent linear and adjoint models difficult to maintain
- All these criteria are difficult to meet simultaneously
Variational methods and Ensemble Kalman Filters

- Optimum Interpolation
  - Unbiased
  - Parallelizable by domain decomposition
  - Not precise – static error covariances
  - No tangent linear or adjoint model

- 4DVAR
  - Precise, but static error covariances
  - Potentially biased – because of strong model constraint
  - Not very parallelizable
  - Tangent linear and adjoint models
Variational methods and Ensemble Kalman Filters

- Weak constraint 4DVAR
  - Precise, partially dynamic error covariance
  - Potentially biased
  - Computationally expensive – big control vector dimension
  - Parallelizable – by domain decomposition in time?
  - Tangent linear and adjoint models

- Ensemble Kalman Filters
  - Potentially unbiased
  - Efficiently parallelizable
  - Dynamic error covariance
  - Not precise – ensemble small compared to state space dimension
  - No tangent linear or adjoint models

6/2010
Extended and Variational Kalman Filters
Kalman Filters: Extended Kalman Filter

**Input:** $x_k, y_{k+1}, C_k, Q_{k+1}, R_{k+1}, M_{k+1}, K_{k+1}$.

1. $x_{k+1}^p := M_{k+1} x_k$

2. $C_{k+1}^p := M_{k+1} C_k M_{k+1}' + Q_{k+1}$

3. $G_{k+1} := C_{k+1}^p K_{k+1}' (K_{k+1} C_{k+1}^p K_{k+1}' + R_{k+1})^{-1}$

4. $x_{k+1} := x_{k+1}^p + G_{k+1} (y_{k+1} - K_{k+1} x_{k+1}^p)$

5. $C_{k+1} := C_{k+1}^p - G_{k+1} K_{k+1} C_{k+1}^p$

**Output:** $x_{k+1}, C_{k+1}$

**Where:** $C_k, Q_{k+1}, R_{k+1}$ are covariance matrices of $x_k, \varepsilon^p_{k+1}, \varepsilon^o_{k+1}$ respectively.
Extended Kalman Filter: drawbacks

- Covariance error matrix propagation requires $O(n^3)$ flops
- Covariance storage requires to store $n^2$ floating-point or double-precision values
- In the case of weather simulation dynamical systems $n \approx 10^{17}$, which makes the basic formulations impossible to implement

Solution: provide a low-memory matrix approximation supporting efficient matrix-vector multiplications
Variational Kalman Filter VKF

- Variational Kalman Filter
  - Precise – equivalent to EKF, hence dynamic error covariance
  - Guaranteed to be stable
  - Bias can be kept under control
  - Not very parallelizable
  - Tangent linear and adjoint models inherited from 4DVAR
Low-memory matrix approximations

- Consider an arbitrary matrix $A$
- The task is to compute its “smallest” update $D$ in terms of Frobenius norm such that $(A + D)v = y$, where $v$ and $y$ is known pair of vectors and $v$ is nonzero.

$$Dv = y - Av = r, \|D\|_{F}^{2} \rightarrow \text{min}$$

- Consider a pair of vectors $v$ and $y$.
- The task is to find a symmetric positive definite matrix which maps $v$ to $y$. 
Theorem. Let $L_C$ be a nonsingular matrix, $H_C = L_C L_C^T$. Let $y$ and $v$ be an arbitrary pair of vectors where $v$ is nonzero. There is a symmetric positive definite matrix $H_+$, such that $(H_C + H_+)v = y$, if and only if $y^T v > 0$. If there is such a matrix, then $H_+ = J_+ J_+^T$, where

$$J_+ = L_C + \left( y - \sqrt{\frac{y^T v}{v^T H_C v}} H_C v \right) (L_C^T v)^T \frac{1}{\sqrt{\frac{y^T v}{v^T H_C v} v^T H_C v}}$$
Low-memory matrix approximations: BFGS update

\{(s_k, y_k), (s_{k-1}, y_{k-1}), \ldots, (s_{k-m+1}, y_{k-m+1})\}

\[ x_{k+1} = x_k - \alpha \tilde{H}_k \nabla f_k \]

- \(f(x)\) – cost function, \(f_k = f(x_k)\).
- \(\tilde{H}_k\) - inverse Hessian approximation.
- \(\alpha\) – step scale.
- \(s_k = x_{k+1} - x_k, y_k = \nabla f_{k+1} - \nabla f_k\).
Variational Kalman Filter

**Input:** $x_k, y_{k+1}, C_k, Q_{k+1}, (R_{k+1})^{-1}, M_{k+1}, K_{k+1}$.

1. $x_{k+1}^p := M_{k+1}(x_k)$

2. Compute L-BFGS approximation $B_{k+1}^*$ of $(C_{k+1}^p)^{-1}$, where $C_{k+1}^p := M_{k+1}C_kM_{k+1}' + Q_{k+1}$.

3. Minimize with L-BFGS

$$l(x) = (y_{k+1} - K_{k+1}x)'(R_{k+1})^{-1}(y_{k+1} - K_{k+1}x) + (x - x_{k+1}^p)'B_{k+1}^*(x - x_{k+1}^p)$$

4. Define $x_{k+1}$ to be the minimizer from step 3 and $C_{k+1}$ to be the L-BFGS approximation of inverse Hessian of the problem on step 2.
Use of LBFGS in stabilized VKF

Assume that an approximation $B_{k-1}^\#$ for covariance $C_{k-1}^{est}$ is available. Then the EKF formulas can be approximated directly, which leads to the following algorithm:

1. Compute prediction $x_k^p = \mathcal{M}_k(x_k^{est})$.
   Define prediction covariance $C_k^p = M_kB_{k-1}^\#M_k^T + C_{\mathcal{E}}^p$.
   Define $A = K_kC_k^pK_k^T + C_{\mathcal{E}}^p$, $b = y_k - K_kx_k^p$.

2. Solve optimization problem $\frac{1}{2}x^TAx - b^Tx \to \min$ with respect to $x$ and compute a low-memory approximation $B^* \approx A^{-1}$.

3. Compute state estimate $x_k^{est} = x_k^p + C_k^pK_k^Tx^*$, where $x^*$ is solution for the optimization problem from step 2.

4. Compute a low-memory approximation $B_k^\#$ for the estimate covariance $C_k^{est} = C_k^p - C_k^pK_k^TB^*K_kC_k^p$.

Matrix $C_k^{est} = C_k^p - C_k^pK_k^TB^*K_kC_k^p$ is not guaranteed to remain positive definite. Therefore, numerical instability may occur in some cases.
Stabilized VKF

Setting $B^* = A^{-1}$ we get

$$C_{k}^{est} = C_k^p - C_k^p K_k^T (2I - B^* A) B^* K_k C_k^p = C_k^p - C_k^p K_k^T A^{-1} K_k C_k^p = C_k^p - G_k K_k C_k^p,$$

which is the exact formula from the EKF. Therefore, the Stabilized VKF still mimics the basic EKF formulas.
Quasigeostrophic 2-layer model
Quasi-Geostrophic model: review

Top layer

Bottom layer

Land

\( U_1 \) - constant zonal flow in the top layer. \( S(x, y) \) - orography term.
\( U_2 \) - constant zonal flow in the bottom layer. \( f_0 \) - Coriolis parameter.
\( D_1 \) - undisturbed depth of the top layer.
\( D_2 \) - undisturbed depth of the bottom layer.
Quasi-Geostrophic model: review

\[ \dot{g} = g \frac{\Delta \theta}{\bar{\theta}}, F_1 = \frac{f_0^2 L^2}{\dot{g} D_1}, F_2 = \frac{f_0^2 L^2}{\dot{g} D_2}, \]

\[ R_s = \frac{f_0 L S(x, y)}{U D_2}, \beta = \beta_0 \frac{L}{U}. \]

\[ \frac{D_1 q_1}{Dt} = \frac{D_2 q_2}{Dt} = 0; \]

\[ q_1 = \nabla^2 \psi_1 - F_1 (\psi_1 - \psi_2) + \beta y, \]

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

\[ \frac{D_i}{Dt} = \frac{\partial}{\partial t} + u_i \frac{\partial}{\partial x} + v_i \frac{\partial}{\partial y}, \]

\[ \nabla \psi_i = (v_i, -u_i) \]
Quasi-Geostrophic model: review

\[
\begin{align*}
\frac{D_1 q_1}{Dt} &= \frac{D_2 q_2}{Dt} = 0, \\
q_1 &= \nabla^2 \psi_1 - F_1 (\psi_1 - \psi_2) + \beta y, \\
q_2 &= \nabla^2 \psi_2 - F_2 (\psi_2 - \psi_1) + \beta y + R_s, \\
D_i \cdot \frac{D}{Dt} &= \frac{\partial}{\partial t} + u_i \frac{\partial}{\partial x} + v_i \frac{\partial}{\partial y}, \\
\nabla \psi_i &= (v_i, -u_i)
\end{align*}
\]

Apply \(\nabla^2\) to the equation for \(q_1\), subtract \(F_1\) times equation for \(q_1\) and \(F_2\) times equation for \(q_2\):

\[
\begin{align*}
\nabla^2 (\nabla^2 \psi_1) - (F_1 + F_2) (\nabla^2 \psi_1) &= \nabla^2 (q_1 - \beta y) - F_2 (q_1 - \beta y) - F_1 (q_2 - \beta y - R_s)
\end{align*}
\]
Quasi-Geostrophic model: integration pipeline

Wind operator: \( \nabla \psi_i = (v_i, -u_i) \)
Experimental Design: simulation model

- Two-layer Quasi-Geostrophic model solved on a cylindrical 40x20 domain
- Spatial discretization steps $\Delta x = \Delta y = 300 km$
- Time discretization step $\Delta t = 21600 s$
- Layer depths $D_1 = 6000 m$, $D_2 = 4000 m$
- Orography term:
  - Gaussian hill
    - 2000m high, 1000km wide at grid vertex (0, 15)
- Domain 12000km x 6000km
Experimental Design: orography component
Experimental Design: assimilation with a biased model

- Assimilation model, and tangent linear and adjoint models:
  - The same settings as for simulation model
  - Different layer depths $D_1 = 5500m$, $D_2 = 4500m$

- Initial state:
  - Propagate assimilation and “truth” models for two weeks with one hour time step

- Observation concept:
  - Observe sparse set of 100 grid vertices at every assimilation step
  - Selection of the vertices observed at every assimilation step remains unchanged
Experimental Design: model error

- Main diagonal hill corresponds to in-layer correlations between the vertices.
- Off diagonal hills correspond to cross-layer correlations.
- Small hills near the corners reveal model’s periodical nature.
Assimilation results
Parallelization concerns with VKF
Parallelization concerns with VKF

- With respect to parallelization, VKF is similar to 4DVAR
- This means it is an inherently serial algorithm
- Both
  - L-BFGS itself,
  - the alternating serial calls to the tangent linear and adjoint models, and
  - the alternation between 3DVAR-like purely spatial observation processing and 4DVAR-like error covariance update process
- are all serial
Parallelization concerns with VKF

- On the other hand, the serial complexity of VKF is almost identical to that of 4DVAR, and it may be even less: it consists of the same operations as 4DVAR, organized in a different manner.
- So instead of a variational form of EKF, VKF can also be seen as an efficient way to provide 4DVAR with:
  - A dynamic error covariance matrix
  - A way to counter model bias without covariance inflation
- But VKF can be run - just like 4DVAR - in an Ensemble of Data Assimilations EDA
- This will yield as ensemble from the right posterior distribution
Conclusions
Conclusions

- Assimilation methods should be
  - Accurate
  - Precise
  - Parallelizable
  - Simple

- Stabilized Variational Kalman Filter is
  - Accurate
  - Precise
  - Not very parallelizable – but serves well in EDA
  - Simple, if 4DVAR has been in use before
Conclusions

- VKF has been implemented in the Lappeenranta version of ECMWF OOPS, dubbed LOOPS
- Integration with IFS is possible once it is brought into OOPS – see the talk by Yannick and Mike in Session 11 tomorrow 😊
Thank You!