Inverse Methods and Data Assimilation
Paper reading – [Lawless et al., 2008]

Hubert Leterme

January 16, 2019

1 Introduction

This is a summary of [Lawless et al., 2008], which introduces a low-order approximation of incremental 4D-Var procedure, which itself is a linearized version of the original 4D-Var algorithm.
The goal is to build a computationally affordable algorithm for data assimilation (e.g. for use in meteorology), while keeping track of the most important properties of the true model. This is not always the case in current implementations, which base their approximations on practical considerations, with a high risk of losing information.

2 Background on Incremental 4D-Var

2.1 Notations

Given $n \in \mathbb{N}$ (dimension of the problem) and $N \in \mathbb{N}$ (number of time steps), we consider the following dynamical system:

- For any $i \in \{0, \dots, N\}$, let's denote:
  - $\boldsymbol{x}_i \in \mathbb{R}^n$: estimated state vector at time t_i;
  - $\boldsymbol{x}_i^* \in \mathbb{R}^n$: (unknown) true state vector at time t_i;
  - $\mathcal{M}_i : \mathbb{R}^n \to \mathbb{R}^n$ $(i \leq N-1)$: nonlinear model operator, assumed perfect:
$$\begin{cases} \boldsymbol{x}_{i+1} = \mathcal{M}_i(\boldsymbol{x}_i) \\ \boldsymbol{x}_{i+1}^* = \mathcal{M}_i(\boldsymbol{x}_i^*) \end{cases} \tag{1}$$
  - $\boldsymbol{y}_i \in \mathbb{R}^{p_i}$: imperfect observation of $\boldsymbol{x}_i^*$;
  - $\mathcal{H}_i : \mathbb{R}^n \to \mathbb{R}^{p_i}$: operator mapping the state vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_i^*$ to the observation space;
  - $\boldsymbol{\eta}_i \in \mathbb{R}^{p_i}$: observation error, assumed unbiased and serially uncorrelated to the others:
$$\boldsymbol{\eta}_i = \boldsymbol{y}_i - \mathcal{H}_i(\boldsymbol{x}_i^*) \tag{2}$$
  - $\mathbf{R}_i \in \mathbb{R}^{p_i} \times \mathbb{R}^{p_i}$: covariance matrix of the observation error;
  - $\boldsymbol{d}_i \in \mathbb{R}^{p_i}$: innovation vector:
$$\boldsymbol{d}_i = \boldsymbol{y}_i - \mathcal{H}_i(\boldsymbol{x}_i) \tag{3}$$
- Let $\boldsymbol{x}^b \in \mathbb{R}^n$ denote a background estimate of $\boldsymbol{x}_0^*$ with error $\boldsymbol{\epsilon}^b$. We assume $\boldsymbol{\epsilon}^b$ to be the realization of a Gaussian vector with mean $\mathbf{0}$ (unbiased estimate) and covariance matrix $\mathbf{B}_0 \in \mathbb{R}^n \times \mathbb{R}^n$.
- Finally, let $\boldsymbol{x}^a$ denote the analysis, i.e. the value of $\boldsymbol{x}_0$ output by the 4D-Var algorithm.

Among the quantities defined above, $\boldsymbol{x}_i^*$, $\boldsymbol{\eta}_i$ and $\boldsymbol{\epsilon}^b$ are unknown; $\mathcal{M}_i$, $\mathcal{H}_i$, $\mathbf{R}_i$, $\boldsymbol{x}^b$ and $\mathbf{B}_0$ are fixed and estimated from our knowledge of the system and $\boldsymbol{y}_i$ are given as input. Then, $\boldsymbol{x}_i$ and $\boldsymbol{d}_i$ will be updated at each iteration of the 4D-Var algorithm, and the output is $\boldsymbol{x}^a$.

2.2 Linearization of the system

Let $\delta \boldsymbol{x}_0$ denote a small perturbation of the initial state $\boldsymbol{x}_0$. Then, for any $i \in \{0, \dots, N-1\}$, let $\delta \boldsymbol{x}_{i+1}$ denote the propagation of this initial perturbation through the model, i.e.:

$$\delta \boldsymbol{x}_{i+1} = \mathcal{M}_i(\boldsymbol{x}_i + \delta \boldsymbol{x}_i) - \boldsymbol{x}_{i+1} \tag{4}$$

For any $i \in \{0, \dots, N\}$, let also $\delta \boldsymbol{d}_i$ denote the innovation gap when applying this perturbation:

$$\delta \boldsymbol{d}_i = (\boldsymbol{y}_i - \mathcal{H}_i(\boldsymbol{x}_i + \delta \boldsymbol{x}_i)) - \boldsymbol{d}_i \tag{5}$$

Then, for each $i \in \{0, \dots, N\}$, the nonlinear equations (1) and (3) can be linearized around the state vector $\boldsymbol{x}_i$ by using Taylor expansions (see details in appendix):

$$\delta \boldsymbol{x}_{i+1} = \mathbf{M}_i \delta \boldsymbol{x}_i \tag{6}$$

$$\delta \boldsymbol{d}_i = \mathbf{H}_i \delta \boldsymbol{x}_i \tag{7}$$

where $\mathbf{M}_i$ and $\mathbf{H}_i$ denote the tangent linear operators (i.e. the Jacobian matrices) of $\mathcal{M}_i$ and $\mathcal{H}_i$ at $\boldsymbol{x}_i$.

2.3 Original versus Incremental 4D-Var algorithms

Both algorithm work under the assumption that the model is perfect. Thus the only value to infer is the initial state $\boldsymbol{x}_0$. The subsequent states will then be deduced by successively applying the model $\mathcal{M}_i$.
The original 4D-Var algorithm consists in minimizing a **non-quadratic** cost function which accounts for both the errors made with respect to the background and to the observations:

$$\begin{aligned} \mathcal{J}(\boldsymbol{x}_0) = &\frac{1}{2}(\boldsymbol{x}_0 - \boldsymbol{x}^b)^\top \mathbf{B}_0^{-1}(\boldsymbol{x}_0 - \boldsymbol{x}^b) \\ &+ \frac{1}{2}\sum_{i=0}^{N}(\mathcal{H}(\boldsymbol{x}_i) - \boldsymbol{y}_i)^\top \mathbf{R}_i^{-1}(\mathcal{H}_i(\boldsymbol{x}_i) - \boldsymbol{y}_i) \end{aligned} \tag{8}$$

where $\forall i \in \{1, \dots, N\}, \boldsymbol{x}_i = \mathcal{M}_{i-1}(\mathcal{M}_{i-2}(\dots \mathcal{M}_0(\boldsymbol{x}_0)))$.

The incremental 4D-Var algorithm actually performs several gradient descent procedures. At each iteration, an approximate quadratic cost function over $\delta \boldsymbol{x}_0$ is computed and minimized, and the initial value $\boldsymbol{x}_0$ is updated. Let's define this cost function, given a fixed $\boldsymbol{x}_0$ inferred at previous iteration:

$$\begin{aligned} \widetilde{\mathcal{J}}(\delta \boldsymbol{x}_0) &= \mathcal{J}(\boldsymbol{x}_0 + \delta \boldsymbol{x}_0) \text{ (by definition)} \\ &= \frac{1}{2}\left(\delta \boldsymbol{x}_0 - (\boldsymbol{x}^b - \boldsymbol{x}_0)\right)^\top \mathbf{B}_0^{-1}\left(\delta \boldsymbol{x}_0 - (\boldsymbol{x}^b - \boldsymbol{x}_0)\right) \\ &\quad + \frac{1}{2}\sum_{i=0}^{N}(\mathbf{H}_i \delta \boldsymbol{x}_i - \boldsymbol{d}_i)^\top \mathbf{R}_i^{-1}(\mathbf{H}_i \delta \boldsymbol{x}_i - \boldsymbol{d}_i) \end{aligned} \tag{9}$$

where $\forall i \in \{1, \dots, N\}, \delta \boldsymbol{x}_i = \mathbf{M}_{i-1}\mathbf{M}_{i-2} \dots \mathbf{M}_0 \, \delta \boldsymbol{x}_0$.

As we can see, all the non-linear operators have been removed from this cost function. This expression is easily obtained by developing $\mathcal{J}(\boldsymbol{x}_0 + \delta \boldsymbol{x}_0)$ from (8) and then using (5) and (7).
N.B.: This model works only if $\delta \boldsymbol{x}_0$ remains small. For that reason, the algorithm must be initialized to a state that isn't too far away from the true state.

For sake of comparison, both the original and incremental 4D-Var algorithms are detailed in table 1.

4D-Var	Incremental 4D-Var
1. Initialization: $\boldsymbol{x}^{(0)} = \boldsymbol{x}^b$	1. Initialization: $\boldsymbol{x}^{(0)} = \boldsymbol{x}^b$
2. Do:	2. Repeat for $k = 0 \dots K-1$:
(a) For each $i \in \{1, \dots, N\}$, compute $\boldsymbol{x}_i$ acc. to (1) and $\boldsymbol{d}_i$ acc. to (3).	(a) For each $i \in \{1, \dots, N\}$, compute $\boldsymbol{x}_i^{(k)}$ acc. to (1) and $\boldsymbol{d}_i^{(k)}$ acc. to (3).
(b) Use a gradient descent algorithm to minimize the cost function $\mathcal{J}$ with respect to $\boldsymbol{x}_0$. This may be computationally very expensive if the model is non-linear.	(b) Use a gradient descent algorithm to minimize the quadratic cost function $\widetilde{\mathcal{J}}^{(k)}$ with respect to $\delta\boldsymbol{x}_0^{(k)}$ ($\widetilde{\mathcal{J}}^{(k)}$ is defined for $\boldsymbol{x}_0 = \boldsymbol{x}_0^{(k)}$).
3. Output: $\boldsymbol{x}^a = \boldsymbol{x}_0$	(c) Update $\boldsymbol{x}_0^{(k+1)} = \boldsymbol{x}_0^{(k)} + \delta\boldsymbol{x}_0^{(k)}$.
	3. Output: $\boldsymbol{x}^a = \boldsymbol{x}_0^{(K)}$

Table 1: Original and Incremental 4D-Var algorithms

3 Using Model Reduction in this context

In practice this algorithm is still computationally too expensive. The article introduces model reduction methods applied to the 4D-Var algorithm.

3.1 Notations

Given the previously defined dynamical system and $r \in \mathbb{N}$ (dimension of the reduced space) such that $r \ll n$, we consider a reduced system where the n-dimensional inferred state vectors are still denoted $\boldsymbol{x}_0, \dots, \boldsymbol{x}_N$ and innovation vectors $\boldsymbol{d}_0, \dots, \boldsymbol{d}_N$. Moreover, we introduce the following quantities:

- For any $i \in \{0, \dots, N\}$, let's denote:
  - $\mathbf{U}_i \in \mathbb{R}^{n \times r}$: linear restriction operator
  - $\mathbf{V}_i \in \mathbb{R}^{n \times r}$: prolongation operator that maps from the lower-dimensional space $\mathbb{R}^r$ to the original space $\mathbb{R}^n$, defined such that:
  $$\mathbf{U}_i^\top \mathbf{V}_i = \mathbf{I}_r \tag{10}$$
  - $\delta\hat{\boldsymbol{x}}_i \in \mathbb{R}^r$: projection of $\delta\boldsymbol{x}_i$ into the reduced space such that:
  $$\delta\hat{\boldsymbol{x}}_i = \mathbf{U}_i^\top \delta\boldsymbol{x}_i \tag{11}$$
  - $\hat{\mathbf{M}}_i \in \mathbb{R}^{r \times r}$ and $\hat{\mathbf{H}}_i \in \mathbb{R}^{p_i \times r}$: simplified dynamic and observation operators defined by:
  $$\begin{cases} \hat{\mathbf{M}}_i = \mathbf{U}_i^\top \mathbf{M}_i \mathbf{V}_i \\ \hat{\mathbf{H}}_i = \mathbf{H}_i \mathbf{V}_i \end{cases} \tag{12}$$
  - $\delta\hat{\boldsymbol{d}}_i \in \mathbb{R}^{p_i}$: innovation gap obtained by using the reduced model, defined by:
  $$\delta\hat{\boldsymbol{d}}_i = (\boldsymbol{y}_i - \mathcal{H}_i(\boldsymbol{x}_i + \mathbf{V}_i \delta\hat{\boldsymbol{x}}_i)) - \boldsymbol{d}_i \tag{13}$$
  where $\mathbf{V}_i \delta\hat{\boldsymbol{x}}_i$ can be interpreted as an estimation of the state perturbation $\delta\boldsymbol{x}_i$, based on the lower-dimensional vector $\delta\hat{\boldsymbol{x}}_i$. To be compared with (5).
- $\hat{\mathbf{B}}_0 \in \mathbb{R}^{r \times r}$: background error covariance matrix in the reduced space.

3.2 Restricted linear system

Then we can write a restricted version of the linear system (6), (7) in $\mathbb{R}^r$:

$$\delta\hat{\boldsymbol{x}}_{i+1} = \hat{\mathbf{M}}_i \delta\hat{\boldsymbol{x}}_i \tag{14}$$

$$\delta\hat{\boldsymbol{d}}_i = \hat{\mathbf{H}}_i \delta\hat{\boldsymbol{x}}_i \tag{15}$$

The proof is given in appendix.

3.3 Restricted Incremental 4D-Var algorithm

At each iteration of the algorithm, the minimization is performed in the lower-dimensional space with respect to $\delta\hat{\boldsymbol{x}}_0$. The cost function is actually very similar to the one defined in (9). Given a fixed $\boldsymbol{x}_0$ inferred at previous iteration, it writes:

$$\begin{aligned}\hat{\mathcal{J}}(\delta\hat{\boldsymbol{x}}_0) = &\frac{1}{2}\left(\delta\hat{\boldsymbol{x}}_0 - \mathbf{U}_0^\top(\boldsymbol{x}^b - \boldsymbol{x}_0)\right)^\top \hat{\mathbf{B}}_0^{-1}\left(\delta\hat{\boldsymbol{x}}_0 - \mathbf{U}_0^\top(\boldsymbol{x}^b - \boldsymbol{x}_0)\right) \\ &+ \frac{1}{2}\sum_{i=0}^{N}(\hat{\mathbf{H}}_i\delta\hat{\boldsymbol{x}}_i - \boldsymbol{d}_i)^\top \mathbf{R}_i^{-1}(\hat{\mathbf{H}}_i\delta\hat{\boldsymbol{x}}_i - \boldsymbol{d}_i)\end{aligned} \tag{16}$$

For sake of comparison, both the full and restricted incremental 4D-Var algorithms are detailed in table 2 below.

Incremental 4D-Var	**Restricted Incremental 4D-Var**
1. Initialization: $\boldsymbol{x}^{(0)} = \boldsymbol{x}^b$	1. Initialization: $\boldsymbol{x}^{(0)} = \boldsymbol{x}^b$
2. Repeat for $k = 0 \dots K-1$:	2. Repeat for $k = 0 \dots K-1$:
(a) For each $i \in \{1, \dots, N\}$, compute $\boldsymbol{x}_i^{(k)}$ acc. to (1) and $\boldsymbol{d}_i^{(k)}$ acc. to (3).	(a) For each $i \in \{1, \dots, N\}$, compute $\boldsymbol{x}_i^{(k)}$ acc. to (1) and $\boldsymbol{d}_i^{(k)}$ acc. to (3).
(b) Use a gradient descent algorithm to minimize the *full* cost function $\widetilde{\mathcal{J}}^{(k)}$ with respect to $\delta\boldsymbol{x}_0^{(k)}$ ($\widetilde{\mathcal{J}}^{(k)}$ is defined for $\boldsymbol{x}_0 = \boldsymbol{x}_0^{(k)}$).	(b) Use a gradient descent algorithm to minimize the *restricted* cost function $\hat{\mathcal{J}}^{(k)}$ with respect to $\delta\hat{\boldsymbol{x}}_0^{(k)}$ ($\hat{\mathcal{J}}^{(k)}$ is defined for $\boldsymbol{x}_0 = \boldsymbol{x}_0^{(k)}$).
(c) Update $\boldsymbol{x}_0^{(k+1)} = \boldsymbol{x}_0^{(k)} + \delta\boldsymbol{x}_0^{(k)}$.	(c) Update $\boldsymbol{x}_0^{(k+1)} = \boldsymbol{x}_0^{(k)} + \mathbf{V}_0\delta\hat{\boldsymbol{x}}_0^{(k)}$.
3. Output: $\boldsymbol{x}^a = \boldsymbol{x}_0^{(K)}$	3. Output: $\boldsymbol{x}^a = \boldsymbol{x}_0^{(K)}$

Table 2: Full and Restricted Incremental 4D-Var algorithms

4 Appendix: proofs

Proof of equation (6)

Let $i \in \{0, \dots, N-1\}$. If $\delta\boldsymbol{x}_i$ is small enough, we have the following approximation (Taylor's theorem):

$$\begin{aligned}\mathcal{M}_i(\boldsymbol{x}_i + \delta\boldsymbol{x}_i) &\approx \mathcal{M}_i(\boldsymbol{x}_i) + \mathbf{M}_i\delta\boldsymbol{x}_i \\ &= \boldsymbol{x}_{i+1} + \mathbf{M}_i\delta\boldsymbol{x}_i\end{aligned}$$

where $\mathbf{M}_i$ denotes the Jacobian matrix of $\mathcal{M}_i$ at state $\boldsymbol{x}_i$.
Using (4), we get:

$$\delta \boldsymbol{x}_{i+1} = \mathbf{M}_i \delta \boldsymbol{x}_i$$

Proof of equation (7)

Let $i \in \{0, \ldots, N\}$. If $\delta \boldsymbol{x}_i$ is small enough, we have the following approximation (Taylor's theorem):

$$\begin{aligned} \mathcal{H}_i(\boldsymbol{x}_i + \delta \boldsymbol{x}_i) &\approx \mathcal{H}_i(\boldsymbol{x}_i) + \mathbf{H}_i \delta \boldsymbol{x}_i \\ &= (\boldsymbol{y}_i - \boldsymbol{d}_i) + \mathbf{H}_i \delta \boldsymbol{x}_i \end{aligned}$$

where $\mathbf{H}_i$ denotes the Jacobian matrix of $\mathcal{H}_i$ at state $\boldsymbol{x}_i$.
Using (5), we get:

$$\delta \boldsymbol{d}_i = \mathbf{H}_i \delta \boldsymbol{x}_i$$

N.B.: There appears to be a mistake in the article, which states that $\boldsymbol{d}_i = \mathbf{H}_i \delta \boldsymbol{x}_i$.

Proof of equation (14)

Proof of equation (15)

References

[Lawless et al., 2008] Lawless, A. S., Nichols, N. K., Boess, C., and Bunse-Gerstner, A. (2008). Using model reduction methods within incremental four-dimensional variational data assimilation. *Monthly Weather Review*, 136(4):1511–1522.