Data assimilation for large scale non linear porblems

Eric Blayo   Arthur Vidard

LABORATOIRE JEAN KUNTZMANN MATHÉMATIQUES APPLIQUÉES - INFORMATIQUE

informatics mathematics

UNIVERSITÉ GRENOBLE ALPES

Peyresq, 25^{th} – 26^{th} of April 2015

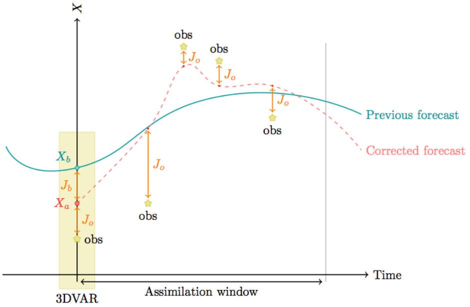
Peyresq Summer School 2015

Generalization: variational approach

Stationary case: $J(\mathbf{x}) = \underbrace{\frac{1}{2}\|\mathbf{x}-\mathbf{x}_b\|_b^2}_{\text{background term } J_b} + \underbrace{\frac{1}{2}\|H(\mathbf{x})-\mathbf{y}\|_o^2}_{\text{observation term } J_o}$

Time dependent case:

$$
\begin{aligned}
J(\mathbf{x}_0) &= \frac{1}{2}\|\mathbf{x}_0-\mathbf{x}_0^b\|_b^2 + \frac{1}{2}\sum_{i=0}^{N}\|H_i(\mathbf{x}(t_i))-\mathbf{y}(t_i)\|_o^2 \\
&= \frac{1}{2}\|\mathbf{x}_0-\mathbf{x}_b\|_b^2 + \frac{1}{2}\sum_{i=0}^{N}\|H_i(M_{0\rightarrow t_i}(\mathbf{x}_0))-\mathbf{y}(t_i)\|_o^2
\end{aligned}
$$



Generalization: statistical approach

Let $\mathbf{X}_b = \mathbf{x} + \varepsilon_b$ and $\mathbf{Y} = \mathbf{H}\mathbf{x} + \varepsilon_o$

Hypotheses:

- $E(\varepsilon_b) = 0$ — unbiased background
- $E(\varepsilon_o) = 0$ — unbiased measurement devices
- $\mathrm{Cov}(\varepsilon_b, \varepsilon_o) = 0$ — independent background and observation errors
- $\mathrm{Cov}(\varepsilon_b) = \mathbf{B}$ et $\mathrm{Cov}(\varepsilon_o) = \mathbf{R}$ — known accuracies and covariances

Statistical approach: BLUE

$$\hat{\mathbf{X}} = \mathbf{X}_b + \underbrace{(\mathbf{B}^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{R}^{-1}}_{\text{gain matrix}} \underbrace{(\mathbf{Y} - \mathbf{H}\mathbf{X}_b)}_{\text{innovation vector}}$$

with $\left[\mathrm{Cov}(\hat{\mathbf{X}})\right]^{-1} = \mathbf{B}^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}$ — accuracies are added

Links between both approaches

Statistical approach: BLUE

$$\hat{\mathbf{X}} = \mathbf{X}_b + (\mathbf{B}^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{R}^{-1}(\mathbf{Y} - \mathbf{H}\mathbf{X}_b)$$

with $\mathrm{Cov}(\hat{\mathbf{X}}) = (\mathbf{B}^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}$

Variational approach in the linear stationary case

$$\begin{aligned} J(\mathbf{x}) &= \frac{1}{2}\|\mathbf{x} - \mathbf{x}_b\|_b^2 + \frac{1}{2}\|H(\mathbf{x}) - \mathbf{y}\|_o^2 \\ &= \frac{1}{2}(\mathbf{x} - \mathbf{x}_b)^T\mathbf{B}^{-1}(\mathbf{x} - \mathbf{x}_b) + \frac{1}{2}(\mathbf{H}\mathbf{x} - \mathbf{y})^T\mathbf{R}^{-1}(\mathbf{H}\mathbf{x} - \mathbf{y}) \end{aligned}$$

$$\min_{\mathbf{x}\in\mathbb{R}^n} J(\mathbf{x}) \longrightarrow \hat{\mathbf{x}} = \mathbf{x}_b + (\mathbf{B}^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{R}^{-1}(\mathbf{y} - \mathbf{H}\mathbf{x}_b)$$

Same remarks as previously

- The statistical approach rationalizes the choice of the norms for J_o and J_b in the variational approach.
- $\underbrace{\left[\mathrm{Cov}(\hat{\mathbf{X}})\right]^{-1}}_{\text{accuracy}} = \mathbf{B}^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H} = \underbrace{\mathrm{Hess}(J)}_{\text{convexity}}$

If the problem is time dependent

Dynamical system: $\mathbf{x}^t(t_{k+1}) = \mathbf{M}(t_k, t_{k+1})\,\mathbf{x}^t(t_k) + \mathbf{e}(t_k)$

- $\mathbf{x}^t(t_k)$ true state at time t_k
- $\mathbf{M}(t_k, t_{k+1})$ model assumed linear between t_k and t_{k+1}
- $\mathbf{e}(t_k)$ model error at time t_k

Observations $\mathbf{y}_k$ distributed in time.

Hypotheses

- $\mathbf{e}(t_k)$ is unbiased, with covariance matrix $\mathbf{Q}_k$
- $\mathbf{e}(t_k)$ and $\mathbf{e}(t_l)$ are independent $(k \neq l)$
- Unbiased observation $\mathbf{y}_k$, with error covariance matrix $\mathbf{R}_k$
- $\mathbf{e}(t_k)$ and analysis error $\mathbf{x}^a(t_k) - \mathbf{x}^t(t_k)$ are independent

If the problem is time dependent

Kalman filter (Kalman and Bucy, 1961)

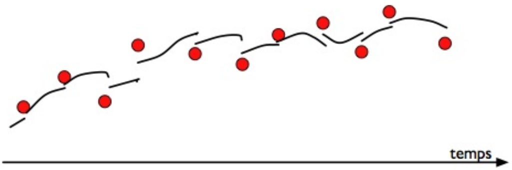
Initialization:

$$\begin{aligned} \mathbf{x}^a(t_0) &= \mathbf{x}_0 && \textit{approximate initial state} \\ \mathbf{P}^a(t_0) &= \mathbf{P}_0 && \textit{error covariance matrix} \end{aligned}$$

Step k: *(prediction - correction, or forecast - analysis)*

$$\begin{aligned} \mathbf{x}^f_{k+1} &= \mathbf{M}_{k,k+1}\,\mathbf{x}^a_k && \text{Forecast} \\ \mathbf{P}^f_{k+1} &= \mathbf{M}_{k,k+1}\mathbf{P}^a_k\mathbf{M}^T_{k,k+1} + \mathbf{Q}_k \end{aligned}$$

$$\begin{aligned} \mathbf{x}^a_{k+1} &= \mathbf{x}^f_{k+1} + \mathbf{K}_{k+1}\left[\mathbf{y}_{k+1} - \mathbf{H}_{k+1}\mathbf{x}^f_{k+1}\right] && \text{BLUE analysis} \\ \mathbf{K}_{k+1} &= \mathbf{P}^f_{k+1}\mathbf{H}^T_{k+1}\left[\mathbf{H}_{k+1}\mathbf{P}^f_{k+1}\mathbf{H}^T_{k+1} + \mathbf{R}_{k+1}\right]^{-1} \\ \mathbf{P}^a_{k+1} &= \mathbf{P}^f_{k+1} - \mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{P}^f_{k+1} \end{aligned}$$



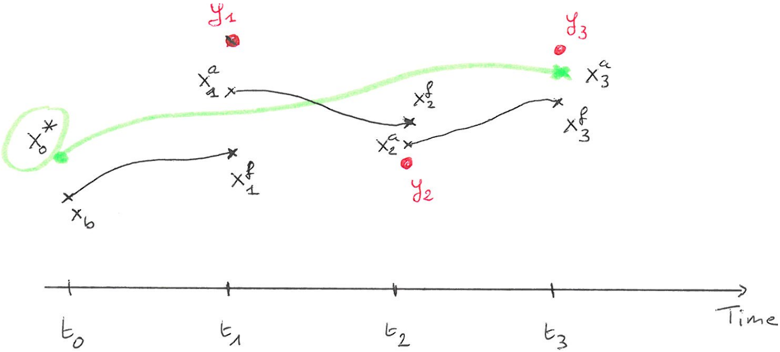
If the problem is time dependent

Equivalence with the variational approach

If $\mathbf{H}_k$ and $\mathbf{M}(t_k, t_{k+1})$ are linear, and if the model is perfect ($\mathbf{e}_k = 0$), then the Kalman filter and the variational method minimizing

$$J(\mathbf{x}) = \frac{1}{2}(\mathbf{x} - \mathbf{x}_0)^T \mathbf{P}_0^{-1}(\mathbf{x} - \mathbf{x}_0) + \frac{1}{2}\sum_{k=0}^{N}(\mathbf{H_k M}(t_0, t_k)\mathbf{x} - \mathbf{y}_k)^T \mathbf{R}_k^{-1}(\mathbf{H_k M}(t_0, t_k)\mathbf{x} - \mathbf{y}_k)$$

lead to the same solution at $t = t_N$.



Common main methodological difficulties

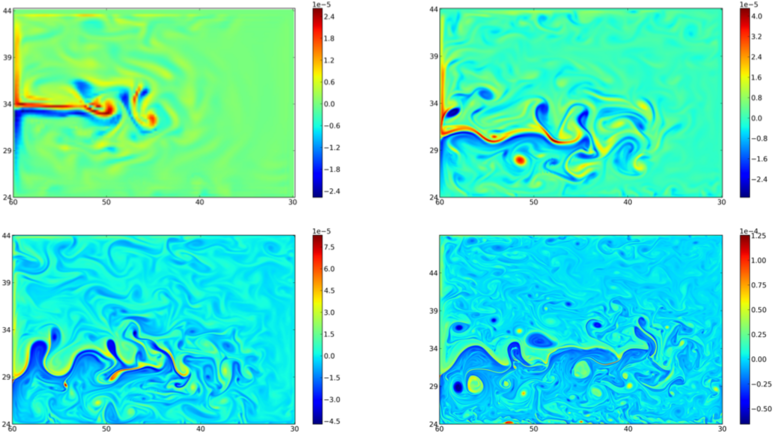
- Non linearities: J non quadratic / what about Kalman filter ?
- Huge dimensions $[\mathbf{x}] = \mathcal{O}(10^6 - 10^9)$: minimization of J / management of huge matrices
- Poorly known error statistics: choice of the norms / $\mathbf{B}, \mathbf{R}, \mathbf{Q}$

- Scientific computing issues (data management, code efficiency, parallelization...)

$\longrightarrow$ TODAY's LECTURE

Towards larger dimensions and stronger nonlinearities

Increasing the model resolution increases the size of the state variable and, for a number of applications, allows for stronger scale interactions.



Snapshots of the surface relative vorticity in the SEABASS configuration of NEMO, for different model resolutions: $1/4^\circ$, $1/12^\circ$, $1/24^\circ$ and $1/100^\circ$.

Outline

Non linearities: extended Kalman filter

The Kalman filter assumes that M and H are linear. If not: linearization

Reminder: derivatives and gradients

$$f : E \longrightarrow F \qquad (E, F \text{ being of finite or infinite dimension})$$

- Gradient (or Fréchet derivative): E, F being Hilbert spaces, f is Fréchet differentiable at point $x \in E$ iff

$$\exists p \in E \text{ such that } f(x+h) = f(x) + p.h + o(\|h\|) \quad \forall h \in E$$

  p is the derivative or gradient of f at point x, denoted $f'(x)$ or $\nabla f(x)$.

- $h \rightarrow p(x).h$ is a linear function, called differential function or tangent linear function or Jacobian of f at point x

Non linearities: extended Kalman filter

The Kalman filter assumes that M and H are linear. If not: linearization

$$\mathbf{x}^f_{k+1} = M_{k,k+1}(\mathbf{x}^a_k) \simeq M_{k,k+1}(\mathbf{x}^t_k) + \mathbf{M}_{k,k+1} \underbrace{(\mathbf{x}^a_k - \mathbf{x}^t_k)}_{\mathbf{e}^a_k}$$

$$\Longrightarrow \quad \mathbf{x}^f_{k+1} - \mathbf{x}^t_{k+1} = \mathbf{e}^f_{k+1} = \underbrace{M_{k,k+1}(\mathbf{x}^t_k) - \mathbf{x}^t_{k+1}}_{\mathbf{e}_k} + \mathbf{M}_{k,k+1}\mathbf{e}^a_k$$

$$\Longrightarrow \quad \mathbf{P}^f_{k+1} = \mathrm{Cov}(\mathbf{e}^f_{k+1}) = \mathbf{M}_{k,k+1}\mathbf{P}^a_k\mathbf{M}^T_{k,k+1} + \mathbf{Q}_k$$

and similarly for the other equations of the filter

Non linearities: extended Kalman filter

Extended Kalman filter

Initialization:

$$\begin{aligned} \mathbf{x}^a(t_0) &= \mathbf{x}_0 && \textit{approximate initial state} \\ \mathbf{P}^a(t_0) &= \mathbf{P}_0 && \textit{error covariance matrix} \end{aligned}$$

Step k: *(prediction - correction, or forecast - analysis)*

$$\begin{aligned} \mathbf{x}^f_{k+1} &= M_{k,k+1}(\mathbf{x}^a_k) && \text{Forecast} \\ \mathbf{P}^f_{k+1} &= \mathbf{M}_{k,k+1}\mathbf{P}^a_k\mathbf{M}^T_{k,k+1} + \mathbf{Q}_k \end{aligned}$$

$$\begin{aligned} \mathbf{x}^a_{k+1} &= \mathbf{x}^f_{k+1} + \mathbf{K}_{k+1}\left[\mathbf{y}_{k+1} - H_{k+1}(\mathbf{x}^f_{k+1})\right] && \text{BLUE analysis} \\ \mathbf{K}_{k+1} &= \mathbf{P}^f_{k+1}\mathbf{H}^T_{k+1}\left[\mathbf{H}_{k+1}\mathbf{P}^f_{k+1}\mathbf{H}^T_{k+1} + \mathbf{R}_{k+1}\right]^{-1} \\ \mathbf{P}^a_{k+1} &= \mathbf{P}^f_{k+1} - \mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{P}^f_{k+1} \end{aligned}$$

Non linearities: extended Kalman filter

Extended Kalman filter

Initialization:
$$\begin{aligned} \mathbf{x}^a(t_0) &= \mathbf{x}_0 && \textit{approximate initial state} \\ \mathbf{P}^a(t_0) &= \mathbf{P}_0 && \textit{error covariance matrix} \end{aligned}$$

Step k: *(prediction - correction, or forecast - analysis)*

$$\begin{aligned} \mathbf{x}^f_{k+1} &= M_{k,k+1}(\mathbf{x}^a_k) && \text{Forecast} \\ \mathbf{P}^f_{k+1} &= \mathbf{M}_{k,k+1}\mathbf{P}^a_k\mathbf{M}^T_{k,k+1} + \mathbf{Q}_k \end{aligned}$$

$$\begin{aligned} \mathbf{x}^a_{k+1} &= \mathbf{x}^f_{k+1} + \mathbf{K}_{k+1}\left[\mathbf{y}_{k+1} - H_{k+1}(\mathbf{x}^f_{k+1})\right] && \text{BLUE analysis} \\ \mathbf{K}_{k+1} &= \mathbf{P}^f_{k+1}\mathbf{H}^T_{k+1}\left[\mathbf{H}_{k+1}\mathbf{P}^f_{k+1}\mathbf{H}^T_{k+1} + \mathbf{R}_{k+1}\right]^{-1} \\ \mathbf{P}^a_{k+1} &= \mathbf{P}^f_{k+1} - \mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{P}^f_{k+1} \end{aligned}$$

- OK if nonlinearities are not too strong
- Requires the availability of $\mathbf{M}_{k,k+1}$ and $\mathbf{H}_k$
- More sophisticated approaches have been developed $\longrightarrow$ unscented Kalman filter (exact up to second order, requires no tangent linear model nor Hessian matrix)

Huge dimension: reduced order filters

As soon as $[\mathbf{x}]$ becomes huge, it's no longer possible to handle the covariance matrices.

Idea: a large part of the system variability can be represented (or is assumed to) in a reduced dimension space.

$\longrightarrow$ RRSQRT filter, SEEK filter, SEIK filter...

Huge dimension: reduced order filters

Example: Reduced Rank SQuare Root filter

- $\mathbf{P}_0^f \simeq \mathbf{S}_0^f \left(\mathbf{S}_0^f\right)^T$ with size$(\mathbf{S}_0^f) = (n, r)$ (r leading modes, $r \ll n$)
- This is injected in the filter equations. This leads for instance to $\mathbf{P}_k^a = \mathbf{S}_k^a \left(\mathbf{S}_k^a\right)^T$,

$$\mathbf{S}_k^a = \underbrace{\mathbf{S}_k^f}_{(n,r)} \left(\underbrace{\mathbf{I}_r - \mathbf{\Psi}_k^T [\mathbf{\Psi}_k \mathbf{\Psi}_k^T + \mathbf{R}_k]^{-1} \mathbf{\Psi}_k}_{(r,r)} \right)^{1/2} \qquad \text{where } \mathbf{\Psi}_k = \underbrace{\mathbf{H}_k \mathbf{S}_k^f}_{(p,r)}$$

Pros: most computations in low dimension
Cons: choice and time evolution of the modes

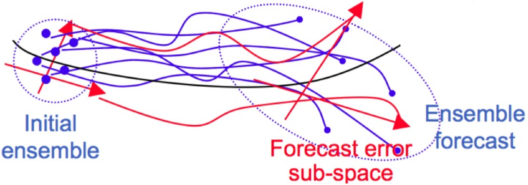
A widely used filter: the Ensemble Kalman filter

- addresses both problems of non linearities and huge dimension
- rather simple and intuitive

Idea: generation of an ensemble of N trajectories, by N perturbations of the set of observations (consistently with $\mathbf{R}$). Standard extended Kalman filter, with covariance matrices computed using the ensemble:

$$\mathbf{P}_k^f = \frac{1}{N-1}\sum_{j=1}^{N}(\mathbf{x}_{j,k}^f - \bar{x}_k^f)(\mathbf{x}_{j,k}^f - \bar{x}_k^f)^T \text{ with} \bar{x}_k^f = \frac{1}{N}\sum_{j=1}^{N}\mathbf{x}_{j,k}^f$$

$$\mathbf{P}_k^a = \frac{1}{N-1}\sum_{j=1}^{N}(\mathbf{x}_{j,k}^a - \bar{x}_k^a)(\mathbf{x}_{j,k}^f - \bar{x}_k^f)^T \text{ with} \bar{x}_k^a = \frac{1}{N}\sum_{j=1}^{N}\mathbf{x}_{j,k}^a$$



Outline

Reminder: derivatives and gradients

$$f : E \longrightarrow \mathbf{R} \qquad (E \text{ being of finite or infinite dimension})$$

- Directional (or Gâteaux) derivative of f at point $x \in E$ in direction $d \in E$:

$$\frac{\partial f}{\partial d}(x) = \hat{f}[x](d) = \lim_{\alpha \to 0} \frac{f(x + \alpha d) - f(x)}{\alpha}$$

Example: partial derivatives $\dfrac{\partial f}{\partial x_i}$ are directional derivatives in the direction of the members of the canonical basis $(d = e_i)$

Reminder: derivatives and gradients

$$f : E \longrightarrow \mathbf{R} \qquad (E \text{ being of finite or infinite dimension})$$

- Gradient (or Fréchet derivative): E being an Hilbert space, f is Fréchet differentiable at point $x \in E$ iff

$$\exists p \in E \text{ such that } f(x+h) = f(x) + (p,h) + o(\|h\|) \quad \forall h \in E$$

  p is the derivative or gradient of f at point x, denoted $f'(x)$ or $\nabla f(x)$.

- $h \to (p(x), h)$ is a linear function, called differential function or tangent linear function or Jacobian of f at point x

Reminder: derivatives and gradients

$$f : E \longrightarrow \mathbf{R} \qquad (E \text{ being of finite or infinite dimension})$$

- Gradient (or Fréchet derivative): E being an Hilbert space, f is Fréchet differentiable at point $x \in E$ iff

$$\exists p \in E \text{ such that } f(x+h) = f(x) + (p,h) + o(\|h\|) \quad \forall h \in E$$

  p is the derivative or gradient of f at point x, denoted $f'(x)$ or $\nabla f(x)$.

- $h \rightarrow (p(x), h)$ is a linear function, called differential function or tangent linear function or Jacobian of f at point x

- Important (obvious) relationship: $\dfrac{\partial f}{\partial d}(x) = (\nabla f(x), d)$
- Fréchet $\Longrightarrow$ Gâteaux

Getting the gradient is not obvious

The computation of $\nabla J(\mathbf{x}_k)$ may be difficult if the dependency of J with regard to the control variable $\mathbf{x}$ is not direct.

It is often difficult (or even impossible) to obtain the gradient through the computation of growth rates.

Example:

$$\begin{cases} \dfrac{d\mathbf{x}(t))}{dt} = M(\mathbf{x}(t)) \quad t \in [0, T] \\ \mathbf{x}(t=0) = \mathbf{u} \end{cases} \qquad \text{with } \mathbf{u} = \begin{pmatrix} u_1 \\ \vdots \\ u_N \end{pmatrix}$$

$$J(\mathbf{u}) = \frac{1}{2}\int_0^T \|\mathbf{x}(t) - \mathbf{x}^{\text{obs}}(t)\|^2 \qquad \longrightarrow \text{ requires one model run}$$

$$\nabla J(\mathbf{u}) = \begin{pmatrix} \dfrac{\partial J}{\partial u_1}(\mathbf{u}) \\ \vdots \\ \dfrac{\partial J}{\partial u_N}(\mathbf{u}) \end{pmatrix} \simeq \begin{pmatrix} [J(\mathbf{u} + \alpha\,\mathbf{e}_1) - J(\mathbf{u})]\,/\alpha \\ \vdots \\ [J(\mathbf{u} + \alpha\,\mathbf{e}_N) - J(\mathbf{u})]\,/\alpha \end{pmatrix}$$

$\longrightarrow$ $N+1$ model runs

Getting the gradient is not obvious

In actual applications like meteorology / oceanography, $N = [\mathbf{u}] = \mathcal{O}(10^6 - 10^9) \quad \longrightarrow$ this method cannot be used.

Alternatively, the adjoint method provides a very efficient way to compute ∇J.

Getting the gradient is not obvious

In actual applications like meteorology / oceanography, $N = [\mathbf{u}] = \mathcal{O}(10^6 - 10^9)$ $\longrightarrow$ this method cannot be used.

Alternatively, the adjoint method provides a very efficient way to compute ∇J.

On the contrary, do not forget that, if the size of the control variable is very small (< 10), ∇J can be easily estimated by the computation of growth rates.

Reminder: adjoint operator

- General definition:
  Let $\mathcal{X}$ and $\mathcal{Y}$ two prehilbertian spaces (i.e. vector spaces with scalar products).
  Let $A : \mathcal{X} \longrightarrow \mathcal{Y}$ an operator.
  The adjoint operator $A^* : \mathcal{Y} \longrightarrow \mathcal{X}$ is defined by:

$$\forall x \in \mathcal{X}, \forall y \in \mathcal{Y}, \qquad < Ax, y >_{\mathcal{Y}} = < x, A^* y >_{\mathcal{X}}$$

  In the case where $\mathcal{X}$ and $\mathcal{Y}$ are Hilbert spaces and A is linear, then A^* always exists (and is unique).

Reminder: adjoint operator

- **General definition:**
  Let $\mathcal{X}$ and $\mathcal{Y}$ two prehilbertian spaces (i.e. vector spaces with scalar products).
  Let $A : \mathcal{X} \longrightarrow \mathcal{Y}$ an operator.
  The adjoint operator $A^* : \mathcal{Y} \longrightarrow \mathcal{X}$ is defined by:

$$\forall x \in \mathcal{X}, \forall y \in \mathcal{Y}, \qquad < Ax, y >_{\mathcal{Y}} = < x, A^* y >_{\mathcal{X}}$$

  In the case where $\mathcal{X}$ and $\mathcal{Y}$ are Hilbert spaces and A is linear, then A^* always exists (and is unique).

- **Adjoint operator in finite dimension:**
  $A : \mathbf{R}^n \longrightarrow \mathbf{R}^m$ a linear operator (i.e. a matrix). Then its adjoint operator A^* (w.r. to Euclidian norms) is A^T.

let F be a non-linear Fréchet differentiable operator. If $J(x) = \langle F(x), F(x) \rangle$

$$
\begin{aligned}
\hat{J}[x](d) &= \lim_{\alpha \to 0} \frac{\langle F(x+\alpha d), F(x+\alpha d)\rangle - \langle F(x), F(x)\rangle}{\alpha} \\
&= \lim_{\alpha \to 0} \frac{\left\langle F(x) + \alpha \hat{F}[x](d) + o(\alpha), F(x) + \alpha \hat{F}[x](d) + o(\alpha) \right\rangle - \langle F(x), F(x)\rangle}{\alpha} \\
&= \lim_{\alpha \to 0} \frac{2\left\langle \alpha \hat{F}[x](d) + o(\alpha), F(x) \right\rangle + \left\langle \alpha \hat{F}[x](d) + o(\alpha), \alpha \hat{F}[x](d) + o(\alpha) \right\rangle}{\alpha} \\
&= 2\left\langle \hat{F}[x](d), F(x) \right\rangle
\end{aligned}
$$

$\hat{F}[x](.)$ being a linear operator,

$$
J[x](d) = 2\left\langle \hat{F}[x](d), F(x) \right\rangle = 2\left\langle d, \hat{F}[x]^*(F(x)) \right\rangle = \langle d, \nabla J \rangle
$$

Hence $\quad \nabla J = 2\hat{F}[x]^*(F(x))$

or in finite dimension: $\quad \nabla J = 2\mathbf{F}^T F(x)$

let F be a non-linear Fréchet differentiable operator. If $J(x) = \langle F(x), F(x) \rangle$

$$
\begin{aligned}
\hat{J}[x](d) &= \lim_{\alpha \to 0} \frac{\langle F(x+\alpha d), F(x+\alpha d)\rangle - \langle F(x), F(x)\rangle}{\alpha} \\
&= \lim_{\alpha \to 0} \frac{\left\langle F(x) + \alpha \hat{F}[x](d) + o(\alpha), F(x) + \alpha \hat{F}[x](d) + o(\alpha) \right\rangle - \langle F(x), F(x)\rangle}{\alpha} \\
&= \lim_{\alpha \to 0} \frac{2\left\langle \alpha \hat{F}[x](d) + o(\alpha), F(x) \right\rangle + \left\langle \alpha \hat{F}[x](d) + o(\alpha), \alpha \hat{F}[x](d) + o(\alpha) \right\rangle}{\alpha} \\
&= 2\left\langle \hat{F}[x](d), F(x) \right\rangle
\end{aligned}
$$

$\hat{F}[x](.)$ being a linear operator,

$$J[x](d) = 2\left\langle \hat{F}[x](d), F(x) \right\rangle = 2\left\langle d, \hat{F}[x]^*(F(x)) \right\rangle = \langle d, \nabla J \rangle$$

Hence $\qquad \nabla J = 2\hat{F}[x]^*(F(x))$

or in finite dimension: $\qquad \nabla J = 2\mathbf{F}^T F(x)$

Why do we care?

if $\qquad J(\mathbf{x}_0) = \frac{1}{2}(\mathbf{y}_i^o - H_i(M_{0,i}(\mathbf{x}_0)))^T \mathbf{R}_i^{-1}(\mathbf{y}_i^o - H_i(M_{0,i}(\mathbf{x}_0)))$

then $\qquad \nabla J = \mathbf{M}_{0,i}^T \mathbf{H}_i^T \mathbf{R}_i^{-1}(\mathbf{y}_i^o - H_i(M_{0,i}(\mathbf{x}_0)))$

4D-Var: Optimality System

- The direct model
$$\begin{cases} \mathbf{x}_i = M_{i-1,i}(\mathbf{x}_{i-1}), & i = 1, N \\ \mathbf{x}_0 = & \mathbf{x}^b \end{cases}$$

- The cost function

$$J(\mathbf{x}_0) = \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}_0^b)^T \mathbf{B}^{-1}(\mathbf{x}_0 - \mathbf{x}_0^b) + \frac{1}{2}\sum_{i=0}^{n}(\mathbf{y}_i^o - H_i(\mathbf{x}_i))^T \mathbf{R}_i^{-1}(\mathbf{y}_i^o - H_i(\mathbf{x}_i))$$

4D-Var: Optimality System

- The direct model and its tangent (Gâteaux-derivative, direction $\mathbf{d}$)

$$\begin{cases} \mathbf{x}_i = M_{i-1,i}(\mathbf{x}_{i-1}), & i = 1, N \\ \mathbf{x}_0 = \mathbf{x}^b \end{cases} \qquad \begin{cases} \hat{\mathbf{x}}_i = \mathbf{M}_{i-1,i}\hat{\mathbf{x}}_{i-1}, & i = 1, N \\ \hat{\mathbf{x}}_0 = \mathbf{d} \end{cases}$$

- The cost function

$$J(\mathbf{x}_0) = \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}_0^b)^T \mathbf{B}^{-1}(\mathbf{x}_0 - \mathbf{x}_0^b) + \frac{1}{2}\sum_{i=0}^{n}(\mathbf{y}_i^o - H_i(\mathbf{x}_i))^T \mathbf{R}_i^{-1}(\mathbf{y}_i^o - H_i(\mathbf{x}_i))$$

4D-Var: Optimality System

- The direct model

$$\begin{cases} \mathbf{x}_i = M_{i-1,i}(\mathbf{x}_{i-1}), & i = 1, N \\ \mathbf{x}_0 = \mathbf{x}^b \end{cases}$$

- The cost function

$$J(\mathbf{x}_0) = \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}_0^b)^T \mathbf{B}^{-1}(\mathbf{x}_0 - \mathbf{x}_0^b) + \frac{1}{2}\sum_{i=0}^{n}(\mathbf{y}_i^o - H_i(\mathbf{x}_i))^T \mathbf{R}_i^{-1}(\mathbf{y}_i^o - H_i(\mathbf{x}_i))$$

- According to previous slide: $\nabla_{\mathbf{x}_0} J^o = \sum_{i=0}^{n} \mathbf{M}_{0,i}^T \mathbf{H}_i^T \mathbf{R}_i^{-1}(\mathbf{y}_i^o - H_i(M_{0,i}(\mathbf{x}_0)))$

4D-Var: Optimality System

- The direct model

$$\begin{cases} \mathbf{x}_i = M_{i-1,i}(\mathbf{x}_{i-1}), & i = 1, N \\ \mathbf{x}_0 = & \mathbf{x}^b \end{cases}$$

- The cost function

$$J(\mathbf{x}_0) = \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}_0^b)^T \mathbf{B}^{-1}(\mathbf{x}_0 - \mathbf{x}_0^b) + \frac{1}{2}\sum_{i=0}^{n}(\mathbf{y}_i^o - H_i(\mathbf{x}_i))^T \mathbf{R}_i^{-1}(\mathbf{y}_i^o - H_i(\mathbf{x}_i))$$

- The adjoint model (backward)

$$\begin{cases} \mathbf{x}_i^* = \mathbf{M}_{i,i+1}^T \mathbf{x}_{i+1}^* + \mathbf{H}_i^T \mathbf{R}_i^{-1}(\mathbf{y}_i^o - H_i(\mathbf{x}_i)), & i = N, 1 \\ \mathbf{x}_n^* = & \mathbf{H}_n^T \mathbf{R}_n^{-1}(\mathbf{y}_i^o - H_i(\mathbf{x}_i)) \end{cases}$$

- Euler equation

$$\nabla_{\mathbf{x}_0} J = \mathbf{B}^{-1}(\mathbf{x}_0 - \mathbf{x}_0^b) + \mathbf{x}_0^* (= 0)$$

4D-Var: Algorithm

Algorithm (4D-Var)

- *Initialization :* $\mathbf{x} = \mathbf{x}^0$, $n = 0$
- *While* $\|\nabla J\| > \varepsilon$ *or* $n \leq n_{max}$, *do :*
  1. *Compute* J *thanks to the direct model* M *and the observation operator* H
  2. *Compute* ∇J *thanks to the backward integration of the adjoint model* $\mathbf{M}^T$ *and the adjoint of the observation operator* $\mathbf{H}^T$.
  3. *Descent and update of* $\mathbf{x}$
  4. $n = n + 1$

Incremental 4D-Var

Remarks: Non linearities If M and H are non linear, the computation of the gradient by adjoint methods remains exact, but J becomes non convex and therefore difficult to minimize.

In the case of M and/or H non linear, in order to avoid the minimization of a non convex function, the incremental 4D-Var algorithm has been implemented. This is an approximation of the 4D-Var, making the so-called Tangent Linear Appoximation (TLA)

Tangent Linear Approximation

$$M_{0\to i}(\mathbf{x}_0) - M_{0\to i}(\mathbf{x}_0^b) \simeq \mathbf{M}_{0\to i}(\mathbf{x}_0 - \mathbf{x}_0^b)$$

and

$$H_i(\mathbf{x}_i) - H_i(\mathbf{x}_i^b) \simeq \mathbf{H}_i(\mathbf{x}_i - \mathbf{x}^b)$$

Incremental 4D-Var (or Gauss-Newton)

Defining

- increment $\delta \mathbf{x}_0^k = \mathbf{x}_0^k - \mathbf{x}_0^b$
- innovation vector $\mathbf{d}_i^k = \mathbf{y}_i^o - H_i(M_{0\to i}(\mathbf{x}^{(k-1)}))$

Incremental 4D-Var cost function:

$$\tilde{J}^k(\delta \mathbf{x}_0^k) = (\delta \mathbf{x}_0^k)^T \mathbf{B}^{-1} \delta \mathbf{x}_0^k + \sum_{i=1}^{n} (\mathbf{d}_i^k - \mathbf{H}_i \mathbf{M}_i \ldots \mathbf{M}_1 \delta \mathbf{x}_0^k)^T \mathbf{R}_i^{-1} (\mathbf{d}_i^k - \mathbf{H}_i \mathbf{M}_i \ldots \mathbf{M}_1 \delta \mathbf{x}_0^k)$$

$\tilde{J}(\delta \mathbf{x}_0^k)$ is now a quadratic function and therefore has a unique minimum.

During the minimization, $\delta \mathbf{x}^k$ will grow and become too large, then the TLH will be invalidated. In order to sort this out, one stops the minimization, updates the linearized operators $\mathbf{M}_i$ and $\mathbf{H}_i$ and the innovation vector $\mathbf{d}^k$ by recomputing the non linear trajectory starting from the initial condition $\mathbf{x}_0^{(k+1)} = \mathbf{x}_0^b + \delta \mathbf{x}_0^k$.

Algorithm (incremental 4D-Var)

– *Initialization :* $\mathbf{x}_0^r = \mathbf{x}_0^b$

($\mathbf{x}^r$ is called reference state ; $\mathbf{x}_0^b$ is the first guess).

START THE OUTER LOOP

- *Non linear model integration:* $\mathbf{x}_i^r = M_{0\to i}[\mathbf{x}^r]$;
  - *compute the* $\mathbf{d}_i = \mathbf{y}_i^o - H_i(\mathbf{x}_i^r)$
  - *store the non linear trajectory (or some checkpoints)* $\mathbf{x}_i^r$ *for the tangent and adjoint integrations*

  START THE INNER LOOP
  - *Linear model integration:*$\delta\mathbf{x}_i = \mathbf{M}_{0\to i}\delta\mathbf{x}$
    - *compute* $\mathbf{d}_i^o - \mathbf{H}_i(\mathbf{x}_i)$
  - *Adjoint model backward integration* $\delta^*\mathbf{x}_i = \mathbf{M}_{N\to i}^T\delta^*\mathbf{x}_N - \mathbf{H}_i{}^T[\mathbf{d}_i^o - \mathbf{H}_i\mathbf{x}_i]$
  - $\nabla J = \delta^*\mathbf{x}_0$
  - *update* $\delta\mathbf{x}_0$ *thanks to the descent algorithm*

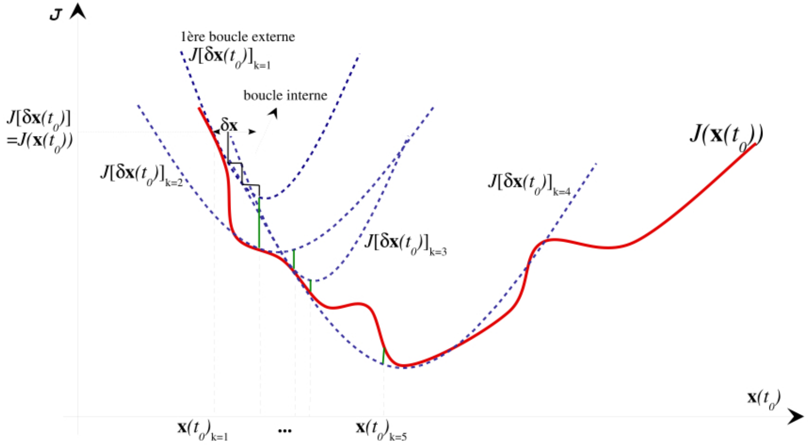
    END OF THE INNER LOOP
  - *Update the reference state* $\mathbf{x}_0^r = \mathbf{x}_0^r + \delta\mathbf{x}_0^a$

  END OF THE OUTER LOOP

  – *Compute the final analysis:* $\mathbf{x}_0^a = \mathbf{x}_0^r$, $\mathbf{x}_i^a = M_{0,i}(\mathbf{x}_0^a)$.

Note: under some "reasonable" assumption the incremental 4D-Var converges toward the minimum of the original 4D-Var problem.

Incremental 4D-Var



Multi-incremental 4D-Var

One can use other kind of hypothesis than the TLH in this framework. In Numercal Weather Prediction center (ECMWF, Météo-France, UK-MetOffice, ...) use the so-called Multi-incremental approach: the models used in the successive minimizations of the inner loop are approximation of the tangent model (lower resolution and simplified physics). The first outer loop is performed using a very coarse grid for the inner loops models, and the resolution (and the physics) is improve for each subsequent outer loops.
The hypothesis here is :

$$M_{0\to i}(\mathbf{x}_0^b + \mathbf{S}^{-I}\delta\mathbf{x}_0^s) - M_{0\to i}(\mathbf{x}_0^b) \simeq \mathbf{S}^{-I}\mathbf{M}_{0\to i}^s\delta\mathbf{x}_0^s$$

the exponent s meaning simplified model/state vector

Algorithm (multi-incremental 4D-Var)

– *Initialization :* $\mathbf{x}_0^r = \mathbf{x}_0^b$
($\mathbf{x}^r$ *is called reference state ;* $\mathbf{x}_0^b$ *is the first guess).*

START THE OUTER LOOP

- *Non linear model integration:* $\mathbf{x}_i^r = M_{0\to i}[\mathbf{x}^r]$;
  - *compute the* $\mathbf{d}_i = \mathbf{y}_i^o - H_i(\mathbf{x}_i^r)$
  - *store the simplified non linear trajectory (or some checkpoints)* $\mathbf{S}\mathbf{x}_i^r$ *for the tangent and adjoint integrations*

  START THE INNER LOOP

  - *Simplified linear model integration:*$\delta\mathbf{x}_i^s = \mathbf{M}_{0\to i}^s \delta\mathbf{x}^s$
    - *compute* $\mathbf{d}_i^o - \mathbf{H}_i^s \delta\mathbf{x}_i^s$
  - *Adjoint model backward integration* $\delta^*\mathbf{x}_i^s = \mathbf{M}_{N\to i}^{s\,T} \delta^*\mathbf{x}_N^s - \mathbf{H}_i^{s\,T}[\mathbf{d}_i^o - \mathbf{H}_i^s \delta\mathbf{x}_i^s]$
  - $\nabla J = \delta^*\mathbf{x}_0^s$
  - *update* $\delta\mathbf{x}_0^s$ *thanks to the descent algorithm*

    END OF THE INNER LOOP
  - *Update the analysis increment* $\delta\mathbf{x}_0^a = \mathbf{S}^{-I}\delta\mathbf{x}_0^s$
  - *Update the reference state* $\mathbf{x}_0^r = \mathbf{x}_0^r + \delta\mathbf{x}_0^a$

END OF THE OUTER LOOP

– *Compute the final analysis:* $\mathbf{x}_0^a = \mathbf{x}_0^r$, $\mathbf{x}_i^a = M_{0,i}(\mathbf{x}_0^a)$.

S is different from one outer iteration to the other (from simpler to more realistic).

3D-Fgat

The 3D-FGAT (First Guess at Appropriate Time) is a further approximation of the incremental 4D-Var algorithm where the evolution of the increment during the assimilation window is assume to be stationnary, *i.e.*:

$$M_{0\to i}(\mathbf{x}_0^b + \delta\mathbf{x}_0) - M_{0\to i}(\mathbf{x}_0^b) \simeq \delta\mathbf{x}_0$$

In other words it assumes that $\mathbf{M} = \mathbf{I}$ and $\mathbf{M}^T = \mathbf{I}$ for the length of the assimilation window.

Algorithm (3D-FGAT)

– *Initialization :* $\mathbf{x}_0^r = \mathbf{x}_0^b$
($\mathbf{x}^r$ is called reference state ; $\mathbf{x}_0^b$ is the first guess).

START THE OUTER LOOP

- *Non linear model integration:* $\mathbf{x}_i^r = M_{0\rightarrow i}[\mathbf{x}^r]$;
  - *compute the* $\mathbf{d}_i = \mathbf{y}_i^o - H_i(\mathbf{x}_i^r)$
  - *store the non linear trajectory* $\mathbf{x}_i^r$ *for the tangent and adjoint Observation operator (if required)*

  START THE INNER LOOP
  - *Linear model integration:*$\delta\mathbf{x}_i = \mathbf{M}_{0\rightarrow i}\delta\mathbf{x}$
    - *compute* $\mathbf{d}_i^o - \mathbf{H}_i\delta\mathbf{x}_0$
  - $\nabla J = -\sum_i \mathbf{H}_i{}^T[\mathbf{d}_i^o - \mathbf{H}_i\delta\mathbf{x}_0]$
  - *update* $\delta\mathbf{x}_0$ *thanks to the descent algorithm*

    END OF THE INNER LOOP
  - *Update the reference state* $\mathbf{x}_0^r = \mathbf{x}_0^r + \delta\mathbf{x}_0^a$

  END OF THE OUTER LOOP

  – *Compute the final analysis:* $\mathbf{x}_0^a = \mathbf{x}_0^r$, $\mathbf{x}_i^a = \mathbf{M}_{0,i}[\mathbf{x}_0^a]$.

- 4D-Var: $\mathbf{x}_i = M_{0,i}(\mathbf{x}_0)$

$$J(\mathbf{x}_0) = \frac{1}{2}\sum_{i=0}^{N}\big(H(\mathbf{x}_i) - \mathbf{y}_i\big)\mathbf{R}^{-1}\big(H(\mathbf{x}_i) - \mathbf{y}_i\big) + \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}^b)^T\mathbf{B}_i^{-1}(\mathbf{x}_0 - \mathbf{x}^b)$$

- 4D-Var: $\mathbf{x}_i = M_{0,i}(\mathbf{x}_0)$

$$J(\mathbf{x}_0) = \frac{1}{2}\sum_{i=0}^{N}\big(H(\mathbf{x}_i) - \mathbf{y}_i\big)\mathbf{R}^{-1}\big(H(\mathbf{x}_i) - \mathbf{y}_i\big) + \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}^b)^T\mathbf{B}_i^{-1}(\mathbf{x}_0 - \mathbf{x}^b)$$

- incremental 4D-Var: $M_{0,N}(\mathbf{x}_0 + \delta\mathbf{x}_0) \simeq M_{0,N}(\mathbf{x}_0) + \mathbf{M}_{0,N}\delta\mathbf{x}_0$

$$J(\delta\mathbf{x}_0) = \frac{1}{2}\sum_{i=0}^{N}\big(\mathbf{H}\delta\mathbf{x}_i - \mathbf{d}_i\big)\mathbf{R}^{-1}\big(\mathbf{H}\delta\mathbf{x}_i - \mathbf{d}_i\big) + \frac{1}{2}\delta\mathbf{x}_0^T\mathbf{B}_i^{-1}\delta\mathbf{x}_0$$

- 4D-Var: $\mathbf{x}_i = M_{0,i}(\mathbf{x}_0)$

$$J(\mathbf{x}_0) = \frac{1}{2}\sum_{i=0}^{N}\left(H(\mathbf{x}_i) - \mathbf{y}_i\right)\mathbf{R}^{-1}\left(H(\mathbf{x}_i) - \mathbf{y}_i\right) + \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}^b)^T\mathbf{B}_i^{-1}(\mathbf{x}_0 - \mathbf{x}^b)$$

- incremental 4D-Var: $M_{0,N}(\mathbf{x}_0 + \delta\mathbf{x}_0) \simeq M_{0,N}(\mathbf{x}_0) + \mathbf{M}_{0,N}\delta\mathbf{x}_0$

$$J(\delta\mathbf{x}_0) = \frac{1}{2}\sum_{i=0}^{N}\left(\mathbf{H}\delta\mathbf{x}_i - \mathbf{d}_i\right)\mathbf{R}^{-1}\left(\mathbf{H}\delta\mathbf{x}_i - \mathbf{d}_i\right) + \frac{1}{2}\delta\mathbf{x}_0^T\mathbf{B}_i^{-1}\delta\mathbf{x}_0$$

- multi-incremental 4D-Var: $M_{0,N}(\mathbf{x}_0 + \delta\mathbf{x}_0) \simeq M_{0,N}(\mathbf{x}_0) + \mathbf{S}^{-I}\mathbf{M}_{0,N}^L\delta\mathbf{x}_0^L$

$$J(\delta\mathbf{x}_0^L) = \frac{1}{2}\sum_{i=0}^{N}\left(\mathbf{H}^L\delta\mathbf{x}_i^L - \mathbf{d}_i\right)\mathbf{R}^{-1}\left(\mathbf{H}^L\delta\mathbf{x}_i^L - \mathbf{d}_i\right) + \frac{1}{2}(\delta\mathbf{x}_0^L)^T\mathbf{B}_i^{-1}\delta\mathbf{x}_0^L$$

- 4D-Var: $\mathbf{x}_i = M_{0,i}(\mathbf{x}_0)$

$$J(\mathbf{x}_0) = \frac{1}{2}\sum_{i=0}^{N}\left(H(\mathbf{x}_i) - \mathbf{y}_i\right)\mathbf{R}^{-1}\left(H(\mathbf{x}_i) - \mathbf{y}_i\right) + \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}^b)^T\mathbf{B}_i^{-1}(\mathbf{x}_0 - \mathbf{x}^b)$$

- incremental 4D-Var: $M_{0,N}(\mathbf{x}_0 + \delta\mathbf{x}_0) \simeq M_{0,N}(\mathbf{x}_0) + \mathbf{M}_{0,N}\delta\mathbf{x}_0$

$$J(\delta\mathbf{x}_0) = \frac{1}{2}\sum_{i=0}^{N}\left(\mathbf{H}\delta\mathbf{x}_i - \mathbf{d}_i\right)\mathbf{R}^{-1}\left(\mathbf{H}\delta\mathbf{x}_i - \mathbf{d}_i\right) + \frac{1}{2}\delta\mathbf{x}_0^T\mathbf{B}_i^{-1}\delta\mathbf{x}_0$$

- multi-incremental 4D-Var: $M_{0,N}(\mathbf{x}_0 + \delta\mathbf{x}_0) \simeq M_{0,N}(\mathbf{x}_0) + \mathbf{S}^{-l}\mathbf{M}_{0,N}^L\delta\mathbf{x}_0^L$

$$J(\delta\mathbf{x}_0^L) = \frac{1}{2}\sum_{i=0}^{N}\left(\mathbf{H}^L\delta\mathbf{x}_i^L - \mathbf{d}_i\right)\mathbf{R}^{-1}\left(\mathbf{H}^L\delta\mathbf{x}_i^L - \mathbf{d}_i\right) + \frac{1}{2}(\delta\mathbf{x}_0^L)^T\mathbf{B}_i^{-1}\delta\mathbf{x}_0^L$$

- 3D-FGAT : $M_{0,N}(\mathbf{x}_0 + \delta\mathbf{x}_0) \simeq M_{0,N}(\mathbf{x}_0) + \delta\mathbf{x}_0$

$$J(\delta\mathbf{x}_0) = \frac{1}{2}\sum_{i=0}^{N}\left(\mathbf{H}\delta\mathbf{x}_0 - \mathbf{d}_i\right)\mathbf{R}^{-1}\left(\mathbf{H}\delta\mathbf{x}_0 - \mathbf{d}_i\right) + \frac{1}{2}\delta\mathbf{x}_0^T\mathbf{B}_i^{-1}\delta\mathbf{x}_0$$

- 4D-Var: $\mathbf{x}_i = M_{0,i}(\mathbf{x}_0)$

$$J(\mathbf{x}_0) = \frac{1}{2}\sum_{i=0}^{N}\left(H(\mathbf{x}_i) - \mathbf{y}_i\right)\mathbf{R}^{-1}\left(H(\mathbf{x}_i) - \mathbf{y}_i\right) + \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}^b)^T\mathbf{B}_i^{-1}(\mathbf{x}_0 - \mathbf{x}^b)$$

- incremental 4D-Var: $M_{0,N}(\mathbf{x}_0 + \delta\mathbf{x}_0) \simeq M_{0,N}(\mathbf{x}_0) + \mathbf{M}_{0,N}\delta\mathbf{x}_0$

$$J(\delta\mathbf{x}_0) = \frac{1}{2}\sum_{i=0}^{N}\left(\mathbf{H}\delta\mathbf{x}_i - \mathbf{d}_i\right)\mathbf{R}^{-1}\left(\mathbf{H}\delta\mathbf{x}_i - \mathbf{d}_i\right) + \frac{1}{2}\delta\mathbf{x}_0^T\mathbf{B}_i^{-1}\delta\mathbf{x}_0$$

- multi-incremental 4D-Var: $M_{0,N}(\mathbf{x}_0 + \delta\mathbf{x}_0) \simeq M_{0,N}(\mathbf{x}_0) + \mathbf{S}^{-l}\mathbf{M}_{0,N}^L\delta\mathbf{x}_0^L$

$$J(\delta\mathbf{x}_0^L) = \frac{1}{2}\sum_{i=0}^{N}\left(\mathbf{H}^L\delta\mathbf{x}_i^L - \mathbf{d}_i\right)\mathbf{R}^{-1}\left(\mathbf{H}^L\delta\mathbf{x}_i^L - \mathbf{d}_i\right) + \frac{1}{2}(\delta\mathbf{x}_0^L)^T\mathbf{B}_i^{-1}\delta\mathbf{x}_0^L$$

- 3D-FGAT : $M_{0,N}(\mathbf{x}_0 + \delta\mathbf{x}_0) \simeq M_{0,N}(\mathbf{x}_0) + \delta\mathbf{x}_0$

$$J(\delta\mathbf{x}_0) = \frac{1}{2}\sum_{i=0}^{N}\left(\mathbf{H}\delta\mathbf{x}_0 - \mathbf{d}_i\right)\mathbf{R}^{-1}\left(\mathbf{H}\delta\mathbf{x}_0 - \mathbf{d}_i\right) + \frac{1}{2}\delta\mathbf{x}_0^T\mathbf{B}_i^{-1}\delta\mathbf{x}_0$$

- 3D-Var : $M_{0,N}(\mathbf{x}_0 + \delta\mathbf{x}_0) \simeq \mathbf{x}_0 + \delta\mathbf{x}_0$

$$J(\mathbf{x}_0) = (\mathbf{x}_0 - \mathbf{x}_0^b)^T\mathbf{B}_0^{-1}(\mathbf{x}_0 - \mathbf{x}_0^b) + \sum_{i=0}^{N}\left(H_i(\mathbf{x}_0) - \mathbf{y}_i\right)^T\mathbf{R}_i^{-1}\left(H_i(\mathbf{x}_0 - \mathbf{y}_i)\right)$$

Outline

Definition

The conditioning of the minimisation is defined by the ratio between the larger and the smaller eigenvalue of $\mathcal{H}$ the Hessian of J. The larger this number is, the more ill-conditioned the problem is. This is the main characteristic that affect the minimization efficiency.

In the following, for the sake of simplicity, we will assume that M and H are linear, and we will use a cost function such as:

$$J(\delta\mathbf{x}_0) = \frac{1}{2}(\delta\mathbf{x}_0)^T\mathbf{B}^{-1}\delta\mathbf{x}_0 + \frac{1}{2}[\mathbf{d}^o - \mathbf{G}\delta\mathbf{x}_0]^T\mathbf{R}^{-1}[\mathbf{d}^o - \mathbf{G}\delta\mathbf{x}_0]$$

$$\delta\mathbf{x}_0 = \mathbf{x}_0 - \mathbf{x}_0^b; \mathbf{G} = \mathbf{HM}; \mathbf{d}^o = \mathbf{y}^o - \mathbf{H}M(\mathbf{x}_0)^b$$

then the hessian can be written as:

$$\mathcal{H} = \mathbf{B}^{-1} + \mathbf{G}^T\mathbf{R}^{-1}\mathbf{G}$$

and we can show that the minimization is equivalent of the inversion of the Hessian.

Change of Variable

Ideally, if we make the following change of variable:

$$\mathbf{v} = \mathcal{H}^{\frac{1}{2}} \delta \mathbf{x}_0$$

$(\mathcal{H} = \mathcal{H}^{\frac{T}{2}} \mathcal{H}^{\frac{1}{2}})$
the cost function becomes:

$$J_{\mathbf{v}}(\mathbf{v}) = [\mathcal{H}^{-\frac{1}{2}}\mathbf{v}]^T \mathbf{B}^{-1} [\mathcal{H}^{-\frac{1}{2}}\mathbf{v}] + \frac{1}{2}[\mathbf{d}^o - \mathbf{G}\mathcal{H}^{-\frac{1}{2}}\mathbf{v}]^T \mathbf{R}^{-1} [\mathbf{d}^o - \mathbf{G}\mathcal{H}^{-\frac{1}{2}}\mathbf{v}]$$

and the gradient:

$$\nabla_{\mathbf{v}} J = \mathcal{H}^{-\frac{T}{2}} \mathbf{B}^{-1} \mathcal{H}^{-\frac{1}{2}} \mathbf{v} - \mathcal{H}^{-\frac{T}{2}} \mathbf{G}^T \mathbf{R}^{-1} [\mathbf{d}^o - \mathbf{G}\mathcal{H}^{-\frac{1}{2}}\mathbf{v}]$$

and finally the hessian:

$$\mathcal{H}_{\mathbf{v}} = \mathcal{H}^{-\frac{T}{2}} \mathbf{B}^{-1} \mathcal{H}^{-\frac{1}{2}} + \mathcal{H}^{-\frac{T}{2}} \mathbf{G}^T \mathbf{R}^{-1} \mathbf{G} \mathcal{H}^{-\frac{1}{2}} = \mathcal{H}^{-\frac{T}{2}} \mathcal{H} \mathcal{H}^{-\frac{1}{2}} = \mathbf{I}$$

Meaning that the conditioning of the minimization of $J_{\mathbf{v}}$ is 1, which is as the best conditioning as you can get

However the $\mathbf{G}^T\mathbf{R}^{-1}\mathbf{G}$ term is difficult to evaluate and to handle. In practice, most of the time the actual change of variable is:

$$\mathbf{v} = \mathbf{B}^{-\frac{1}{2}}\delta\mathbf{x} \tag{1}$$

where the new control variable $\mathbf{v}$ is without dimension.
In practice, only the inverse of the change of variable is required ($\delta\mathbf{x}_0$ is required in order to integrate the model) and the gradient of $J_{\delta\mathbf{x}_o}$ is easily retrieved by:

$$\nabla J_{\delta\mathbf{v}} = \mathbf{B}^{T/2}\nabla J_{\delta\mathbf{x}_0} \tag{2}$$

With this change of variable, the hessian becomes:

$$\mathcal{H}_{\mathbf{v}} = \mathbf{B}^{\frac{T}{2}} \mathcal{H}_{\delta \mathbf{x}} \mathbf{B}^{\frac{1}{2}} = Id + \mathbf{B}^{\frac{T}{2}} \mathbf{G}^{T} \mathbf{R}^{-1} \mathbf{G} \mathbf{B}^{\frac{1}{2}}$$

with $\mathbf{B}^{\frac{T}{2}} \mathbf{G}^{T} \mathbf{R}^{-1} \mathbf{G} \mathbf{B}^{\frac{1}{2}}$ symmetric definite positive.
This preconditioning ensure that the smallest eigenvalue of $\mathcal{H}$ is larger than 1 and therefore its conditioning is bounded Moreover the size of the control vector is generally far larger than the number of observation therefore the hessian of the preconditioned problem has a majority of 1

Remark 1:

If we are not in the case of a non dimensional problem, for the minimization to be effective, one needs to account for the problems of the possible different order of magnitude of the different variables. Such preconditioning as presented above have the advantage to make the control vector without dimension.

Remark 2:

The Hessian of the variational data assimilation cost function is equal to the inverse of the analysis error covariance matrix

$$\mathcal{H}^{-1} = \mathbf{P}^a = [\mathbf{B}^{-1} + \mathbf{G}^T\mathbf{R}^{-1}\mathbf{G}]^{-1} \tag{3}$$

Outline

The role of B

- The dependence of the optimal state is significantly depending on **B** for 3D-Var is quite strong since

$$\mathbf{x}^a = \mathbf{x}^b + \mathbf{B}\mathbf{H}^T(\mathbf{H}\mathbf{B}\mathbf{H}^T + \mathbf{R})^{-1}(\mathbf{y}^o - H(\mathbf{x}^b))$$

  In the same way, for the 4D-Var we get:

$$\mathbf{x}^a = \mathbf{x}^b + \mathbf{B}\mathbf{M}^T\mathbf{H}^T(\mathbf{H}\mathbf{M}\mathbf{B}\mathbf{M}^T\mathbf{H}^T + \mathbf{R})^{-1}(\mathbf{y}^o - H(\mathbf{x}^b))$$

Lets assume that we get one observation in one grid point corresponding to the k^{th} element of the state vector. The observation operator is then very simple:

$$H = \begin{pmatrix} 0 & \dots & 0 & 1 & 0 & \dots & 0 \end{pmatrix}$$

The k^{th} column only contains 1, the others are set to 0.

- 3DVAR: The increment is proportional to a column of **B**! The choice of **B** is then crucial, it will describe how this observation will influence what is happening to the neighbouring points and for the other variables.
- 4DVAR:The increment at time i is proportional to a column of $\mathbf{M}\mathbf{B}\mathbf{M}^T$ which describe the background error covariances at time i. The **B** implicitly evolve with time in the 4D-Var algorithm.

Remarks

- Moreover, as presented before, the **B** matrix is often use as a preconditioner for the minimization, using the change of variable:

$$\mathbf{v} = \mathbf{B}^{-1/2}\delta\mathbf{x}_0$$

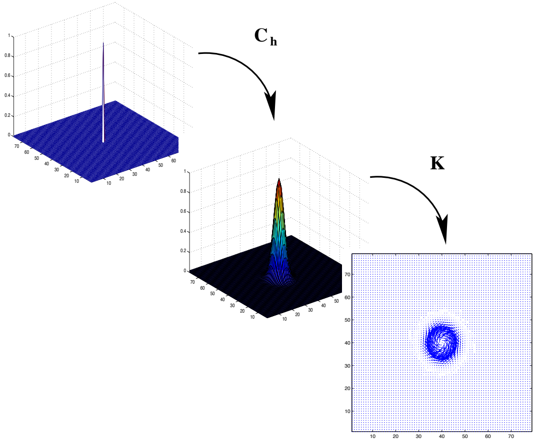
- If N is the number of the model variables multiplied by the number of grid point, then **B** is of size N^2. N being of the order of magnitude 10^7 à 10^10 for current geophysical applications. In practice **B** is never formed not stored, we rather code operators the take **v** as input and gives **Bv** or $\mathbf{B}^{1/2}\mathbf{x}$ as output.
- information spreading : in poorly observed areas, the form of the analysis increment is determined by the covariances structures.
- information smoothing : in densly observed areas, the smoothing will be governed by the correlation of **B**, assuring that the scales represented in the increment are compatible with the one of the model.
- balance properties : thanks to the off-diagonal blocks
- ill-conditionning of the problem : The data assimilation problems are under determined. The background term allows to introduce more information.
- flow dependent structure function

B as a sequence of operator

In order to get the full (or multivariate) error covariance matrix **B** one can write

$$\mathbf{B}_{multi} = \mathbf{K}\boldsymbol{\Sigma}\mathbf{C}\boldsymbol{\Sigma}\mathbf{K}^T \qquad (4)$$

where **K** is the *balance* operator
Σ the diagonal matrix of StdDev,
C the spatial correlation matrix



Outline

Faire face à l'évolution des moyens de calcul

- Its components $(\mathcal{M}, \mathcal{H}, \mathbf{B}, \ldots)$ can be parallel, but...
- Variational data assimilation is intrinsically sequential (iterative minimisation)
- In NWP, 4D-Var typically scales well up to an order 10^3s processors
- in the near future we need an order of magnitude more

Une piste : les méthodes variationnelles d'ensemble

Liu et al 2008

- Let $\epsilon_p = \mathbf{x}_p - \mathbf{x}_0$ denote an ensemble of $p = 1, \cdots, N_e$ state error realisations with respect to an unperturbed control member $\mathbf{x}_0$
- Let $\epsilon'_p = \epsilon_p - \bar{\epsilon}$ where $\bar{\epsilon} = \sum_{p=1}^{N_e} \epsilon_p / N_e$ and $\mathbf{B}_{sam} = \mathbf{X}'\mathbf{X}'^T$ where $\mathbf{X}' = \left(\epsilon'_p, \cdots, \epsilon'_{N_e}\right) / \sqrt{N_e - 1}$.
- the reduced order incremental 4Dvar problem would be (preconditioned by $\delta\mathbf{x}^{(k)} = \mathbf{X}'^{(k)}\mathbf{w}^{(k)}$)

$$\min_{\mathbf{w}^{(k)} \in \mathbb{R}^p} J(\mathbf{w}^{(k)}) = \mathbf{w}^{(k)T}\mathbf{w}^{(k)} + \sum_{i \in obs} (\mathbf{H}_i^{(k)}\mathbf{M}_{0,t_i}^{(k)}\mathbf{X}'^{(k)}\mathbf{w}^{(k)} - \mathbf{y}_i^{(k-1)})^T\mathbf{R}_i^{-1}(\mathbf{H}_i^{(k)}\mathbf{M}_{0,t_i}^{(k)}\mathbf{X}'^{(k)}\mathbf{w}^{(k)} - \mathbf{y}_i^{(k-1)})$$

- whose gradient is

$$\nabla J = \mathbf{w}^{(k)} + \sum_{i \in obs} (\mathbf{H}_i^{(k)}\mathbf{M}_{0,t_i}^{(k)}\mathbf{X}'^{(k)})^T\mathbf{R}_i^{-1}(\mathbf{H}_i^{(k)}\mathbf{M}_{0,t_i}^{(k)}\mathbf{X}'^{(k)}\mathbf{w}^{(k)} - \mathbf{y}_i^{(k-1)})$$

Une piste : les méthodes variationnelles d'ensemble

Liu et al 2008

- the idea here is to form the matrices $(\mathbf{H}_i^{(k)}\mathbf{M}_{0,t_i}^{(k)}\mathbf{X}'^{(k)})$, using the ensemble, and transpose it directly

$$(\mathbf{H}_i^{(k)}\mathbf{M}_{0,t_i}^{(k)}\mathbf{X}'^{(k)}) \approx \frac{1}{\sqrt{N_e - 1}} \Big(\mathcal{H}_i(\mathcal{M}_{0,t_i}(\mathbf{x}_1^{b(k-1)})) - \mathcal{H}_i(\mathcal{M}_{0,t_i}(\overline{\mathbf{x}^{b(k-1)}})), \\ , \cdots , \mathcal{H}_i(\mathcal{M}_{0,t_i}(\mathbf{x}_p^{b(k-1)})) - \mathcal{H}_i(\mathcal{M}_{0,t_i}(\overline{\mathbf{x}^{b(k-1)}}))\Big)$$

- Pros :
  - At last, we are rid of the adjoint !
  - the inner loop is far cheaper
  - the expensive part (outer loop) does scale well in parallel
  - same as for any ensemble methods
- Cons:
  - same as for any ensemble methods (localization, inflation, cost increases with number of obs)
  - convergence properties ? (toward the 4Dvar minimum)
  - approximate TL/AD
  - For highly non linear systems, the number of required outer loops will increase

Une autre piste : parallélisme en temps

Idée de départ Weak constraint 4D-Var

$$\begin{cases} \dfrac{d\mathbf{x}}{dt} & = & \mathcal{M}(\mathbf{x}) + \eta(t), \quad t \in [0, T], \\ \mathbf{x}(0) & = & \mathbf{x}_0, \end{cases}$$

and the cost function to be minimised becomes

$$J(\mathbf{x}_0, \eta) = \frac{1}{2} \parallel \mathbf{x}_0 - \mathbf{x}^b \parallel^2_{\mathcal{X}} + \frac{1}{2}\int_0^T \parallel \mathbf{y}(t) - \mathcal{H}[\mathbf{x}(\mathbf{x}_0, \eta, t)] \parallel^2_{\mathcal{O}} dt + \frac{1}{2}\int_0^T \parallel \eta(t) \parallel^2_{\mathcal{E}} dt,$$

Early attempts: η constant or with prescribed evolution
(Griffith and Nichols 2000), (Vidard 2000, Vidard et al 2004)
For a modest additional cost, the fit to the data was slightly improve, but mainly :

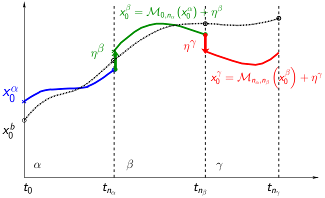
- it reduces the 'jump' between two subsequent assimilation windows
- it allows for longer assimilation windows

Une autre piste : parallélisme en temps

Weak constraint 4D-Var

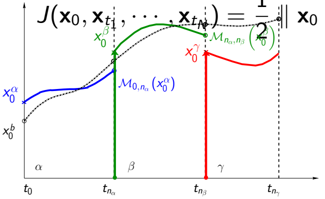
(Vidard 2001, Lemieux and Vidard 2012)

$$J(\mathbf{x}_0, \eta_{t_1}, \cdots, \eta_{t_N}) = \frac{1}{2} \parallel \mathbf{x}_0 - \mathbf{x}^b \parallel^2_{\mathcal{X}} + \frac{1}{2} \int_0^T \parallel \mathbf{y}(t) - \mathcal{H}[\mathbf{x}(t)] \parallel^2_{\mathcal{O}} dt + \frac{1}{2N} \sum_{i=1}^{N} \parallel \eta_{t_j} \parallel^2_{\mathcal{E}}$$



(Trémolet 2007):

$$J(\mathbf{x}_0, \mathbf{x}_{t_1}, \cdots, \mathbf{x}_{t_N}) = \frac{1}{2} \parallel \mathbf{x}_0 - \mathbf{x}^b \parallel^2_{\mathcal{X}} + \frac{1}{2} \int_0^T \parallel \mathbf{y}(t) - \mathcal{H}[\mathbf{x}(t)] \parallel^2_{\mathcal{O}} dt + \frac{1}{2N} \sum_{i=1}^{N} \parallel \mathcal{M}_{t_{j-1}\to t_j}(\mathbf{x}_{t_{j-1}}) - \mathbf{x}_{t_j} \parallel^2_{\mathcal{E}}$$

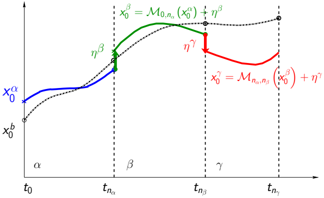


Une autre piste : parallélisme en temps

Weak constraint 4D-Var

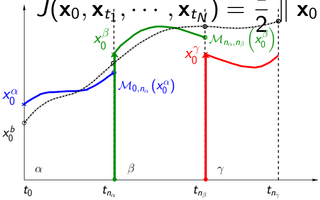
(Vidard 2001, Lemieux and Vidard 2012)

$$J(\mathbf{x}_0, \eta_{t_1}, \cdots, \eta_{t_N}) = \frac{1}{2} \parallel \mathbf{x}_0 - \mathbf{x}^b \parallel^2_{\mathcal{X}} + \frac{1}{2} \int_0^T \parallel \mathbf{y}(t) - \mathcal{H}[\mathbf{x}(t)] \parallel^2_{\mathcal{O}} dt + \frac{1}{2N} \sum_{i=1}^{N} \parallel \eta_{t_j} \parallel^2_{\mathcal{E}}$$



(Trémolet 2007):

$$J(\mathbf{x}_0, \mathbf{x}_{t_1}, \cdots, \mathbf{x}_{t_N}) = \frac{1}{2} \parallel \mathbf{x}_0 - \mathbf{x}^b \parallel^2_{\mathcal{X}} + \sum_{i=1}^{N} \frac{1}{2} \int_{t_{i-1}}^{t_i} \parallel \mathbf{y}(t) - \mathcal{H}[\mathbf{x}(t)] \parallel^2_{\mathcal{O}} dt + \frac{1}{2N} \sum_{i=1}^{N} \parallel \mathcal{M}_{t_{j-1} \to t_j}(\mathbf{x}_{t_{j-1}}) - \mathbf{x}_{t_j} \parallel^2_{\mathcal{E}}$$



Résolution de plus en plus grandes

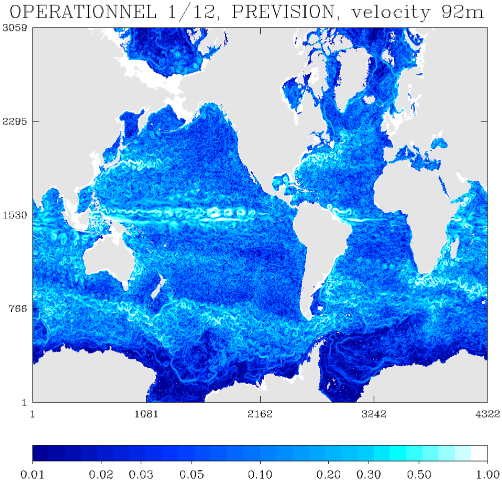
Actuelle

- Mercator operational model: NEMO 1/12°
- Number of gridpoints:

$$4322 \times 3059 \times 75 \sim 10^9$$

- 1 year of simulation costs 414 Gb memory, 90000 CPU hours, 1Tb storage (daily outputs)

courtesy MEOM team at LGGE (UGAS/CNRS)

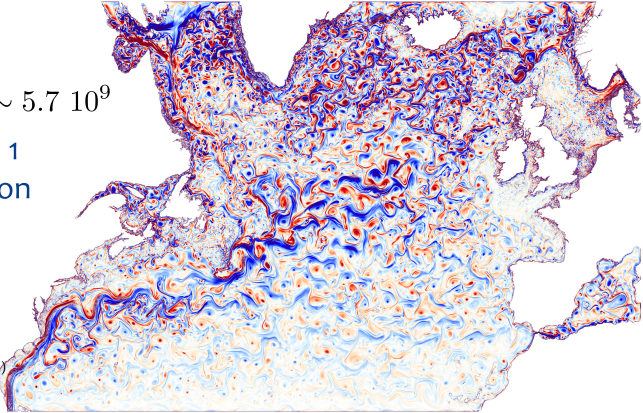


(In NWP operational models produce 13 millions fields daily (Totalling 8 TB/day)… target is ×40 in 2030)

Résolution de plus en plus grandes

Future

* NATL60
* Gridpoints: $5454 \times 3474 \times 300 \sim 5.7\ 10^9$
* 13000 processors, 1 month of simulation takes 1 day
* Full storage impossible



courtesy MEOM team at LGGE (UGAS/CNRS)

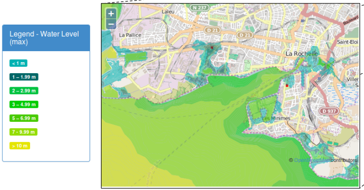
Why do we care?

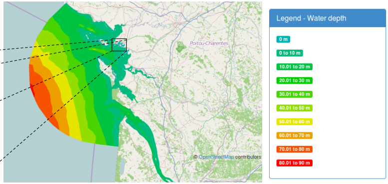
- The higher the resolution, the more expensive the model.
- 4DVar needs iterations, EnKF requires an ensemble and accurate error covariances.

$\longrightarrow$ Toward model reduction and multi resolution methods

Vers des systèmes complexes

Système de modélisation - plusieurs échelles





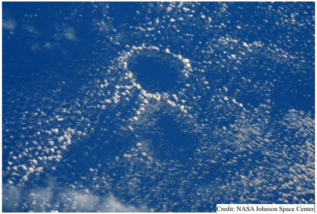
Couplage de modèles multi-dimensionels appliqués au littoral (M.-P. Daou et al., 2015)

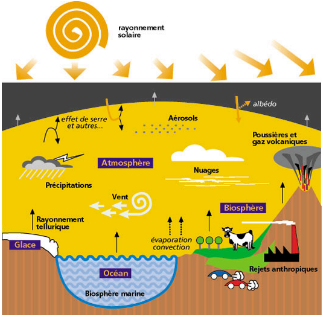
Vers des systèmes complexes

Système de modélisation - plusieurs milieux

The Earth system:

- Numerous different physical processes and media.
- Heterogeneous time and space scales.





- Different models and scientific cummunities
- Strong interactions between media