Variational data assimilation for numerical weather prediction

Roel Stappers

Norwegian Meteorological Institute
roels@met.no

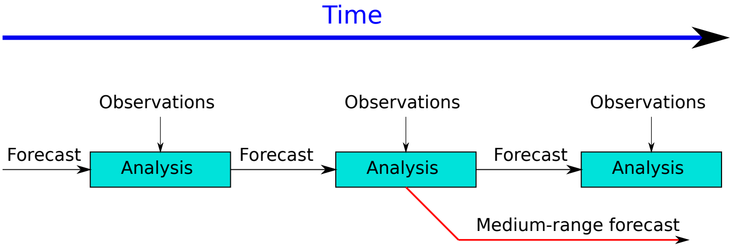
Hungarian Meteorological Service
February 14, 2018

Overview

- Variational Data assimilation (3DVAR, 4DVAR)
- Preconditioning
- Variational Bias Correction
- Background error covariance matrix
- Large scale constraint (Jk, LSMIX)
- The Object oriented prediction system (OOPS)

Data assimilation

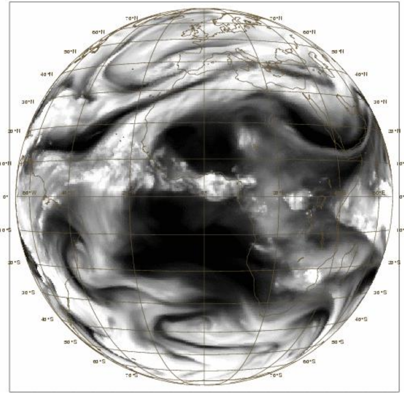
- The aim of data assimilation is to produce (optimal) estimates of the state of the atmosphere (the analysis) by combining information from observations with a short range forecast.
- The analysis is used as initial condition for the numerical weather prediction model to produce the forecast.



Observation operator

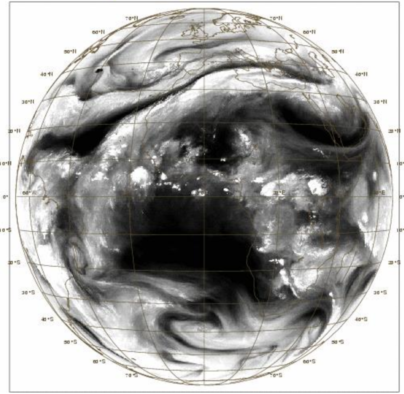
- The observation operator $\mathcal{H}$ maps model state to observation space.
- e.g.

Saturday 25 October 2003 12UTC ECMWF Forecast t+42 VT:Monday 27 October 2003 06UTC
Satellite Image, Water vapour



$\mathcal{H}(\mathbf{x})$

Satellite Image Monday 27 October 2003 0500UTC



$\mathbf{y}$

[0] Picture from Lars Isaksen

Variational data assimilation (3D-VAR)

In variational data assimilation the *analysis* $\mathbf{x}^a$ is the model state that minimizes the nonlinear cost function.

$$J(\mathbf{x}) = \underbrace{\frac{1}{2}\|\mathbf{x} - \mathbf{x}^b\|^2_{\mathbf{B}^{-1}}}_{J_b} + \underbrace{\frac{1}{2}\|\mathcal{H}(\mathbf{x}) - \mathbf{y}\|^2_{\mathbf{R}^{-1}}}_{J_o}$$

Where

- $\mathbf{x}^b$ is the *background* state. (a short range forecast).
- $\mathbf{y}$ is a vector with observations.
- $\mathbf{B}$ is the background error covariance. (Isotropic, homogeneous)
- $\mathbf{R}$ is the observation error covariance (Often simply Diagonal)
- $\mathcal{H}$ is the observation operator

Variational data assimilation (3D-VAR)

In variational data assimilation the *analysis* $\mathbf{x}^a$ is the model state that minimizes the nonlinear cost function.

$$J(\mathbf{x}) = \underbrace{\frac{1}{2}\|\mathbf{x} - \mathbf{x}^b\|^2_{\mathbf{B}^{-1}}}_{J_b} + \underbrace{\frac{1}{2}\|\mathcal{H}(\mathbf{x}) - \mathbf{y}\|^2_{\mathbf{R}^{-1}}}_{J_o}$$

Where

- $\mathbf{x}^b$ is the *background* state. (a short range forecast).
- $\mathbf{y}$ is a vector with observations.
- $\mathbf{B}$ is the background error covariance. (Isotropic, homogeneous)
- $\mathbf{R}$ is the observation error covariance (Often simply Diagonal)
- $\mathcal{H}$ is the observation operator

The cost function can be derived from Bayes' rule

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}$$

Assuming Gaussian errors and using maximum likelihood.

Variational data assimilation (4D-VAR)

In variational data assimilation the *analysis* $\mathbf{x}^a$ is the model state $\mathbf{x}$ that minimizes the nonlinear cost function.

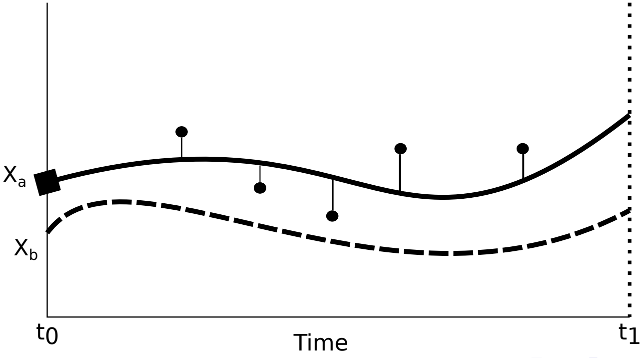
$$J(\mathbf{x}) = \frac{1}{2}\|\mathbf{x} - \mathbf{x}^b\|^2_{\mathbf{B}^{-1}} + \frac{1}{2}\|\mathcal{H}(\mathbf{x}) - \mathbf{y}\|^2_{\mathbf{R}^{-1}}$$

Variational data assimilation (4D-VAR)

In variational data assimilation the *analysis* $\mathbf{x}^a$ is the model state $\mathbf{x}$ that minimizes the nonlinear cost function.

$$J(\mathbf{x}) = \frac{1}{2}\|\mathbf{x} - \mathbf{x}^b\|^2_{\mathbf{B}^{-1}} + \frac{1}{2}\|\mathcal{H}(\mathbf{x}) - \mathbf{y}\|^2_{\mathbf{R}^{-1}}$$

In 4D-VAR nonlinear model integrations are performed in the observation operator $\mathcal{H}$ to compare with observations at the correct time.



Variational data assimilation (4D-VAR)

In variational data assimilation the *analysis* $\mathbf{x}^a$ is the model state $\mathbf{x}$ that minimizes the nonlinear cost function.

$$J(\mathbf{x}) = \frac{1}{2}\|\mathbf{x} - \mathbf{x}^b\|^2_{\mathbf{B}^{-1}} + \frac{1}{2}\|\mathcal{H}(\mathbf{x}) - \mathbf{y}\|^2_{\mathbf{R}^{-1}}$$

Variational data assimilation (4D-VAR)

In variational data assimilation the *analysis* $\mathbf{x}^a$ is the model state $\mathbf{x}$ that minimizes the nonlinear cost function.

$$J(\mathbf{x}) = \frac{1}{2}\|\mathbf{x} - \mathbf{x}^b\|^2_{\mathbf{B}^{-1}} + \frac{1}{2}\|\mathcal{H}(\mathbf{x}) - \mathbf{y}\|^2_{\mathbf{R}^{-1}}$$

In 4D-VAR nonlinear model integrations are performed in the observation operator $\mathcal{H}$ to compare with observations at the correct time.

$$J(\mathbf{x}) = \frac{1}{2}\|\mathbf{x} - \mathbf{x}^b\|^2_{\mathbf{B}^{-1}} + \frac{1}{2}\sum_{n=0}^{N}\|\mathcal{H}_n(\mathcal{M}_{t_0\rightarrow t_n}(\mathbf{x})) - \mathbf{y}_i\|^2_{\mathbf{R}_n^{-1}}$$

The challenges for data assimilation in NWP

- There are $\approx 10^8$ unknown parameters (temperature, humidity, pressure, wind speed and direction at each grid point in the model).
- The cost function is non-convex and the constraints are nonlinear.
- There is a strict time budget available for solving the problem.
- For 4DVAR: evaluations of the constraints via forward integration of the high resolution model is expensive and can only be performed a few times in each forecast cycle.

Incremental 3D/4D-VAR (Gauss-Newton approach)

The background $\mathbf{x}^b$ is normally a good estimate of the analysis $\mathbf{x}^a$.
Write $\mathbf{x} = \mathbf{x}^b + \delta\mathbf{x}$ then linearization of the observation operator gives:

$$\mathcal{H}(\mathbf{x}) \approx \mathcal{H}(\mathbf{x}^b) + \mathbf{H}_{\mathbf{x}^b}\delta\mathbf{x}$$

For simplicity we will drop the subscript $\mathbf{x}^b$ and write $\mathbf{H}$ for the linearized observation operator (Jacobian).
For 4D-VAR $\mathbf{H}$ includes the linearized model equations (with simplified physics).
The nonlinear cost function

$$J(\mathbf{x}) = \frac{1}{2}\|\mathbf{x} - \mathbf{x}^b\|^2_{\mathbf{B}^{-1}} + \frac{1}{2}\|\mathcal{H}(\mathbf{x}) - \mathbf{y}\|^2_{\mathbf{R}^{-1}}$$

can be written in incremental form

$$J(\delta\mathbf{x}) = \frac{1}{2}\|\delta\mathbf{x}\|^2_{\mathbf{B}^{-1}} + \frac{1}{2}\|\mathbf{H}\delta\mathbf{x} - \mathbf{d}\|^2_{\mathbf{R}^{-1}}$$

Where the innovation vector $\mathbf{d} = \mathbf{y} - \mathcal{H}(\mathbf{x}^b)$.
This is a standard (regularized) linear least squares problem.

Incremental 3D/4D-VAR (Gauss-Newton approach)

- The incremental cost function:

$$J(\delta\mathbf{x}) = \frac{1}{2}\|\delta\mathbf{x}\|^2_{\mathbf{B}^{-1}} + \frac{1}{2}\|\mathbf{H}\delta\mathbf{x} - \mathbf{d}\|^2_{\mathbf{R}^{-1}}$$

- Taking the gradient gives[1]

$$g(\delta\mathbf{x}) = \mathbf{B}^{-1}\delta\mathbf{x} + \mathbf{H}^T\mathbf{R}^{-1}(\mathbf{H}\delta\mathbf{x} - \mathbf{d})$$

- Setting the gradient to zero and rearranging gives

$$(\mathbf{B}^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})\delta\mathbf{x} = \mathbf{H}^T\mathbf{R}^{-1}\mathbf{d}$$

- Which gives

$$\delta\mathbf{x} = (\mathbf{B}^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{R}^{-1}\mathbf{d}$$

- Explicit computation of the inverse Hessian is not possible. Instead iterative methods are used (Either Quasi-Newton or Krylov subspace methods).

[1] The transpose $\mathbf{H}^T$ (a.k.a. adjoint) is derived manually from the $\mathbf{H}$ code

Iterative methods

Iterative methods that minimize the cost function follow the same basic pattern

1. At iterate $\mathbf{x}_k$ compute the cost function $J(\mathbf{x}_k)$ and the gradient $\mathbf{g}_k = \mathbf{g}(\mathbf{x}_k)$
2. Compute a search directions $\mathbf{p}_k$ and do a line search to (approximately) minimize $J(\mathbf{x}_k + \alpha_k \mathbf{p}_k)$ as a function of α_k
3. Set $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$
4. Check the convergence criteria. If not converged set $k := k + 1$ and repeat.

The methods differ in the way they update the search direction $\mathbf{p}_k$. A large class of methods use update equations of the form

$$\mathbf{p}_k = \beta_k \mathbf{p}_{k-1} - \mathbf{A}_k \mathbf{g}_k \quad (1)$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \quad (2)$$

Iterative methods: Quasi-Newton and conjugate gradient

$$\mathbf{p}_k = \beta_k \mathbf{p}_{k-1} - \mathbf{A}_k \mathbf{g}_k \tag{3}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \tag{4}$$

- Family of (nonlinear) conjugate gradient methods is obtain by setting $\mathbf{A}_k = \mathbf{I}$.

Table: Nonlinear conjugate gradient methods ($\mathbf{A}_k = \mathbf{I}$)

Name	β
Fletcher-Reeves	$\beta_k = \mathbf{g}_k^T \mathbf{g}_k / \mathbf{g}_{k-1}^T \mathbf{g}_{k-1}$
Polak-Ribière	$\beta_k = \mathbf{g}_k^T (\mathbf{g}_k - \mathbf{g}_{k-1}) / \mathbf{g}_{k-1}^T \mathbf{g}_{k-1}$
Hestenes-Stiefel	$\beta_k = \mathbf{g}_k^T (\mathbf{g}_k - \mathbf{g}_{k-1}) / (\mathbf{g}_k - \mathbf{g}_{k-1})^T \mathbf{p}_{k-1}$

For quadratic cost function the line search to compute α can be avoided

Iterative methods: Quasi-Newton and conjugate gradient

$$\mathbf{p}_k = \beta_k \mathbf{p}_{k-1} - \mathbf{A}_k \mathbf{g}_k \tag{5}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \tag{6}$$

The family of quasi-Newton methods is obtained by setting $\beta_k = 0$.

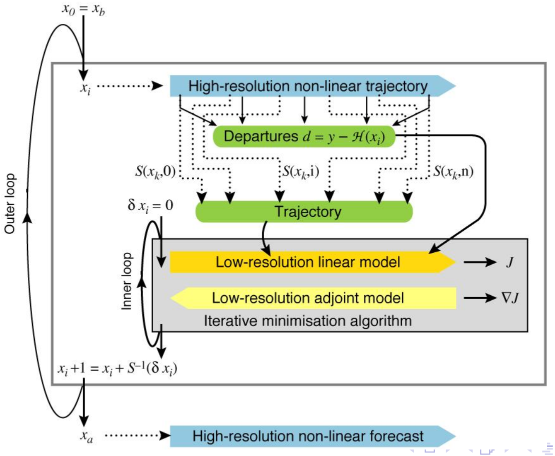
Table: Quasi-Newton methods ($\beta_k = 0$)

Name	Inverse Hessian approximation
BFGS	$\mathbf{A}_k =$ ... you can look this up
DFP	$\mathbf{A}_k = \mathbf{A}_{k-1} - \frac{\mathbf{A}_{k-1}\mathbf{y}_{k-1}\mathbf{y}_{k-1}^T\mathbf{A}_{k-1}^T}{\mathbf{y}_{k-1}^T\mathbf{A}_{k-1}\mathbf{y}_{k-1}} + \frac{\mathbf{s}_{k-1}\mathbf{s}_{k-1}^T}{\mathbf{y}_{k-1}^T\mathbf{s}_{k-1}}$
L-BFGS	$\mathbf{A}_k = ...$
Steepest descent	$\mathbf{A}_k = \mathbf{I}$

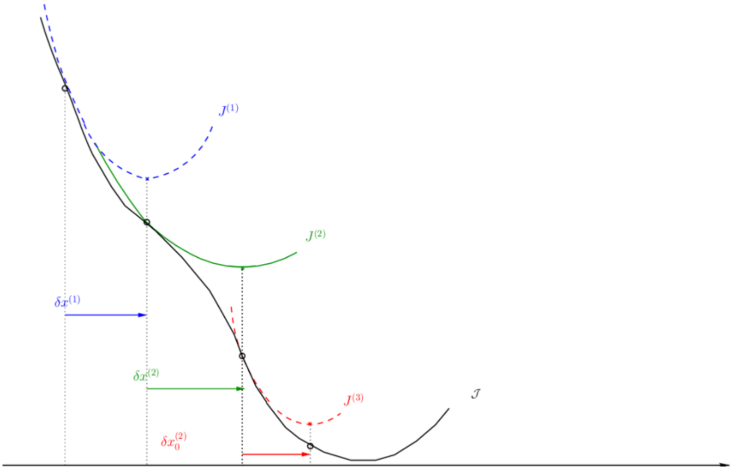
Where: $\mathbf{y}_{k-1} = \mathbf{g}_k - \mathbf{g}_{k-1}$ and $\mathbf{s}_k = \alpha_k \mathbf{p}_k$ often $\mathbf{A}_0 = \mathbf{I}$ or $\mathbf{A} = \mathrm{diag}(...)$.
In the code either M1QN3 (which implements an L-BFGS quasi-Netwon method) or conjugate gradient is used.

Multi-incremental strong constraint 4D-VAR

- The integrations with the linear and adjoint model in $\mathbf{H}$ and $\mathbf{H}^T$ are expensive and normally run at lower resolution with simplified physics.
- To take nonlinearities better into account the linear operators are relinearized around updated guesses in, so called, outer loops.



Multi-incremental strong constraint 4D-VAR



Convergence properties conjugate gradient method

For the linear system $Ax = b$. Let x_* be the minimum of the cost function and x_k the solution at inner loop iteration k Define $e_k = x_k - x_*$
Then we have

$$\|e_k\|_A \leq \left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1} \right)^k \|e_0\|_A$$

Where $\kappa(A) = \frac{\sigma_{max}(A)}{\sigma_{min}(A)}$ denotes the condition number.
The pattern of the increment is related to the shape of the leading eigenvector of the Hessian of the 4D-Var cost function. Andersson et al. (2000) have shown that this eigenvector is driven by the density and accuracy of observations. They have shown that, in a simplified example with n observations in the same location, an approximation of the condition number of the minimisation problem is given by: [2]

$$\kappa \approx 2n \frac{\sigma_b^2}{\sigma_o^2} + 1$$

[2] taken From Tremolet, a incremental 4D-VAR convergence study.

First level Preconditioning

- Let $\mathbf{B} = \mathbf{U}\mathbf{U}^T$. It can be shown that the solution is of the form

$$\delta\mathbf{x} = \mathbf{U}\chi$$

- Using this *control variable transform*[3] the cost function is

$$J(\chi) = \frac{1}{2}\|\chi\|^2 + \frac{1}{2}\|\mathbf{H}\mathbf{U}\chi - \mathbf{d}\|^2_{\mathbf{R}^{-1}}$$

- With gradient[4]

$$g(\chi) = \chi + \mathbf{U}^T\mathbf{H}^T\mathbf{R}^{-1}(\mathbf{H}\mathbf{U}\chi - \mathbf{d})$$

- Setting the gradient of the preconditioned system to zero then gives

$$(\mathbf{I} + \mathbf{U}^T\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{U})\chi = \mathbf{U}^T\mathbf{H}^T\mathbf{R}^{-1}\mathbf{d}$$

- The identity matrix here ensure that all eigenvalues are ≥ 1

[3] See chavarin.F90 for implementation

[4] See sim4d.F90 for implementation (add VarBC, TOVSCV, and allow for FG$\neq$BG)

Modelling the B matrix

Derber and Bouttier suggested [5]

$$\zeta_{bal} = \zeta \tag{7}$$

$$D_u = D - D_{bal} P_{bal}(\zeta) \tag{8}$$

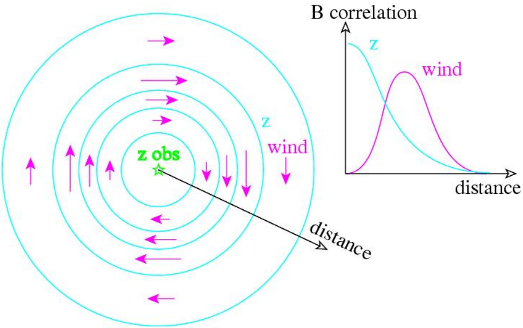
$$T_u = T - T_{bal} P_{bal}(\zeta) - T_{div}(D_u) \tag{9}$$

$$(p_s)_u = p_s - (p_s)_{bal} P_{bal}(\zeta) - (p_s)_{div}(D_u) \tag{10}$$

- P_{bal} is a linearized mass variable, determined by statistical regression between spectral coefficients of vorticity and geopotential.
- Tbal (etc.) is determined by statistical regression between geopotential and temperature (etc.).
- T_{div} [and $(p_s)_{div}$] are given by statistical regression between temperature [and ps] and divergence.
- ζ_{bal}, D_u, T_u, $(p_s)_u$ are assumed to be uncorrelated.

[5] See also https://www.ecmwf.int/sites/default/files/elibrary/2003/9404-background-error-covariance-modelling.pdf

B matrix



Variational Bias correction

Up to now it has been been assumed that observations are unbiased or have been bias corrected.
Modification of cost function

$$J(\mathbf{x}, \beta) = \underbrace{\frac{1}{2}\|\mathbf{x} - \mathbf{x}^b\|^2_{\mathbf{B}^{-1}}}_{J_b} + \underbrace{\frac{1}{2}\|\beta - \beta^b\|^2_{\mathbf{B}_\beta^{-1}}}_{J_p} + \underbrace{\frac{1}{2}\|\mathcal{H}(\mathbf{x}) + \mathbf{P}\beta - \mathbf{y}\|^2_{\mathbf{R}^{-1}}}_{J_o}$$

[6] https://www.ecmwf.int/sites/default/files/elibrary/2004/8930-variational-bias-correction-radiance-data-ecmwf-system.pdf

Variational Bias correction

Up to now it has been been assumed that observations are unbiased or have been bias corrected.
Modification of cost function

$$J(\mathbf{x}, \beta) = \underbrace{\frac{1}{2}\|\mathbf{x} - \mathbf{x}^b\|^2_{\mathbf{B}^{-1}}}_{J_b} + \underbrace{\frac{1}{2}\|\beta - \beta^b\|^2_{\mathbf{B}_\beta{}^{-1}}}_{J_p} + \underbrace{\frac{1}{2}\|\mathcal{H}(\mathbf{x}) + \mathbf{P}\beta - \mathbf{y}\|^2_{\mathbf{R}^{-1}}}_{J_o}$$

- Note 1) in LAM the background for the VarBC parameters is taken from the previous day not from the previous cycle.
- Also the control variable transform for VarBC is not based purely on B_β but include estimate of $\mathbf{P}^T\mathbf{R}^{-1}\mathbf{P}$. [6]

[6] https://www.ecmwf.int/sites/default/files/elibrary/2004/8930-variational-bias-correction-radiance-data-ecmwf-system.pdf

Jk and LSMIX

To take advantage of the high quality ECMWF forecast for the large scales an addition background term J_k can be added to the cost function

$$J(\mathbf{x}) = \underbrace{\frac{1}{2}\|\mathbf{x} - \mathbf{x}^b\|^2_{\mathbf{B}^{-1}}}_{J_b} + \underbrace{\frac{1}{2}\|\mathbf{x} - \mathbf{x}_{LS}\|^2_{\mathbf{V}^{-1}}}_{J_k} + \underbrace{\frac{1}{2}\|\mathcal{H}(\mathbf{x}) - \mathbf{y}\|^2_{\mathbf{R}^{-1}}}_{J_o}$$

Here V^{-1} will only penalize deviations for the large scales. It can be shown that adding J_k is equivalent to [7]

$$J(\mathbf{x}) = \frac{1}{2}\|\mathbf{x} - \tilde{\mathbf{x}}^b\|^2_{\tilde{\mathbf{B}}^{-1}} + \frac{1}{2}\|\mathcal{H}(\mathbf{x}) - \mathbf{y}\|^2_{\mathbf{R}^{-1}}$$

In Harmonie a simplified versions is available[8] Activated by setting LSMIXBC=true in config_exp.h

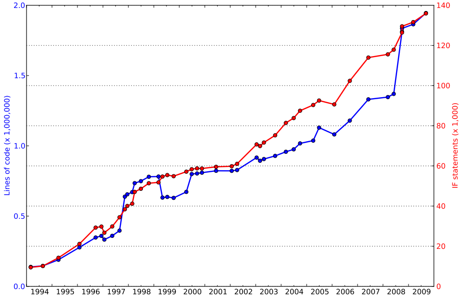
---

[7] https://hirlam.org/trac/wiki/HarmonieSystemDocumentation/38h1.1/LSMIXandJk

[8] https://hirlam.org/trac/attachment/wiki/HarmonieSystemDocumentation/lsmixbc.ppt

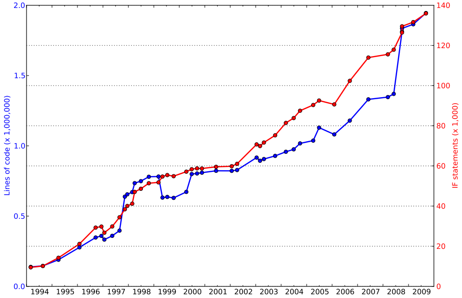
OOPS

The object oriented prediction system



- Fortran code becomes difficult to maintain and new data assimilation techniques become difficult to implement.
- Concerns about scalability of model and DA for >100k cores

The object oriented prediction system



- Fortran code becomes difficult to maintain and new data assimilation techniques become difficult to implement.
- Concerns about scalability of model and DA for >100k cores

ECMWF has decided to recode the "top-level" routines in C++ to obtain are more flexible/modular code in which it is easier to formulate new DA algorithms.

Formulations of DA and flexibility in OOPS

Primal formulation ($\mathbf{d} = \mathbf{y} - \mathcal{H}(x_0^g)$, $b = x_0^b - x_0^g$)

$$(\mathbf{B}^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})\delta x_0 = \mathbf{B}^{-1}b + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{d}$$

Formulations of DA and flexibility in OOPS

Primal formulation ($\mathbf{d} = \mathbf{y} - \mathcal{H}(x_0^g)$, $b = x_0^b - x_0^g$)

$$(\mathbf{B}^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})\delta x_0 = \mathbf{B}^{-1}b + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{d}$$

Saddle point formulation

$$\begin{bmatrix} \mathbf{B}^{-1} & \mathbf{H}^T \\ \mathbf{H} & -\mathbf{R} \end{bmatrix} \begin{bmatrix} \delta x \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathbf{B}^{-1}b \\ \mathbf{d} \end{bmatrix}$$

Dual formulation (3D/4D-PSAS)

$$(\mathbf{HBH}^T + \mathbf{R})\lambda = -\mathbf{d} + \mathbf{H}b$$
$$\delta x = -\mathbf{BH}^T\lambda + b$$

Formulations of DA and flexibility in OOPS

Primal formulation ($\mathbf{d} = \mathbf{y} - \mathcal{H}(x_0^g)$, $b = x_0^b - x_0^g$)

$$(\mathbf{B}^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})\delta x_0 = \mathbf{B}^{-1}b + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{d}$$

Saddle point formulation

$$\begin{bmatrix} \mathbf{B}^{-1} & \mathbf{H}^T \\ \mathbf{H} & -\mathbf{R} \end{bmatrix} \begin{bmatrix} \delta x \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathbf{B}^{-1}b \\ \mathbf{d} \end{bmatrix}$$

Dual formulation (3D/4D-PSAS)

$$(\mathbf{HBH}^T + \mathbf{R})\lambda = -\mathbf{d} + \mathbf{H}b$$

$$\delta x = -\mathbf{BH}^T\lambda + b$$

Weak constraint 4D-VAR

$$(\mathbf{L}^T\mathbf{D}^{-1}\mathbf{L} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})\delta\mathbf{x} = \mathbf{L}^T\mathbf{D}^{-1}\mathbf{b} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{d}$$

- Saddle point weak constraint 4D-VAR etc. EDA, EnKF, ETKF
- Flexibility to change linear equation solvers (PCG, MINRES, RPCG, GMRES)

Other issues

- Flow dependent background covariance matrices
- Long window weak constraint 4D-VAR
- Ensemble data assimilation (Hybrid systems between ensemble Kalman filtering and 4D-VAR)

Other issues

- Flow dependent background covariance matrices
- Long window weak constraint 4D-VAR
- Ensemble data assimilation (Hybrid systems between ensemble Kalman filtering and 4D-VAR)

Thank you!

Single Obs

Exercise: Show that

$$\delta \mathbf{x} = (\mathbf{B}^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{R}^{-1}\mathbf{d}$$

Can also be written as

$$\delta \mathbf{x} = \mathbf{B}\mathbf{H}^T(\mathbf{R} + \mathbf{H}\mathbf{B}\mathbf{H}^T)^{-1}\mathbf{d}$$

For a single obs $(\mathbf{R} + \mathbf{H}\mathbf{B}\mathbf{H}^T)^{-1}\mathbf{d}$ is a scalar showing that a single obs experiment can be used to visualize the structure (length scales and correclations) of the **B** matrix

Adjoint code an example

- Suppose in nonlinear model we have the statement: $x = y + z^2$
- Corresponding line in tangent linear code: $\delta x = \delta y + 2z\delta z$
- Write as a matrix

$$\begin{bmatrix} \delta z \\ \delta y \\ \delta x \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 2z & 1 & 0 \end{bmatrix} \begin{bmatrix} \delta z \\ \delta y \\ \delta x \end{bmatrix}$$

- In the adjoint code corresponding statement is

$$\begin{bmatrix} \delta z^* \\ \delta y^* \\ \delta x^* \end{bmatrix} = \begin{bmatrix} 1 & 0 & 2z \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta z^* \\ \delta y^* \\ \delta x^* \end{bmatrix}$$

- i.e

$$\delta z^* = \delta z^* + 2z\delta x^*$$
$$\delta y^* = \delta y^* + \delta x^*$$
$$\delta x^* = 0$$

Adjoint test and gradient test

- By definition the adjoint

$$(\mathbf{H}\delta\mathbf{x}, \delta\mathbf{y}) = \left\langle \delta\mathbf{x}, \mathbf{H}^T\delta\mathbf{y} \right\rangle$$

Validity of the adjoint can be test by computing the left and right hand side for some δx and δy. Equality should hold up to machine precision.

- Gradient test

$$\lim_{h\to 0} \frac{J(\chi + h\delta\chi) - J(\chi)}{\langle \nabla J, h\delta\chi \rangle}$$

ratio should approach 1 for small enough values of h (but not too small because of round-off errors)
See NAMVARTEST logicals LADTEST LGRTEST.