École Doctorale des Sciences de l'Environnement d'Île-de-France
Année 2009-2010

Modélisation Numérique de l'Écoulement Atmosphérique et Assimilation d'Observations

Olivier Talagrand

Cours 7

4 Juin 2010

Two solutions :

- Low-rank filters (Heemink, Pham, …)

  Reduced Rank Square Root Filters, Singular Evolutive Extended Kalman Filter, ….

- Ensemble filters (Evensen, Anderson, …)

  Uncertainty is represented, not by a covariance matrix, but by an ensemble of point estimates in state space which are meant to sample the conditional probability distribution for the state of the system (dimension $N \approx O(10\text{-}100)$).

  Ensemble is evolved in time through the full model, which eliminates any need for linear hypothesis as to the temporal evolution.

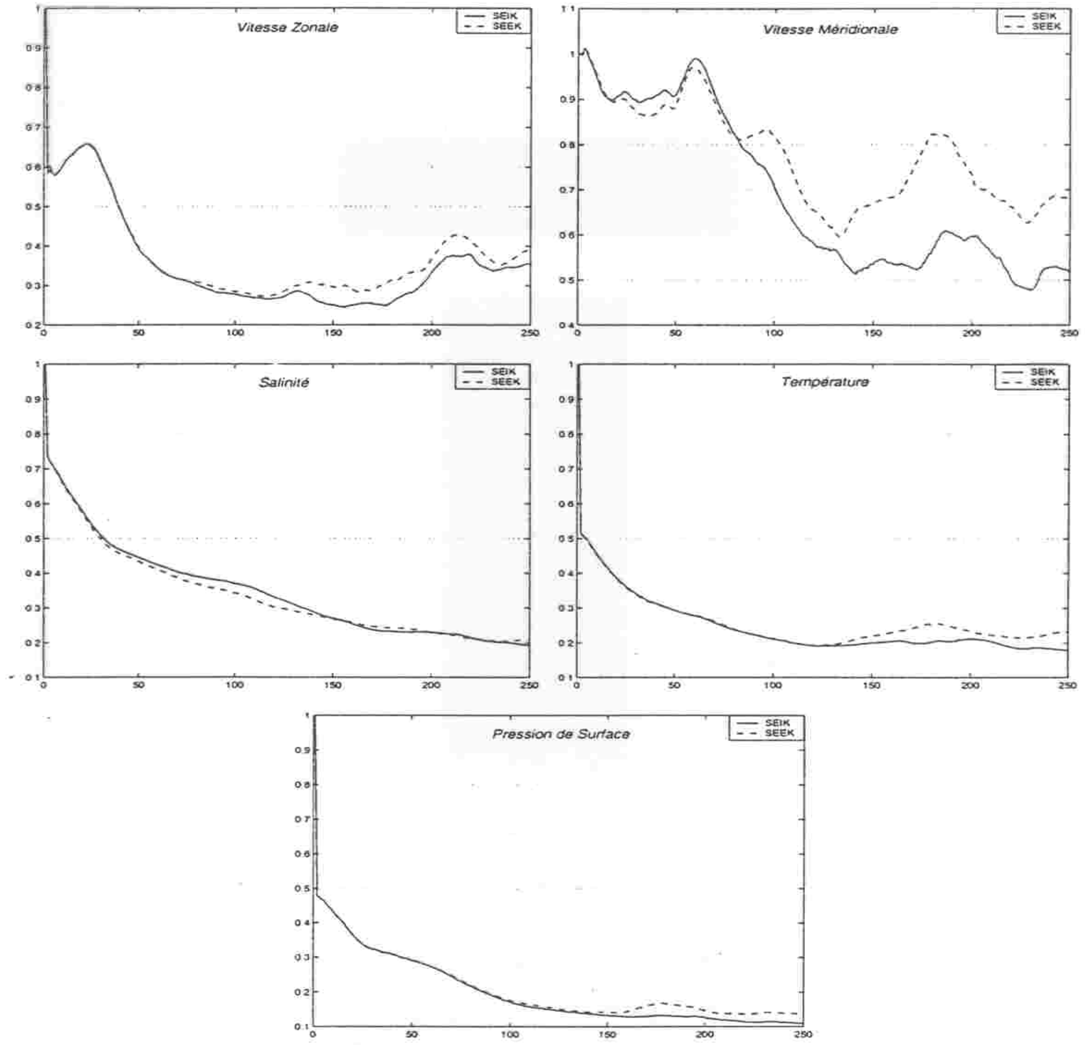


FIG. 5.2 – *Evolution dans le temps de la RRMS des filtres SEIK et SEEK*

I. Hoteit, Doctoral Dissertation, Université Joseph Fourier, Grenoble, 2001

How to update predicted ensemble with new observations ?

Predicted ensemble at time t : $\{x^b{}_n\}$, $\quad n = 1, \ldots, N$
Observation vector at same time : $y = Hx + \varepsilon$

- Gaussian approach

  Produce sample of probability distribution for real observed quantity Hx
  $y_n = y - \varepsilon_n$
  where ε_n is distributed according to probability distribution for observation error ε
  .

  Then use Kalman formula to produce sample of 'analysed' states

  $$x^a{}_n = x^b{}_n + P^b H^T [HP^bH^T + R]^{-1} (y_n - Hx^b{}_n) , \qquad n = 1, \ldots, N \tag{2}$$

  where P^b is covariance matrix of predicted ensemble $\{x^b{}_n\}$.

  Remark. If P^b was exact covariance matrix of background error, (2) would achieve Bayesian estimation, in the sense that $\{x^a{}_n\}$ would be a sample of conditional probability distribution for x, given all data up to time t.

Called ***Ensemble Kalman Filter*** (*EnKF*)

But problems

- Collapse of ensemble for small ensemble size (less than a few hundred). Empirical 'covariance inflation'

- Spurious correlations appear at large geographical distances. Empirical 'localization'.

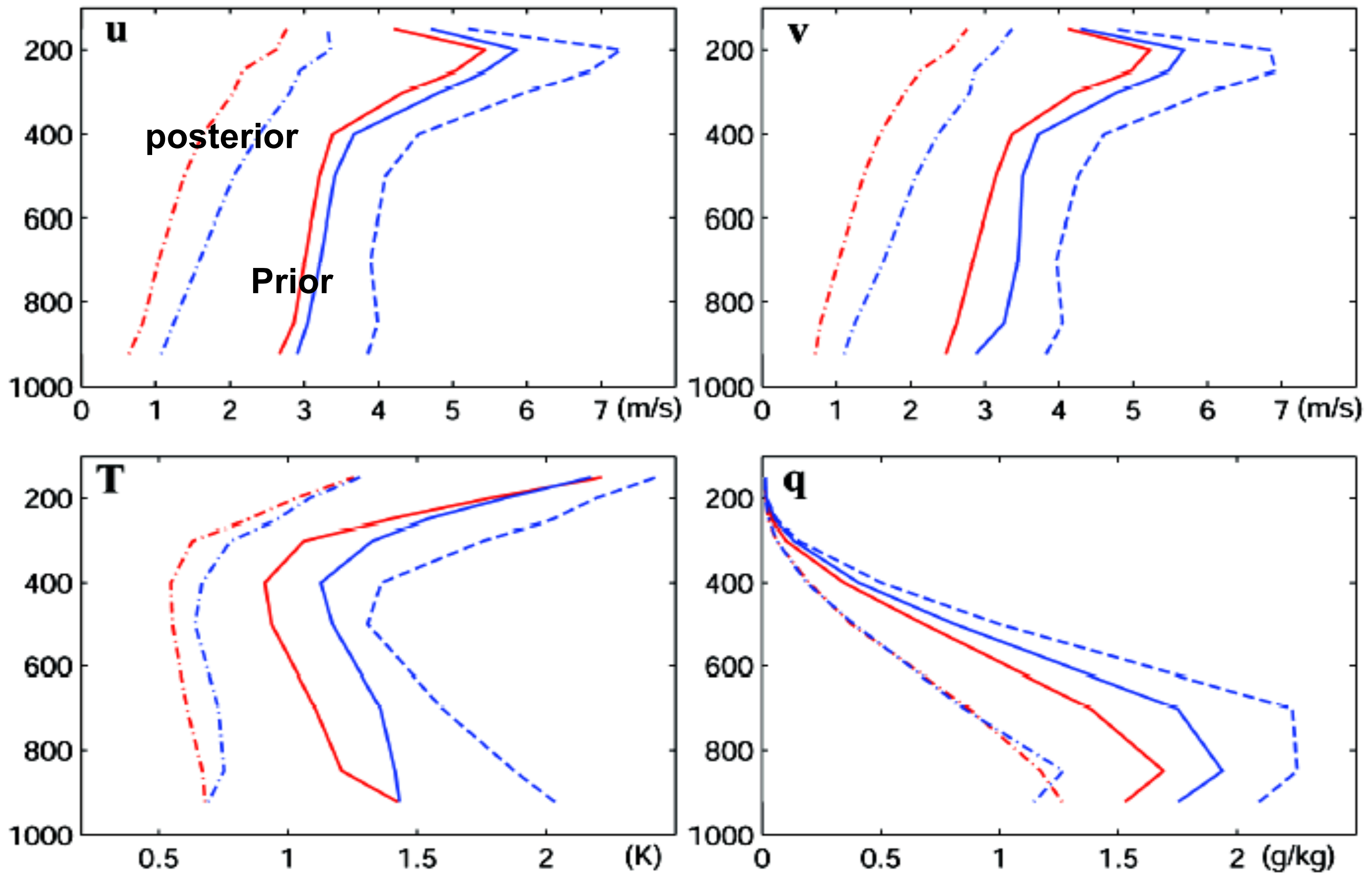
In formula

$$x^a{}_n = x^b{}_n + P^b H^T [HP^b H^T + R]^{-1} (y_n - Hx^b{}_n) \,, \qquad n = 1, \ldots, N$$

P^b, which is covariance matrix of an N-size ensemble, has rank N-1 at most. This means that corrections made on ensemble elements are contained in a subspace with dimension N-1. Obviously very restrictive if $N \ll p$, $N \ll n$.

Month-long Performance of EnKF vs. 3Dvar with WRF

— EnKF — 3DVar (prior, solid; posterior, dotted)



Better performance of EnKF than 3DVar also seen in both 12-h forecast and posterior analysis in terms of root-mean square difference averaged over the entire month

(Meng and Zhang 2007c, MWR, in review)

Situation still not entirely clear.

Houtekamer and Mitchell (1998) use two ensembles, the elements of each of which are updated with covariance matrix of other ensemble.

Local Ensemble Transform Kalman Filter (*LETKF*) defined by Kalnay and colleagues. Correction is performed locally in space on the basis of neighbouring observations.

In any case, optimality always requires errors to be independent in time. That constraint can be relaxed to some extent by augmenting the state vector in the temporal dimension.

Variational approach can easily be extended to time dimension.

Suppose for instance available data consist of

- Background estimate at time 0

  $x_0^b = x_0 + \zeta_0^b$ $\qquad E(\zeta_0^b \zeta_0^{bT}) = P_0^b$

- Observations at times $k = 0, \dots, K$

  $y_k = H_k x_k + \varepsilon_k$ $\qquad E(\varepsilon_k \varepsilon_j^T) = R_k$

- Model (supposed for the time being to be exact)

  $x_{k+1} = M_k x_k$ $\qquad k = 0, \dots, K\text{-}1$

Errors assumed to be unbiased and uncorrelated in time, H_k and M_k linear

Then objective function

$$\xi_0 \in \mathcal{S} \rightarrow$$

$$\mathcal{J}(\xi_0) = (1/2)\,(x_0^b - \xi_0)^T [P_0^b]^{-1} (x_0^b - \xi_0) + (1/2)\,\Sigma_k [y_k - H_k \xi_k]^T R_k^{-1} [y_k - H_k \xi_k]$$

subject to $\xi_{k+1} = M_k \xi_k$, $\qquad k = 0, \dots, K\text{-}1$

$$\mathcal{J}(\xi_0) = (1/2)\,(x_0^b - \xi_0)^T\,[P_0^b]^{-1}\,(x_0^b - \xi_0) + (1/2)\,\Sigma_k[y_k - H_k\xi_k]^T\,R_k^{-1}\,[y_k - H_k\xi_k]$$

Background is not necessary, if observations are in sufficient number to overdetermine the problem. Nor is strict linearity.

How to minimize objective function with respect to initial state $u = \xi_0$ (u is called the *control variable* of the problem) ?

Use iterative minimization algorithm, each step of which requires the explicit knowledge of the local gradient $\nabla_u \mathcal{J} \equiv (\partial \mathcal{J}/\partial u_i)$ of $\mathcal{J}$ with respect to u.

Gradient computed by *adjoint method*.

How to numerically compute the gradient $\nabla_u \mathcal{J}$?

Direct perturbation, in order to obtain partial derivatives $\partial\mathcal{J}/\partial u_i$ by finite differences ? That would require as many explicit computations of the objective function $\mathcal{J}$ as there are components in u. Practically impossible.

Adjoint Method

Input vector $\boldsymbol{u} = (u_i)$, $\dim \boldsymbol{u} = n$

Numerical process, implemented on computer (*e. g.* integration of numerical model)

$$\boldsymbol{u} \rightarrow \boldsymbol{v} = \boldsymbol{G}(\boldsymbol{u})$$

$\boldsymbol{v} = (v_j)$ is *output vector* , $\dim \boldsymbol{v} = m$

Perturbation $\delta\boldsymbol{u} = (\delta u_i)$ of input. Resulting first-order perturbation on $\boldsymbol{v}$

$$\delta v_j = \Sigma_i\, (\partial v_j/\partial u_i)\ \delta u_i$$

or, in matrix form

$$\delta\boldsymbol{v} \;=\; \boldsymbol{G}'\,\delta\boldsymbol{u}$$

where $\boldsymbol{G}' \equiv (\partial v_j/\partial u_i)$ is local matrix of partial derivatives, or jacobian matrix, of $\boldsymbol{G}$.

Adjoint Method (continued 1)

$$\delta\boldsymbol{v} \;=\; \boldsymbol{G}'\,\delta\boldsymbol{u} \qquad \text{(D)}$$

Scalar function of output

$$\mathcal{J}(\boldsymbol{v}) \;=\; \mathcal{J}[\boldsymbol{G}(\boldsymbol{u})]$$

Gradient $\nabla_{\boldsymbol{u}}\mathcal{J}$ of $\mathcal{J}$ with respect to input $\boldsymbol{u}$?

‘Chain rule’

$$\partial\mathcal{J}/\partial u_i = \Sigma_j\, \partial\mathcal{J}/\partial v_j\,(\partial v_j/\partial u_i)$$

or

$$\nabla_{\boldsymbol{u}}\mathcal{J} \;=\; \boldsymbol{G}'^{\mathrm{T}}\,\nabla_{\boldsymbol{v}}\mathcal{J} \qquad \text{(A)}$$

Adjoint Method (continued 2)

$\boldsymbol{G}$ is the composition of a number of successive steps

$$\boldsymbol{G} = \boldsymbol{G}_N \circ \ldots \circ \boldsymbol{G}_2 \circ \boldsymbol{G}_1$$

'Chain rule'

$$\boldsymbol{G}' = \boldsymbol{G}_N' \ldots \boldsymbol{G}_2' \, \boldsymbol{G}_1'$$

Transpose

$$\boldsymbol{G}'^{\mathrm{T}} = \boldsymbol{G}_1'^{\mathrm{T}} \, \boldsymbol{G}_2'^{\mathrm{T}} \ldots \boldsymbol{G}_N'^{\mathrm{T}}$$

Transpose, or *adjoint*, computations are performed in reversed order of direct computations.

If $\boldsymbol{G}$ is nonlinear, local jacobian $\boldsymbol{G}'$ depends on local value of input $\boldsymbol{u}$. Any quantity which is an argument of a nonlinear operation in the direct computation will be used again in the adjoint computation. It must be kept in memory from the direct computation (or else be recomputed again in the course of the adjoint computation).

If everything is kept in memory, total operation count of adjoint computation is at most 4 times operation count of direct computation (in practice about 2).

Adjoint Approach

$\mathcal{J}(\xi_0) = (1/2)\,(x_0^b - \xi_0)^{\mathrm{T}}\,[P_0^b]^{-1}\,(x_0^b - \xi_0) + (1/2)\,\Sigma_k[y_k - H_k\xi_k]^{\mathrm{T}}\,R_k^{-1}\,[y_k - H_k\xi_k]$

subject to $\xi_{k+1} = M_k\xi_k$, $\quad k = 0, \ldots, K\text{-}1$

Control variable $\quad \xi_0 = \boldsymbol{u}$

Adjoint equation

$$\lambda_K = \qquad H_K^{\mathrm{T}}\,R_K^{-1}\,[H_K\,\xi_K - y_K]$$

$$\lambda_k = M_k^{\mathrm{T}}\lambda_{k+1} + H_k^{\mathrm{T}}\,R_k^{-1}\,[H_k\,\xi_k - y_k] \qquad k = K\text{-}1, \ldots, 1$$

$$\lambda_0 = M_0^{\mathrm{T}}\lambda_1 + H_0^{\mathrm{T}}\,R_0^{-1}\,[H_0\,\xi_0 - y_0] + [P_0^b]^{-1}\,(\xi_0 - x_0^b)$$

$$\nabla_u\mathcal{J} = \lambda_0$$

Result of direct integration (ξ_k), which appears in quadratic terms in expression of objective function, must be kept in memory from direct integration.

Adjoint Approach (continued 2)

Nonlinearities ?

$$\mathcal{J}(\xi_0) \;=\; (1/2)\,(x_0^b - \xi_0)^T\,[P_0^b]^{-1}\,(x_0^b - \xi_0) + (1/2)\,\Sigma_k[y_k - H_k(\xi_k)]^T\,R_k^{-1}\,[y_k - H_k(\xi_k)]$$

subject to $\xi_{k+1} = M_k(\xi_k)$, $\qquad k = 0, \ldots, K\text{-}1$

Control variable $\qquad \xi_0 = \boldsymbol{u}$

Adjoint equation

$$\lambda_K = \qquad H_K'^T R_K^{-1}\,[H_K(\xi_K) - y_K]$$

$$\lambda_k = M_k'^T \lambda_{k+1} + H_k'^T R_k^{-1}\,[H_k(\xi_k) - y_k] \qquad k = K\text{-}1, \ldots, 1$$

$$\lambda_0 = M_0'^T \lambda_1 \;+ H_0'^T R_0^{-1}\,[H_0(\xi_0) - y_0] \;+\; [P_0^b]^{-1}\,(\xi_0 - x_0^b)$$

$$\nabla_u \mathcal{J} \;=\; \lambda_0$$

Not heuristic (it gives the exact gradient $\nabla_u\mathcal{J}$), and really used as described here.

How to write the adjoint of a code ?

Operation $a = b \times c$

Input b, c Output a but also b, c

For clarity, we write

$$a = b \times c$$
$$b' = b$$
$$c' = c$$

$\partial J/\partial a$, $\partial J/\partial b'$, $\partial J/\partial c'$ available. We want to determine $\partial J/\partial b$, $\partial J/\partial c$

Chain rule

$$\partial J/\partial b = (\partial J/\partial a)\underset{c}{(\partial a/\partial b)} + (\partial J/\partial b')\underset{1}{(\partial b'/\partial b)} + (\partial J/\partial c')\underset{0}{(\partial c'/\partial b)}$$

$$\partial J/\partial b = (\partial J/\partial a)\, c + \partial J/\partial b'$$

Similarly

$$\partial J/\partial c = (\partial J/\partial a)\, b + \partial J/\partial c'$$