Projet SuNDAE : Point d'étape

Nicolas Baillot d'Etivaux - Postdoc GMAP/RECYF
Under the supervision of Yann Michel

In collaboration with :
Benjamin Ménétrier, Selime Gürol, Anthony Weaver

Content

- Multi-Incremental Multi-Resolution Data Assimilation scheme.
- Practical computation.
- Guess consistency and preconditionning equivalence.
- Building a toy model code.
- Preliminary results
- Future prospects

Full nonquadratic cost function

In data assimilation, one wants to minimize a non linear cost function which can be written as :

$$\mathcal{J}(\mathbf{x}) = \frac{1}{2}\left(\mathbf{x}-\mathbf{x}^b\right)^{\mathrm{T}} \mathbf{B}^{-1}\left(\mathbf{x}-\mathbf{x}^b\right) + \frac{1}{2}\left(\mathbf{y}^o-\mathcal{H}(\mathbf{x})\right)^{\mathrm{T}} \mathbf{R}^{-1}\left(\mathbf{y}^o-\mathcal{H}(\mathbf{x})\right) \quad (1)$$

where :

- $\mathbf{x} \in \mathbb{R}^n$ is the state in model space,
- $\mathbf{x}^b \in \mathbb{R}^n$ is the background state,
- $\mathbf{B} \in \mathbb{B}^{n\times n}$ is the background error covariance matrix,
- $\mathbf{y}^o \in \mathbb{R}^p$ is the observation vector,
- $\mathbf{R} \in \mathbb{R}^{p\times p}$ is the observation error covariance matrix,
- $\mathcal{H} : \mathbb{R}^n \to \mathbb{R}^p$ is the observation operator, nonlinear.

Linearization of the problem

- One can linearize the observation operator around a guess state $\mathbf{x}_k^g \in \mathbb{R}^n$ and define the increment $\delta\mathbf{x}_k$ at iteration k of the minimization by $\delta\mathbf{x}_k = \mathbf{x} - \mathbf{x}_k^g$, so that $\mathcal{H}(\mathbf{x}) \approx \mathcal{H}(\mathbf{x}_k^g) + \mathbf{H}_k \delta\mathbf{x}_k$,
  where $\mathbf{H}_k \in \mathbb{R}^{p\times m}$ is defined as : $\mathbf{H}_{k,ij} = \left.\frac{\partial \mathcal{H}_i}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_k^g}$.
- Instead of minimizing the full cost function $\mathcal{J}(\mathbf{x})$, we minimize successive quadratic approximations $J(\mathbf{x})$ around the guess $\mathbf{x}_k^g$:

$$\begin{aligned} J(\delta\mathbf{x}_k) = {} & \frac{1}{2}\left(\delta\mathbf{x}_k - \delta\mathbf{x}_k^b\right)^{\mathrm{T}} \mathbf{B}^{-1}\left(\delta\mathbf{x}_k - \delta\mathbf{x}_k^b\right) \\ & + \frac{1}{2}\left(\mathbf{d}_k - \mathbf{H}_k\delta\mathbf{x}_k\right)^{\mathrm{T}} \mathbf{R}^{-1}\left(\mathbf{d}_k - \mathbf{H}_k\delta\mathbf{x}_k\right) \end{aligned} \tag{2}$$

  where $\delta\mathbf{x}_k^b \in \mathbb{R}^n$ is the background increment : $\delta\mathbf{x}_k^b = \mathbf{x}^b - \mathbf{x}_k^g$ and $\mathbf{d}_k \in \mathbb{R}^p$ is the innovation vector : $\mathbf{d}_k = \mathbf{y}^o - \mathcal{H}(\mathbf{x}_k^g)$.

Linear system

Setting the gradient of $J(\delta \mathbf{x}_k)$ to zero gives the analysis increment $\delta \mathbf{x}_k^a$:

$$\begin{aligned} & \mathbf{B}^{-1}\left(\delta \mathbf{x}_k^a - \delta \mathbf{x}_k^b\right) - \mathbf{H}_k^{\mathrm{T}} \mathbf{R}^{-1}\left(\mathbf{d}_k - \mathbf{H}_k \delta \mathbf{x}_k^a\right) = 0 \\ \Leftrightarrow \; & \left(\mathbf{B}^{-1} + \mathbf{H}_k^{\mathrm{T}} \mathbf{R}^{-1} \mathbf{H}_k\right) \delta \mathbf{x}_k^a = \mathbf{B}^{-1} \delta \mathbf{x}_k^b + \mathbf{H}_k^{\mathrm{T}} \mathbf{R}^{-1} \mathbf{d}_k \\ \Leftrightarrow \; & \boxed{\mathbf{A}_k^{\mathbf{x}} \delta \mathbf{x}_k^a = \mathbf{b}_k^{\mathbf{x}}} \end{aligned} \tag{3}$$

with $\mathbf{A}_k^{\mathbf{x}} \in \mathbb{R}^{n \times n}$ and $\mathbf{b}_k^{\mathbf{x}} \in \mathbb{R}^n$ defined as :

$$\mathbf{A}_k^{\mathbf{x}} = \mathbf{B}^{-1} + \mathbf{H}_k^{\mathrm{T}} \mathbf{R}^{-1} \mathbf{H}_k \tag{4}$$

$$\mathbf{b}_k^{\mathbf{x}} = \mathbf{B}^{-1} \delta \mathbf{x}_k^b + \mathbf{H}_k^{\mathrm{T}} \mathbf{R}^{-1} \mathbf{d}_k \tag{5}$$

Linear system : preconditionning

The dimension of the **B** matrix being huge in practice, one can transform this linear system in order to lower its conditionning number by using preconditionners and the associated algorithms. We compare two types of preconditionning :

- The full **B** preconditionning, defining a new variable $\delta\overline{\mathbf{x}}_k = \mathbf{B}_k^{-1}\delta\mathbf{x}_k$ so that the linear system (3) is transformed into : $\boxed{\mathbf{A}_k^{\overline{\mathbf{x}}}\delta\overline{\mathbf{x}}_k^a = \mathbf{b}_k^{\overline{\mathbf{x}}}}$ with :
$$\mathbf{A}_k^{\overline{\mathbf{x}}} = \mathbf{I}_n + \mathbf{H}_k^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}_k\mathbf{B}_k, \text{ and } \mathbf{b}_k^{\overline{\mathbf{x}}} = \delta\overline{\mathbf{x}}_k^b + \mathbf{H}_k^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{d}_k.$$

- The square root $\mathbf{B}^{1/2}$ preconditionning : since **B** is positive definite, there is an infinity of square-roots $\mathbf{U} \in \mathbb{R}^{n\times m}$ with $m \geq n$ verifying $\mathbf{B} = \mathbf{U}\mathbf{U}^{\mathrm{T}}$, so that a new variable can be defined as $\delta\mathbf{v}_k = \mathbf{U}^{\mathrm{T}}\mathbf{B}^{-1}\delta\mathbf{x}_k$, transforming the linear system (3) into : $\boxed{\mathbf{A}_k^{\mathbf{v}}\delta\mathbf{v}_k^a = \mathbf{b}_k^{\mathbf{v}}}$ with :
$$\mathbf{A}_k^{\mathbf{v}} = \mathbf{I}_m + \mathbf{U}^{\mathrm{T}}\mathbf{H}_k^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}_k\mathbf{U}, \text{ and } \mathbf{b}_k^{\mathbf{v}} = \delta\mathbf{v}_k^b + \mathbf{U}^{\mathrm{T}}\mathbf{H}_k^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{d}_k$$

Linear system : preconditionning

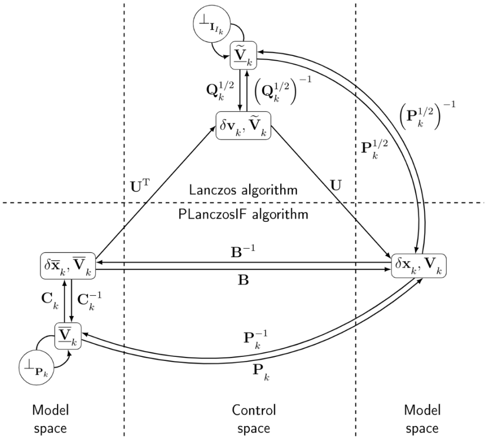
Once the system has been preconditionned with one of two above-mentionned preconditionners, the system can be solved by using different algorithms :

- For the full B preconditionning, the **PLanczosIF** method with a preconditioner $\mathbf{P}_k = \mathbf{BC}_k$, where $\mathbf{C}_k \in \mathbb{R}^{n\times n}$ and $\mathbf{P}_k \in \mathbb{R}^{n\times n}$ is detailed in ref selime.
- For the square root $\mathbf{B}^{1/2}$ preconditionning, the **Lanczos** method with a preconditioner $\mathbf{Q}_k = \mathbf{Q}_k^{1/2}\mathbf{Q}_k^{\mathrm{T}/2}$, where $\mathbf{Q}_k^{1/2} \in \mathbb{R}^{m\times m}$, is detailed in algorithm (ref selime).

A common (but not unique) way to define the preconditioners $\mathbf{P}_k = \mathbf{BC}_k$ or $\mathbf{Q}_k$ consists in using the spectral Limited Memory Preconditioner (LMP) approximated from the Ritz pairs.
We do not detail it here since we have left these LMPs for the moment in order to better understand the results, and we plan to take them into account in a near future.

Equivalence condition for preconditionners

Both approaches are equivalent if the preconditioners are linked via $\mathbf{P}_k^{1/2} = \mathbf{Q}_k^{1/2}\mathbf{U}$ which is verified for the spectral preconditioner approximated with the Ritz pairs.



$\perp_{\mathbf{I}_{I_k}}$
$\widetilde{\mathbf{V}}_k$
$\mathbf{Q}_k^{1/2}$
$\left(\mathbf{Q}_k^{1/2}\right)^{-1}$
$\delta\mathbf{v}_k, \widetilde{\mathbf{V}}_k$
$\left(\mathbf{P}_k^{1/2}\right)^{-1}$
$\mathbf{P}_k^{1/2}$
$\mathbf{U}^{\mathrm{T}}$
$\mathbf{U}$
Lanczos algorithm
PLanczosIF algorithm
$\mathbf{B}^{-1}$
$\delta\overline{\mathbf{x}}_k, \overline{\mathbf{V}}_k$
$\delta\mathbf{x}_k, \mathbf{V}_k$
$\mathbf{B}$
$\mathbf{C}_k$
$\mathbf{C}_k^{-1}$
$\overline{\mathbf{V}}_k$
$\perp_{\mathbf{P}_k}$
$\mathbf{P}_k^{-1}$
$\mathbf{P}_k$
Model space
Control space
Model space

FIGURE 1 – Summary of the relationship between all the introduced variables

Practical computation : Getting rid of $\mathbf{B}^{-1}$

In practice, the inverse of the background error covariance matrix $\mathbf{B}^{-1}$ is not available, even if it is needed in general to compute the right-hand sides $\mathbf{b}_k^{\bar{\mathbf{x}}}$ and $\mathbf{b}_k^{\mathbf{v}}$. However, it is very usual to define the guess as follows :

- for $k = 1$:

$$\delta \mathbf{x}_1^b = \mathbf{x}_1^g - \mathbf{x}^b = 0 \tag{6}$$

- for $k > 1$:

$$\begin{aligned}\delta \mathbf{x}_k^b &= \mathbf{x}^b - \mathbf{x}_k^g \\ &= \mathbf{x}^b - \mathbf{x}_{k-1}^a \\ &= \mathbf{x}^b - \left(\mathbf{x}_{k-1}^g + \delta \mathbf{x}_{k-1}^a\right) \\ &= \delta \mathbf{x}_{k-1}^b - \delta \mathbf{x}_{k-1}^a \end{aligned} \tag{7}$$

which can be combined recursively to yield :

$$\delta \mathbf{x}_k^b = -\sum_{i=1}^{k-1} \delta \mathbf{x}_i^a \tag{8}$$

Practical computation : Getting rid of $\mathbf{B}^{-1}$

With the full $\mathbf{B}$ preconditioning, $\mathbf{B}^{-1}$ can be applied on both side of equation (8) :

$$\boxed{\delta\bar{\mathbf{x}}_k^b = -\sum_{i=1}^{k-1} \delta\bar{\mathbf{x}}_i^a} \tag{9}$$

and with the square-root $\mathbf{B}$ preconditioning, $\mathbf{U}^{\mathrm{T}}\mathbf{B}^{-1}$ can be applied on both side of equation (8) :

$$\boxed{\delta\mathbf{v}_k^b = -\sum_{i=1}^{k-1} \delta\mathbf{v}_i^a} \tag{10}$$

Equations (9) and (10) can be used to compute $\mathbf{b}_k^{\bar{\mathbf{x}}}$ and $\mathbf{b}_k^{\mathbf{v}}$ respectively, without requiring $\mathbf{B}^{-1}$.

Practical computation : Updating **B**

The **B** matrix can be updated between outer iterations and may depend on iteration k. In this case, the previous trick to get rid of $\mathbf{B}^{-1}$ does not work systematically. Equation (8) is valid and yield :

- With the full **B** preconditioning : $\delta\overline{\mathbf{x}}_k^b = -\mathbf{B}_k^{-1}\sum_{i=1}^{k-1}\delta\mathbf{x}_i^a = -\sum_{i=1}^{k-1}\mathbf{B}_k^{-1}\mathbf{B}_i\delta\overline{\mathbf{x}}_i^a$

  If $\mathbf{B}_k^{-1}\mathbf{B}_i\delta\overline{\mathbf{x}}_i^a \neq \delta\overline{\mathbf{x}}_i^a$, equation (9) cannot be used consistently.

- With the square-root **B** preconditioning :
  $\delta\mathbf{v}_k^b = -\mathbf{U}_k^{\mathrm{T}}\mathbf{B}_k^{-1}\sum_{i=1}^{k-1}\delta\mathbf{x}_i^a = -\sum_{i=1}^{k-1}\mathbf{U}_k^{\mathrm{T}}\mathbf{B}_k^{-1}\mathbf{U}_i\delta\mathbf{v}_i^a$

  If $\mathbf{U}_k^{\mathrm{T}}\mathbf{B}_k^{-1}\mathbf{U}_i\delta\mathbf{v}_i^a \neq \delta\mathbf{v}_i^a$, equation (10) cannot be used consistently.

Changing the resolution : Interpolation

For computational efficiency, it is common to start the optimization at a lower resolution, and to increase it at each outer iteration k. We define two interpolators from resolution $\mathcal{R}_i$ at iteration i to resolution $\mathcal{R}_k$ at iteration k :

- $\mathbf{T}^{\mathbf{x}}_{i\to k} \in \mathbb{R}^{n_k \times n_i}$ in model space,
- $\mathbf{T}^{\mathbf{v}}_{i\to k} \in \mathbb{R}^{m_k \times m_i}$ in control space,

A special class of interpolators called "transitive interpolators" have three extra properties :

- Upscaling transitivity : for $n_i \leq n_j$ and $n_i \leq n_k$:

$$\mathbf{T}^{\mathbf{x}}_{j\to k}\mathbf{T}^{\mathbf{x}}_{i\to j} = \mathbf{T}^{\mathbf{x}}_{i\to k} \tag{11}$$

- Downscaling transitivity : for $n_i \leq n_j \leq n_k$:

$$\mathbf{T}^{\mathbf{x}}_{j\to i}\mathbf{T}^{\mathbf{x}}_{k\to j} = \mathbf{T}^{\mathbf{x}}_{k\to i} \tag{12}$$

- Right-inverse : for $n_i \leq n_k$

$$\mathbf{T}^{\mathbf{x}}_{k\to i}\mathbf{T}^{\mathbf{x}}_{i\to k} = \mathbf{I}_{n_i} \tag{13}$$

and similarly for $\mathbf{T}^{\mathbf{v}}_{i\to k}$ in control space.

Changing the resolution : Projective B matrix family

Associated to the various resolution, we qualify a **B** matrix family as "projective" if the low-resolution members can be defined as a projection of the high-resolution one, using transitive interpolators. For $n_i \leq n_k$, that would mean :

$$\mathbf{B}_k \mathbf{T}^{\mathbf{x}}_{i\to k} = \mathbf{T}^{\mathbf{x}}_{i\to k} \mathbf{B}_i \qquad (14)$$

and for the square-root of **B** :

$$\mathbf{U}_k \mathbf{T}^{\mathbf{v}}_{i\to k} = \mathbf{T}^{\mathbf{x}}_{i\to k} \mathbf{U}_i \qquad (15)$$

Changing the resolution : General requirements

The multi-resolution problem should be solved with the following requirements in mind :

- The background $\mathbf{x}^b$ is provided at full resolution, but it can be simplified at resolution $\mathcal{R}_k$:

$$\boxed{\mathbf{x}_k^b = \mathbf{T}^{\mathbf{x}}_{K\rightarrow k}\mathbf{x}^b} \tag{16}$$

- A full resolution guess denoted $\mathbf{x}_k^{g+}$ has to be computed at each outer iteration to run model trajectories used in the operators linearization. This full resolution guess can be simplified at resolution $\mathcal{R}_k$ to give the actual guess of the outer iteration k :

$$\boxed{\mathbf{x}_k^g = \mathbf{T}^{\mathbf{x}}_{K\rightarrow k}\mathbf{x}_k^{g+}} \tag{17}$$

- Only δ-quantities should be interpolated to higher resolution, and then possibly added to full quantities at high resolution.

Guess Consistency

In the linear systems solved at each outer iteration, the guess appears explicitely as the linearization state for nonlinear operators and to compute the innovation, and implicitely in the first term of the right-hand side.

For the first iteration, the full resolution guess is the background state, also provided at full resolution : $\mathbf{x}_1^{g+} = \mathbf{x}^b$.

At the end of iteration k, an analysis increment $\delta \mathbf{x}_k^a$ is produced at resolution $\mathcal{R}_k$. In the previous section, the guess of iteration k was defined as the analysis of iteration $k-1$, which is not possible anymore since the resolution increases.

Some interpolation of the output of the previous outer iteration is required to generate the analysis increment at full resolution $\delta \mathbf{x}_{k-1}^{a+}$, which updates the full resolution guess : $\mathbf{x}_k^{g+} = \mathbf{x}_{k-1}^{g+} + \delta \mathbf{x}_{k-1}^{a+}$.

Guess Consistency : Theoretical method

If we assume that the $\mathbf{B}^{-1}$ is available, the first term of the right-hand side can be obtained as :

$$\begin{aligned}\delta\overline{\mathbf{x}}_k^b &= \mathbf{B}_k^{-1}\delta\mathbf{x}_k^b \\ &= \mathbf{B}_k^{-1}\left(\mathbf{x}_k^b - \mathbf{x}_k^g\right)\end{aligned} \tag{18}$$

or

$$\begin{aligned}\delta\mathbf{v}_k^b &= \mathbf{U}_k^{\mathrm{T}}\mathbf{B}_k^{-1}\delta\mathbf{x}_k^b \\ &= \mathbf{U}_k^{\mathrm{T}}\mathbf{B}_k^{-1}\left(\mathbf{x}_k^b - \mathbf{x}_k^g\right)\end{aligned} \tag{19}$$

Here, both explicit and implicit guesses are consistent, thanks to the use of $\mathbf{B}^{-1}$.

We refer to this method as the **theoretical method**.

Guess consistency : Standard method

If $\mathbf{B}^{-1}$ is not available, the first term of the right-hand side is computed separately, using transformed versions of equations (9) and (10) with appropriate interpolations :

$$\boxed{\delta\underline{\overline{\mathbf{x}}}_k^b = -\sum_{i=1}^{k-1} \mathbf{T}_{i\to k}^{\mathbf{x}} \delta\overline{\mathbf{x}}_i^a} \tag{20}$$

and

$$\boxed{\delta\underline{\mathbf{v}}_k^b = -\sum_{i=1}^{k-1} \mathbf{T}_{i\to k}^{\mathbf{v}} \delta\mathbf{v}_i^a} \tag{21}$$

Underlines emphasize the fact that these quantities are not necessarily equal to their exact counterparts of equations (18) and (19).

We refer to this method as the **standard method**.

Guess consistency : Alternative method

Reversing the order of computations yields the same results as the consistent standard method, but with fewer and simpler computations. The first term of the right-hand side is computed first from equations (20) or (21). Then, the background increment is given by :

$$\delta \mathbf{x}_k^b = \mathbf{B}_k \delta \underline{\overline{\mathbf{x}}}_k^b \tag{22}$$

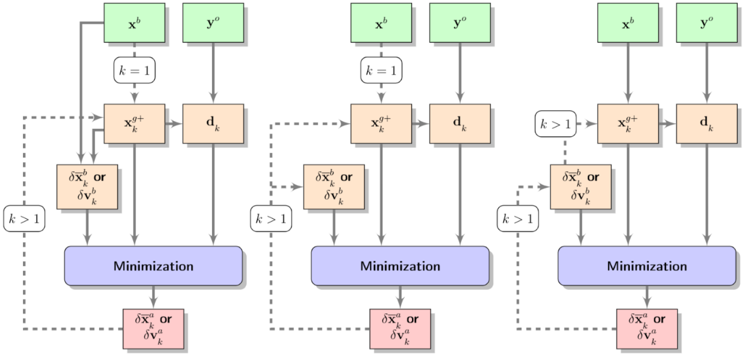
or

$$\delta \mathbf{x}_k^b = \mathbf{U}_k \delta \underline{\mathbf{v}}_k^b \tag{23}$$

Finally, the explicit guess is deduced at resolution $\mathcal{R}_k$ as $\mathbf{x}_k^g = \mathbf{x}_k^b - \delta \mathbf{x}_k^b$ and at full resolution as $\mathbf{x}_k^{g+} = \mathbf{x}^b - \mathbf{T}_{k \to K}^{\mathbf{x}} \delta \mathbf{x}_k^b$.

We refer to this method as the **alternative method**.

Summary on the methods



$\mathbf{x}^b$
$\mathbf{y}^o$
$k=1$
$k>1$
$\mathbf{x}_k^{g+}$
$\mathbf{d}_k$
$\delta\overline{\mathbf{x}}_k^b$ or $\delta\mathbf{v}_k^b$
Minimization
$\delta\overline{\mathbf{x}}_k^a$ or $\delta\mathbf{v}_k^a$

FIGURE 2 – Summary of the different methods : Theoretical (left) ; Standard (middle) ; Alternative (right).

Guess consistency

We are looking for sufficient conditions to make sure that both explicit and implicit guesses are consistent, i.e. $\delta\underline{\overline{\mathbf{x}}}_k^b = \delta\overline{\mathbf{x}}_k^b$ and $\delta\underline{\mathbf{v}}_k^b = \delta\mathbf{v}_k^b$.

One can write the guess as $\mathbf{x}_k^{g+} = \mathbf{x}^b + \sum_{i=1}^{k-1} \delta\mathbf{x}_i^{a+}$ which can be introduced into equations (18) and (19) to get background increments as functions of the analysis increment at full resolution $\delta\mathbf{x}_i^{a+}$, leading to :

$$\delta\overline{\mathbf{x}}_k^b = -\mathbf{B}_k^{-1}\mathbf{T}_{K\to k}^{\mathbf{x}} \sum_{i=1}^{k-1} \delta\mathbf{x}_i^{a+} \tag{24}$$

and

$$\delta\mathbf{v}_k^b = -\mathbf{U}_k^{\mathrm{T}}\mathbf{B}_k^{-1}\mathbf{T}_{K\to k}^{\mathbf{x}} \sum_{i=1}^{k-1} \delta\mathbf{x}_i^{a+} \tag{25}$$

Guess consistency

It is mathematically possible to show that $\delta \underline{\overline{\mathbf{x}}}_k^b = \delta \overline{\mathbf{x}}_k^b$ if :

$$\mathbf{T}^{\mathbf{x}}_{K \to k} \delta \mathbf{x}^{a+}_{k-1} = \mathbf{B}_k \sum_{i=1}^{k-1} \mathbf{T}^{\mathbf{x}}_{i \to k} \delta \overline{\mathbf{x}}^a_i - \mathbf{T}^{\mathbf{x}}_{K \to k} \sum_{i=1}^{k-2} \delta \mathbf{x}^{a+}_i \tag{26}$$

And similarly in control space $\delta \underline{\mathbf{v}}_k^b = \delta \mathbf{v}_k^b$ if and only if :

$$\mathbf{T}^{\mathbf{x}}_{K \to k} \delta \mathbf{x}^{a+}_{k-1} = \mathbf{U}_k \sum_{i=1}^{k-1} \mathbf{T}^{\mathbf{v}}_{i \to k} \delta \mathbf{v}^a_i - \mathbf{T}^{\mathbf{x}}_{K \to k} \sum_{i=1}^{k-2} \delta \mathbf{x}^{a+}_i \tag{27}$$

Conditions (26) and (27) on $\delta \mathbf{x}^{a+}_{k-1}$ are based on its interpolation at full resolution, and do not provide an explicit expression.

One can show that these conditions are verified in the case of transitive interpolators and projective **B** matrix.

Preconditioning equivalence

When the resolution changes between outer iterations, the equivalence between full **B** and square-root **B** preconditionings is not guaranteed anymore. Two kinds of conditions can be required to get similar results with both preconditionings :

1. $\delta \mathbf{x}_i^{a+}$ is computed in a similar way for both preconditionings, whatever the interpolation space and the order of interpolations and **B** matrix (or its square-root) applications.
2. Equations (20) and (21) can be related by :

$$\mathbf{U}_k^{\mathrm{T}} \delta \underline{\overline{\mathbf{x}}}_k^b = \delta \underline{\mathbf{v}}_k^b \Leftrightarrow \; -\mathbf{U}_k^{\mathrm{T}} \sum_{i=1}^{k-1} \mathbf{T}_{i\rightarrow k}^{\mathbf{x}} \delta \overline{\mathbf{x}}_i^a = -\sum_{i=1}^{k-1} \mathbf{T}_{i\rightarrow k}^{\mathbf{v}} \delta \mathbf{v}_i^a \tag{28}$$

If the previous outer iterations were equivalent for both preconditionings, then $\delta \mathbf{v}_i^a = \mathbf{U}_i^{\mathrm{T}} \delta \overline{\mathbf{x}}_i^a$ so this condition becomes $\mathbf{U}_k^{\mathrm{T}} \mathbf{T}_{i\rightarrow k}^{\mathbf{x}} = \mathbf{T}_{i\rightarrow k}^{\mathbf{v}} \mathbf{U}_i^{\mathrm{T}}$ which is the condition for a projective **B** matrix family.

Preconditioning equivalence

Depending on the method, the equivalence conditions may vary :

- **Theoretical method** : condition 1. is sufficient since the first term of the right-hand side is deduced from the full resolution guess.
- **Standard method** : both conditions 1. and 2. are required since the full resolution guess and the first term of the right-hand side are computed independently.
- **Alternative method** : condition 2. is sufficient, since the full resolution guess is deduced from the first term of the right-hand side.

Building a toy model code

We have built a 2D toy model in order to monitor the differences between the three mentionned strategies and the two types of preconditionning according to the choice of the **B** matrix family and the interpolation method used to compute the guess between two outer loops.
We briefly describe the fortran code in the following slides :

Toy model code : Initialisation of the problem

- Generation of a spectral **B** matrix (Background error covariance matrix)
- Generation of a gaussian random background state $\mathbf{x^b}$ and a gaussian random truth state $\mathbf{x^t}$ at full resolution (by applying $\mathbf{B}^{1/2}$ on random fields with normal distributions).
- Generation of a diagonal **R** matrix (Observation error covariance matrix) : $\mathbf{R}_{ij} = (\sigma^o)^2\delta_{ij}$, where σ^o is a scalar and δ_{ij} is the Kronecker symbol.
- Generation of an observation state from the truth, and its associated errors drawn from applying $\mathbf{R}^{1/2}$ to a random vector matching the observation points with a normal distribution.
- Generation of an observation operator $\mathcal{H}$ that maps the grid points to the observation points. We have choosen the following form in order to be able to tune the non-linearity induced by this operator : $\mathcal{H}\mathbf{x} = (1-\alpha)\mathbf{x} + \alpha\mathbf{x}^3$, where α is a scalar and $\mathbf{x}$ a is grid point field stored as a vector.

Toy model code : Solving the problem

In this toy problem, one can vary the values of the error attached to the observations σ^o ; the number of observations ; the correlation lenghtscale L_b, and the variance of the elements of the **B** matrix σ^b_{var} ; the number of inner and outer loops and their respective resolutions ; and the α parameter.

The minimization will be calculated in several ways according to :

- The method used to compute the guess between the outer loops (theoretical, standard, and alternative).
- The preconditionning method used in the inner loops and the associated algorithm (Full **B** or $\mathbf{B}^{1/2}$).
- The interpolators used for the change of resolution between the outer loops (nearest neighbor, bilinear or spectral). Note that only the spectral interpolator is transitive.
- The **B** matrix "family" (projective or not).
- The Low Memory Preconditionners used in the inner loops - (we have left this part for the moment and we plan to go back to it in a near future).

Toy model code : Tuning and Monitoring

Finally we can monitor various quantities in order to better understand the behaviour of this Multi-Incremental Multi-Resolution Data Assimilation scheme and the differences that appear between the different methods.

These quantities include all the guesses, the increments in the inner and outer loops, the cost functions (non-linear and linearized ; in observation space and in model space), the residues in the inner loops, the innovations and the **B** matrices.

Some preliminary results are shown in the following. We have arbitrarily used 2000 observations with a full resolution of 101x101 grid points, $\sigma^o = 0.01$, $L_b = 0.1$, $\sigma^b_{var} = 0$.

Results : Equivalence at full resolution

As expected, when no resolution change are maid between the outer loops, the three methods (theoretical, standard and alternative) are equivalent, whatever the choice of interpolators, preconditionning, or the **B** matrix family.

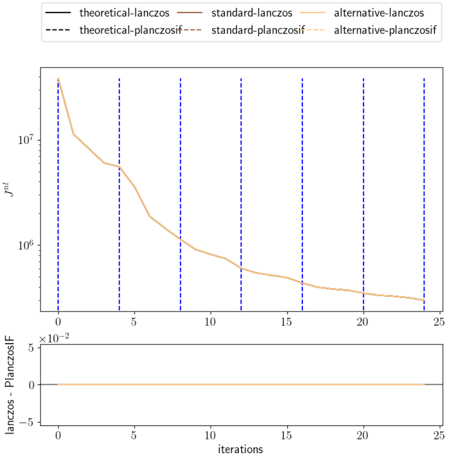


FIGURE 3 – **Top plot** : illustrative case of the non linear cost functions calculated with all the strategies with 6 outer loops of 4 inner loops, with a non linear observation operator ($\alpha = 0.1$) and using a spectral interpolation and projective $\mathcal{B}$ matrix. **Bottom plot** : differences between full **B** and square root **B** preconditionning.

Results : Effects of a non linear observation operator

When the problem is linearized at the begining of a new outer loop around a NEW guess state (obtained from the previous analysis), the linearized cost function in the new outer loop may start at a higher value than its last value in the previous outer loop, since the problem have been updated.

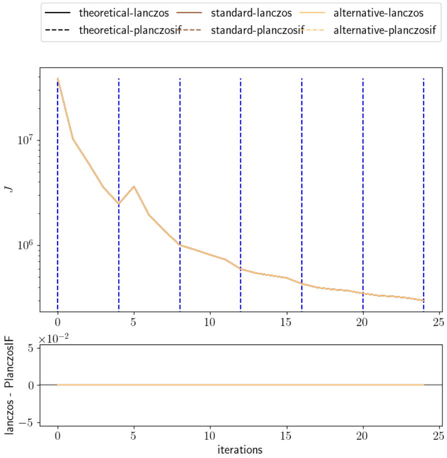


FIGURE 4 – **Top plot** : Corresponding linearized cost functions calculated with all the strategies with 6 outer loops of 4 inner loops, with a non linear observation operator ($\alpha = 0.1$) and using a spectral interpolation and projective **B** matrix. **Bottom plot** : differences between full **B** and square root **B** preconditionning.

Results : Changing the resolution

When the resolution changes between the outer loops but with a linear $\mathcal{H}$, J and J^{nl} are the same and may show the same feature as previously, but this time the "jump" is induced by the resolution changing and not by $\mathcal{H}$. We found that the "jump" occurs at the biggest resolution changing (other examples not shown).

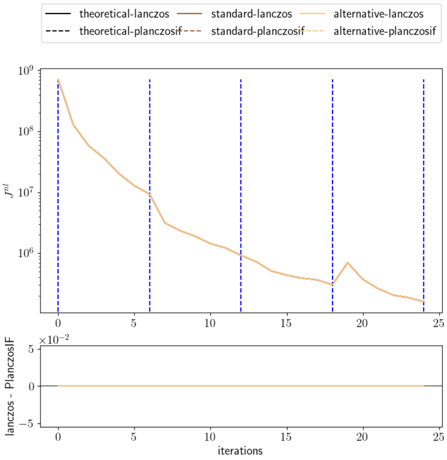


FIGURE 5 – Illustration with the same configuration as previously but with a linear $\mathcal{H}$ and the resolutions in the 4 outer loops being sccessively 11x11, 31x31, 51x51 and 101x101.

Equivalence of the methods

As expected, when changing the resolution and even using a non linear $\mathcal{H}$ operator, the three methods give the same results when the interpolation is transitive and the **B** matrix is projective.

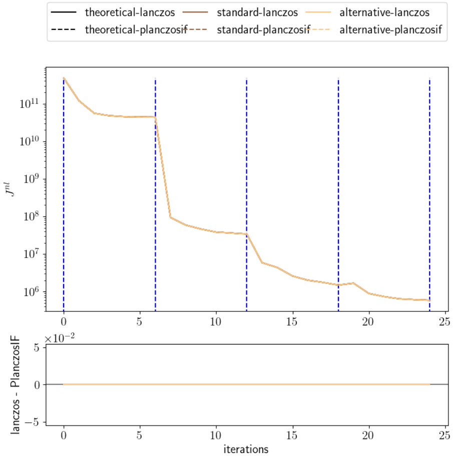


FIGURE 6 – Illustration of the non linear cost function with a non linear observation operator ($\alpha = 0.1$) and using a spectral interpolation and projective **B** matrix, the resolutions in the 4 outer loops being sccessively 11x11, 31x31, 51x51 and 101x101. **Bottom plot** : differences between full **B** and square root **B** preconditionning.

Breaking the condition on the transitive interpolators

If one uses non transitive interpolators, the three methods differ as shown below. The difference occurs after the first outer loop since the problem is the same in the first one. In this example, the alternative method diverge from the others (the theoretical one being supperposed with the standard).

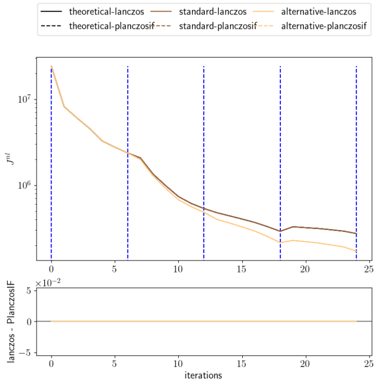


FIGURE 7 – same figure as figure 6 but using a bilinear interpolation between the outer loops

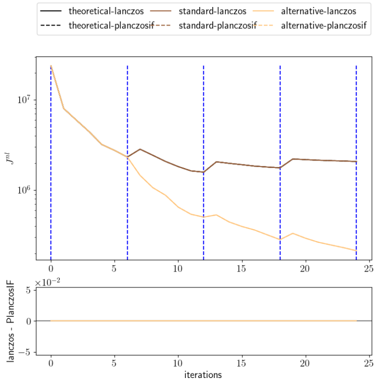


FIGURE 8 – same figure as figure 6 but using a nearest neighbor interpolation between the outer loops

Breaking the condition on projective B matrix

If one satisfy the transitive interpolation condition but violate the condition on the projective **B** matrix, the difference between the methods also appears but this time, the theoretical one differs from the others.

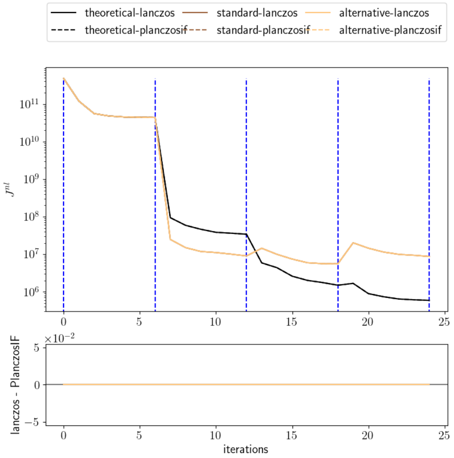


FIGURE 9 – Same figure as 6 (spectral interpolation) but using a non projective **B** matrix.

Breaking the condition on the transitive interpolators and B matrix family

Finally if none of the conditions are respected, obviously the three methodes differ (not really explicit in the present case for altrenative vs. standard), and only the theoretical method does not diverge.

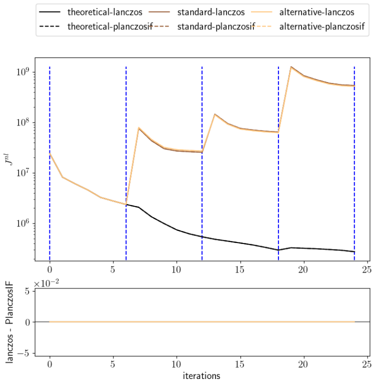


FIGURE 10 – same figure as but using a bilinear interpolation between the outer loops and a non projective **B** matrix

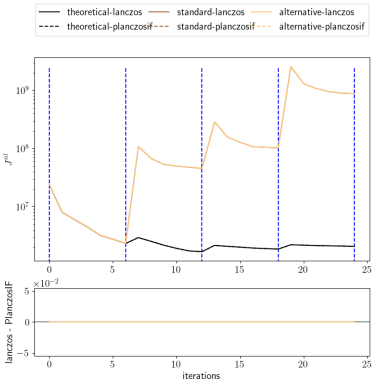


FIGURE 11 – same figure as but using a nearest neighbor interpolation between the outer loops and a non projective **B** matrix

Short conclusion

- As expected, if the resolution changes between the outer loops, we found some differences between the different methods only if at least one of the two conditions (on the transitive interpolators and **B** matrix family) is not satisfied.
- It seems that breaking the condition on the projective **B** matrix have the effect of making the theoretical method diverging from the others.
- Similarly, it seems that breaking the condition on the transitive interpolator makes the alternative method differ from the others.
- Finally of cours, breaking the two conditions at the same time makes the three methods diverging each other.
- The two preconditionning methods are shown equivalent in the presented examples BUT it might be due to a bug in the code that we need to check (work in progress).
- The choosen example may be too non-linear for a good illustration since some methods are diverging in the last plots.

Future prospects

- These results ar preleminar and need to be cross checked and discussed a little bit more.
- We have to find configurations of the code (tuning the parameters as for example the number of inner vs. outer loops, σ^o etc...) in which we are certain that there is no "side effects".
- A future prospect would be a MCMC-like sampling of the parameter space in order to study the dependance of the differences between the methods/preconditionnings as a function of those parameters (as for example the factor by which the resolution is changed betwwen the outer loops).
- Once our understanding of the behaviour of the Multi-Incremental Multi-Resolution will be better, we should try to implement it in a more realistic model.