Data assimilation at local scale to improve CFD simulations of atmospheric dispersion: application to 1D shallow-water equations and method comparisons

Cécile Defforge, Bertrand Carissimo, Marc Bocquet, Patrick Armand, Raphaël Bresson

Data assimilation at local scale to improve CFD simulations of atmospheric dispersion: application to 1D shallow-water equations and method comparisons

Cécile L. Defforge*, Bertrand Carissimo, Marc Bocquet

CEREA, Joint laboratory École des Ponts ParisTech / EDF R&D, Université Paris-Est, Marne-la-Vallée, France
E-mail: cecile.defforge@enpc.fr
E-mail: bertrand.carissimo@enpc.fr
E-mail: marc.bocquet@enpc.fr
*Corresponding author

Patrick Armand

CEA, DAM, DIF, F-91297 Arpajon, France
E-mail: patrick.armand@cea.fr

Raphaël Bresson

CEREA, Joint laboratory École des Ponts ParisTech / EDF R&D, Université Paris-Est, Marne-la-Vallée, France
E-mail: raphael.bresson@enpc.fr

Abstract Atmospheric dispersion modelling requires meteorological inputs over local domains with possibly complex topographies. These local wind fields may be difficult to simulate with CFD models, in particular because of their sensitivity to geometrical features and to model inputs, especially the boundary conditions which are generally provided by larger-scale models or measurements. Using data assimilation, a few measurements inside the domain could add information to the imprecise boundary conditions and thus greatly enhance the precision of the dispersion simulations. Three data assimilation techniques (3DVar, the back and forth nudging algorithm, and the iterative ensemble Kalman smoother) have been adapted to local scale simulations by taking boundary conditions into account instead of initial conditions for which they are usually applied. Their performances have been evaluated at small scales, with a simple representation of the atmosphere into two layers, using 1D solution of the shallow water equations.

Keywords: Data assimilation, local scale simulation, boundary conditions, shallow water model, 3D-Var, back and forth nudging algorithm, iterative ensemble Kalman smoother.

Reference to this paper should be made as follows: Defforge, C.L., B. Carissimo, M. Bocquet, P. Armand, and R. Bresson (201X) 'Data assimilation at local scale to improve CFD simulations of atmospheric dispersion: application to 1D shallow-water equations and method comparisons', *International Journal of Environment and Pollution*, Vol. x, No. x, pp.xxx–xxx.

Biographical notes: Cécile Defforge is an Engineer in Environmental Sciences and has a Master's degree in Atmospheric and Oceanic Sciences. She is also a PhD student at CEREA on the subject of 'Data assimilation in CFD code for atmospheric simulations at local scale'.

Bertrand Carissimo is Senior Scientist at CEREA and EDF R&D.

Marc Bocquet is Senior Researcher at CEREA and Professor at ENPC.

Raphaël Bresson is Research Engineer at CEREA and EDF R&D.

Patrick Armand is Head of Laboratory and Senior Expert at CEA.

This paper is a revised and expanded version of a paper entitled "Data assimilation at local scale to improve CFD simulations of dispersion around industrial sites and urban neighbourhoods" presented at the 18th International conference on Harmonisation within Atmospheric Dispersion Modelling for Regulatory Purposes, Bologna, Italy, 9-12 October 2017.

1 Introduction

1.1 Context

Wind fields around industrial sites and in urban neighbourhoods might have very complex structures, which are sensitive to geometrical features such as topography and buildings. These wind field structures may be difficult to simulate accurately with CFD models, especially because of the sensitivity of these models to input data. Yet, these simulations are important to address various issues related to micrometeorology and dispersion of pollutants. To perform small scale simulations, CFD models – for instance the atmospheric version of *Code_Saturne* (Archambeau et al., 2004) – use inputs (initial and boundary conditions) that are meteorological data usually obtained from measurements or larger-scale model outputs. These data often lack precision, may not contain all necessary information, and are not adapted to the detailed features of local scale, especially the topography and the presence of buildings.

A few measurements inside the domain, although very local and potentially perturbed by, for instance, the buildings, have the potential to greatly enhance the precision of the simulations and thus the prediction of pollutants concentrations. Using measurements (e.g., wind velocity, concentration, etc.) to improve the estimate of the model state is the goal of data assimilation (DA).

Data assimilation techniques developed so far in meteorology (e.g., Kalnay, 2003; Asch et al., 2016) are generally applied to large scale simulations that are mainly driven by initial conditions (IC) and deal with simple geometries without obstacles. The smaller the domain is, the more sensitive the simulations are to boundary conditions (BC, e.g. Section 11.3 of Pielke, 2013), such that local scale simulations are dominantly influenced by boundary conditions (BC). Consequently, the present work aims at developing local-scale data assimilation techniques that focus on BC rather than IC and may deal with very complex geometries. To first test these DA methods, we use a simple representation of the

atmospheric boundary layer into two layers, using the shallow water equations which are an approximation of the Navier-Stokes equations for shallow flows.

1.2 Data assimilation methods

The goal of data assimilation is to improve the estimate of a model state using all available information: the *a priori* state of the system, the observations, the physical model, etc. This improvement may involve the optimisation of some control variables ($\mathbf{z}$) on which the state of the system depends. Data assimilation methods thus seek to correct the first guess of the variables state $\mathbf{z}^{\mathrm{b}}$, denoted background, using in particular the available observations.

One of the first and most basic data assimilation method is *nudging*, which consists in adding a relaxation term to the dynamical equations (e.g., Section 5.2.2 of Kalnay (2003), Chapter 4 of Asch et al. (2016)). The relaxation, or feedback term is proportional to the distance between the observations and the projection of the system state onto the observation space. An improvement of this method has been recently developed: the back and forth nudging algorithm (BFN) which consists of consecutive iterations of forward and backward integrations with nudging (Auroux & Blum, 2005, 2008; Auroux et al., 2013). It has been tested and proved to converge on simple models where the control variables are the initial conditions (Auroux, 2008; Auroux et al., 2011; Auroux & Nodet, 2012).

Apart from nudging, data assimilation techniques developed so far and operationally implemented are generally divided into two classes: statistical (or filtering methods) and variational methods. The variational methods such as 3D-Var (e.g. Kalnay, 2003; Asch et al., 2016) are based on the minimisation of a cost function that generally requires the use of the adjoint of the forward operator.

Recently, ensemble variational methods such as the iterative ensemble Kalman filter and smoother (IEnKF and IEnKS) (Sakov et al., 2012; Bocquet & Sakov, 2014) have been developed, combining the advantages of both variational and ensemble-based methods. The IEnKS is based on the iterative minimisation, using Gauss-Newton method, of a cost function defined in the ensemble space. In the linear model case, the filtering solution (i.e. at the end of the data assimilation time window) is equivalent to that obtained with the ensemble Kalman filter.

Both the BFN algorithm and the IEnKS have the great advantage of avoiding the use of the adjoint model and the tangent linear of the forward operator (see definitions in Section 2), and of handling non-linear analyses. Consequently, they have been adapted in the present study so as to consider boundary conditions as control variables.

1.3 Outline

In Section 2 we present the shallow water model used to represent the atmosphere and test the data assimilation methods in the present study. We acknowledge that the shallow water model used here is quite simple and that it is only a first step in the validation process of the methods, that should be tested on more complex cases. Then we present the three data assimilation methods studied here: 3D-Var, the back and forth nudging algorithm, and the IEnKS. In Section 3 we show the results obtained with the three methods with perfect observations. Afterwards, we compare the performances of the methods, in particular their sensitivities to observation errors and to background errors. In the last Section, we give some concluding remarks and perspectives for future work.

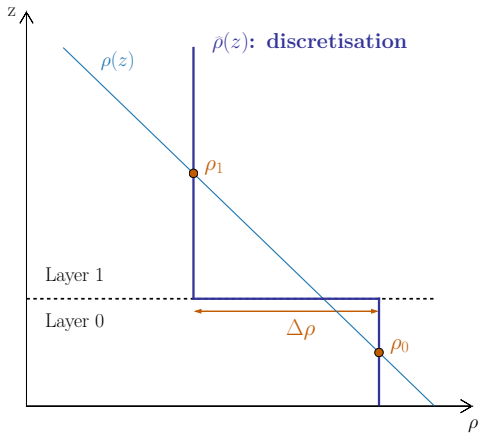


Figure 1: The layer system considered in the present study corresponds to a crude discretisation of the troposphere (light blue line) into two layers of constant density (dark blue line). The dynamics in the upper layer is neglected and the bottom layer follows the shallow water equations with a reduced gravity $g' = \frac{\Delta\rho}{\rho_0} g$.

2 Methods

2.1 Shallow layer model of the atmospheric boundary layer

The shallow water equations (SWE) are derived from the Navier-Stokes equations in the approximation of a small layer height, compared to horizontal spatial scales. These equations can be used to represent atmospheric flows if the gravity g is replaced by the reduced gravity $g' = \frac{\rho - \rho_\mathrm{a}}{\rho} g$ to account for the small difference of density between the simulated boundary layer (ρ) and the free atmosphere above (ρ_a), as shown on Figure 1. Moreover, the often used discretised 'level' models – which use, for instance, vertical finite-difference approximations – to simulate continuously stratified fluid, can be shown to be equivalent to multi-layer models – where SWE are applied to each layer (Audusse et al., 2006; Pedlosky, 2013). Therefore the simple model described below can be considered as a crude vertical representation of the atmosphere into two layers. This equivalence between 'level' models and 'layer' models, together with the relative simplicity of the SWE, motivate the use of a shallow layer model to first test the data assimilation methods for atmospheric local scale simulations, before applying them to CFD models.

In one dimension, the two state variables of the SWE are the thickness of the fluid layer (h, here the boundary layer), and the mean horizontal velocity (u) which are related as follows, if ground friction and diffusion are neglected:

$$\frac{\partial \mathbf{X}}{\partial t} + \mathbf{M}\frac{\partial \mathbf{X}}{\partial x} = \mathbf{S}, \tag{1}$$

where $\mathbf{X} = \begin{pmatrix} h \\ u \end{pmatrix}$, $\mathbf{M} = \begin{pmatrix} u & h \\ g' & u \end{pmatrix}$, and $\mathbf{S} = \begin{pmatrix} 0 \\ -g' \frac{\partial z_\mathrm{r}}{\partial x} \end{pmatrix}$ with z_r the bottom topography. The problem is well-posed if two boundary conditions are prescribed, one on h and one on u (e.g., Abbott, 1966). The regime of a flow is determined by the value of the Froude number,

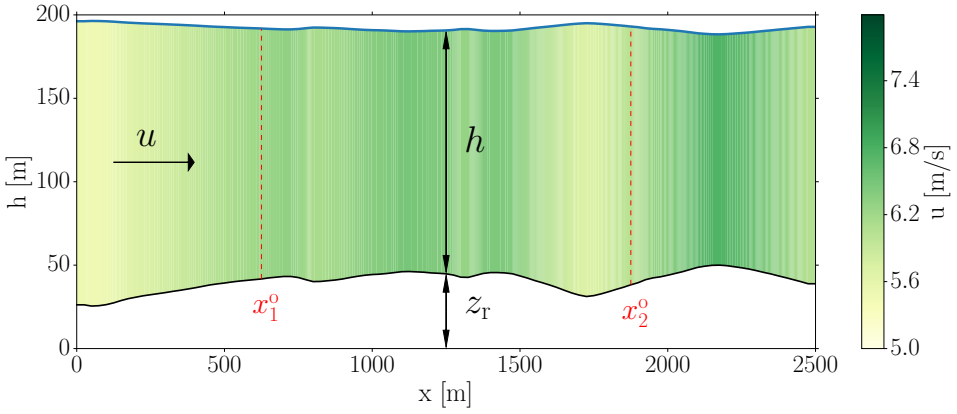


Figure 2: Results of the 1D shallow water equations over a realistic topography. The topography (z_r) is shown in black and the absolute height of the fluid ($z_\mathrm{r} + h$) is shown in blue. The fluid area is coloured by the mean vertical velocity (u). The vertical dashed, red lines show the locations of the two velocity observations assimilated in the following experiments.

which is the ratio of the flow inertia to the external forces: $Fr \equiv \frac{u}{\sqrt{gh}}$. If $Fr < 1$ the flow is subcritical, if $Fr = 1$ it is critical, and if $Fr > 1$ the flow is supercritical. In what follows, we consider flows that are subcritical, meaning that their velocities are smaller than the interface wave velocity. For such flows, the information is propagated downstream and also upstream. Consequently, it is necessary to specify a boundary condition on each side of the domain of size L (e.g., Alcrudo & Garcia-Navarro, 1993). We thus prescribe the velocity on the left of the domain: $u(x = 0) = u_\mathrm{L}$, and the fluid height on the right: $h(x = L) = h_\mathrm{R}$. These boundary conditions are here considered constant in time and we look for the steady state obtained with these boundary conditions.

The 1D shallow water equations with topography and without ground friction nor diffusion (Eq. 1) can be solved analytically by the Bernoulli equation (2) (e.g., Goutal & Maurel, 1997) which is based on the energy conservation and states that the sum of the specific charge ($\mathcal{H}_s \equiv h + \frac{u^2}{2g'}$) and the topographic elevation (z_r) is conserved over the domain:

$$\frac{u^2}{2g'} + h + z_\mathrm{r} = \mathcal{H}_0, \tag{2}$$

where $\mathcal{H}_0$ is the specific charge determined at one point of the domain where the system state is known.

The resolution of the Bernoulli equation is used to determine the steady state with two given boundary conditions, when no other forcing than the topography takes action. When nudging is added to the shallow water equations, the SWE are solved numerically by a finite-volume method.

Figure 2 shows the results of the 1D shallow water equations with the boundary conditions: $u_\mathrm{L} = 5.5\mathrm{m/s}$ and $h_\mathrm{R} = 154\mathrm{m}$. This figure illustrates the steady state that is considered as the reference for the data assimilation experiments.

2.2 3D-Var

The 3D-Var method seeks the optimal control vector $\mathbf{z}^{\mathrm{a}}$ that minimises a cost function $\mathcal{J}$ (Daley, 1991, e.g.). This cost function combines the distance between $\mathbf{z}$ and the first guess – or background – $\mathbf{z}^{\mathrm{b}}$, and the distance between the observations $\mathbf{y}$ and the projection of $\mathbf{z}$ in the observation space ($\widetilde{\mathcal{H}}(\mathbf{z})$). These distances are weighted by the inverse of the background error covariance matrix ($\mathbf{B}$) and the inverse of the observation error covariance matrix ($\mathbf{R}$) respectively. These inverses correspond to the confidence placed in each information (background and observations). The cost function thus reads:

$$\mathcal{J}(\mathbf{z}) = \frac{1}{2}\|\mathbf{z}-\mathbf{z}^{\mathrm{b}}\|^2_{\mathbf{B}^{-1}} + \frac{1}{2}\|\mathbf{y}-\widetilde{\mathcal{H}}(\mathbf{z})\|^2_{\mathbf{R}^{-1}}, \tag{3}$$

where $\|\mathbf{x}\|^2_{\mathbf{A}} = \mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x}$.

In what follows, we consider the one-dimensional case at local scale for which the control vector only includes the two constant boundary conditions:

$$\mathbf{z} = \begin{pmatrix} u_{\mathrm{L}} \\ h_{\mathrm{R}} \end{pmatrix}. \tag{4}$$

Here, the forward operator $\widetilde{\mathcal{H}}$ of the 3D-Var algorithm combines the shallow layer model $\mathcal{M}$, which gives the steady state of the system given $\mathbf{z}$, and an observation operator $\mathcal{H}$, which transforms the model variables into observed variables: $\widetilde{\mathcal{H}} = \mathcal{H} \circ \mathcal{M}$. Because the operator $\widetilde{\mathcal{H}}$ includes dynamical evolution ($\mathcal{M}$), the algorithm is similar to 4D-Var, though we keep the name 3D-Var since we consider steady states. In particular, the cost function considered here is similar to the one used in the IEnKS (Section 2.4). The minimisation of the cost function is performed using the L-BFGS-B non-linear, constrained optimisation algorithm (Byrd et al., 1995), through the ADAO module of the SALOME open-source platform (http://www.salome-platform.org/). The gradient is estimated using finite differences.

2.3 Back and forth nudging algorithm

The back and forth nudging algorithm (BFN) is an iterative algorithm: iterations of forward and backward integrations, both with nudging, are performed over a time period T during which observations are available (Fig. 3). The evolution of the system is governed by the two equations:

$$\begin{cases} \textbf{(F)} \quad \frac{\partial \mathbf{X}_k^{(\mathrm{f})}}{\partial t} + \mathbf{M}^{(\mathrm{f})}\frac{\partial \mathbf{X}_k^{(\mathrm{f})}}{\partial x} = \mathbf{S} + \mathbf{K}\left[\mathbf{y} - \mathcal{H}(\mathbf{X}_k^{(\mathrm{f})})\right] \quad \text{for } 0 \le t \le T,\ \delta t > 0, \\ \textbf{(B)} \quad \frac{\partial \mathbf{X}_k^{(\mathrm{b})}}{\partial t} + \mathbf{M}^{(\mathrm{b})}\frac{\partial \mathbf{X}_k^{(\mathrm{b})}}{\partial x} = \mathbf{S} - \widetilde{\mathbf{K}}\left[\mathbf{y} - \mathcal{H}(\mathbf{X}_k^{(\mathrm{b})})\right] \text{ for } T \ge t \ge 0,\ \delta t < 0, \end{cases} \tag{5}$$

where $\mathbf{K}$ and $\widetilde{\mathbf{K}}$ are gain matrices, the superscripts (f) and (b) refer to forward and backward variables, and the subscript k refers to the index of the BFN iteration (Auroux & Blum, 2005).

For the shallow water equations considered here, with the changes of variables $\widetilde{t} = T - t$, $\widetilde{\mathbf{X}} = \begin{pmatrix} h \\ \widetilde{u} \end{pmatrix}$, and $\widetilde{\mathbf{M}} = \begin{pmatrix} \widetilde{u}\, h \\ g\ \widetilde{u} \end{pmatrix}$, where $\widetilde{u} = -u$, the backward equation is exactly the same as the forward equation (1) where u is formally replaced with $\widetilde{u} = -u$.

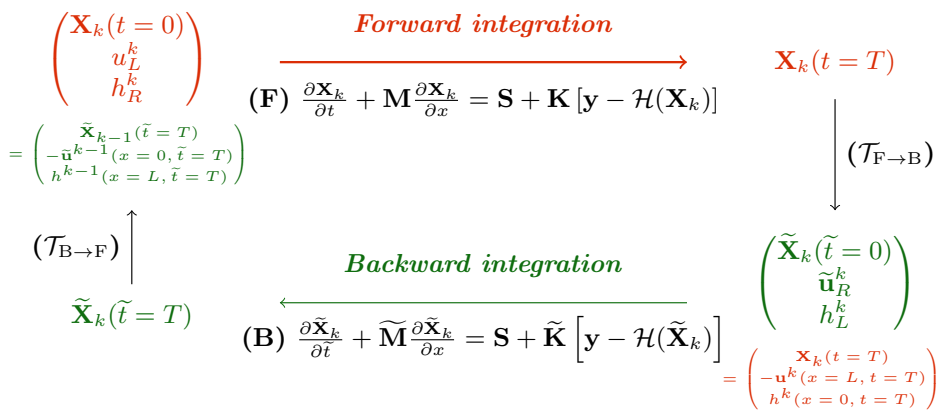


Figure 3: Diagram of the k^{th} iteration of the BFN algorithm. The transformations $(\mathcal{T}_{\mathrm{F}\to\mathrm{B}})$ and $(\mathcal{T}_{\mathrm{B}\to\mathrm{F}})$ shift from forward system to backward system and vice versa. They include a change of variable $\widetilde{t} = T - t$ and an inversion of boundary conditions. In the case of shallow water equations, the backward integration is equivalent to performing an integration with positive time steps but with $\widetilde{\mathbf{u}} = -\mathbf{u}$.

Applying the method of characteristics (Abbott, 1966) to the backward equation, it can be shown that the boundary conditions must be reversed between forward and backward integrations, and reciprocally (Fig. 3). The boundary conditions that are prescribed upstream in the forward integrations (here the velocity on the left of the domain, u_{L}) must be prescribed downstream in the backward integrations (here $\widetilde{u}_{\mathrm{R}}$ on the right). Similarly, the boundary conditions prescribed downstream in the forward integrations (here the height on the right, h_{R}) must be prescribed upstream in the backward integrations (here h_{L} on the left). The values of the boundary conditions $\widetilde{u}_{\mathrm{R}}$ and h_{L} for the k^{th} backward integration are given by the system state on the right and left of the domain, respectively, at the end of the k^{th} forward integration:

$$\widetilde{u}_{\mathrm{R}}^k = -u^k(x = L, t = T), \tag{6}$$

$$h_{\mathrm{L}}^k = h^k(x = 0, t = T). \tag{7}$$

Similarly, before the $(k+1)^{\text{th}}$ forward integration, the boundary conditions are reversed and updated according to the state of the system at the end of the k^{th} backward integration.

At each iteration of BFN, the control variables that are the boundary conditions (here u_{L} and h_{R}) are revised. After a sufficient number of BFN iterations, the system converges toward an analysis state, which depends in particular on the nudging matrices. It is important to note that the BFN algorithm does not solve the same problem as 3D-Var or the IEnKS. In fact, the BFN algorithm assimilates several times each observation in order to correct the system state. Except initially, it does not take into account any information on the background statistics of errors. In theory, the analysis state obtained at convergence is independent of the background state (Auroux & Blum, 2005), whereas 3D-Var and the IEnKS seek an optimal combination of the background and observations.

During each integration (either forward or backward), a conflict might appear between the prescribed external boundary conditions – i.e. the background – and the internal dynamic of the system, modified by the nudging. Moreover, the shallow layer model used here

does not have perfectly transparent boundaries, such that the waves created by the nudging are spuriously reflected. To avoid these two problems, the length of the period of model integration, T, has been chosen equal to the time needed for the nudging information to propagate throughout the domain. If the integration period is too short, the nudging information does not have time to propagate through the domain, and if it is too long, there is a mismatch between the prescribed boundary conditions (strong constraint) and the forcing produced by the nudging and spurious reflected waves might appear. The appropriate length of the integration period is estimated using the method of characteristics, and the iterations of forward and backward integrations are performed over this time period.

2.4 Iterative ensemble Kalman smoother

The iterative ensemble Kalman smoother (IEnKS, Bocquet & Sakov, 2014) is an ensemble variational method of data assimilation. As a variational method, it is based on the minimisation of a cost function. The outcome of the analysis process, $\mathbf{z}^{\mathrm{a}}$, might differ from the (unknown) truth, $\mathbf{z}^{\mathrm{t}}$, by the analysis error: $\epsilon^{\mathrm{a}} = \mathbf{z}^{\mathrm{a}} - \mathbf{z}^{\mathrm{t}}$. As the IEnKS is an ensemble-based method, the analysis error space is spanned by a limited number of vectors: the ensemble members. In fact, the analysis error covariance matrix, defined as as $\mathbf{P}^{\mathrm{a}} = \mathbb{E}\left[\epsilon^{\mathrm{a}} \left(\epsilon^{\mathrm{a}}\right)^{\mathrm{T}}\right]$ where $\mathbb{E}$ represents the expectation operator, can be directly calculated from the ensemble. The ensemble is also used in the minimisation of the cost function, to avoid the use of the adjoint of the model and the observation operator.

Similar to the 3D-Var algorithm, the cost function of the IEnKS measures the distance to the background and to the observations over a specific time-range, referred to as assimilation window. The IEnKS considers assimilation time windows of size $L\Delta t$ with $L \geq 0$, and might assimilate asynchronous observations available over this time window. The iterative ensemble Kalman filter (IEnKF, Sakov et al., 2012) is the particular case where $L = 1$. If $L > 1$, the analysis is performed at a time t_{L} and optimises the state of the control vector at any time between the beginning of the assimilation time window (t_0) and t_{L-1}. Here, we are interested in the steady state obtained with some given constant boundary conditions. Consequently we consider a long data assimilation window – long enough to reach the steady state – and the observations are only available at the end of this time window. Thus, we only perform one analysis cycle of the IEnKS.

In the original derivation of the IEnKS and IEnKF (Bocquet & Sakov, 2014; Sakov et al., 2012), the control vectors include the initial conditions but not the boundary conditions. Here we are interested in local scale simulations, which are mainly driven by boundary conditions. Hence, in what follows we perform a similar derivation, albeit including only the boundary conditions in the control vector. Similar to 3D-Var, we consider here the simple case of constant boundary conditions, such that the control vector is reduced to $\mathbf{z} = \begin{pmatrix} u_{\mathrm{L}} \\ h_{\mathrm{R}} \end{pmatrix}$. However, the method presented here can be generalised to more complex cases where the control vector includes both the initial and the boundary conditions, and the latest might vary in time.

We follow a derivation similar to the one in Bocquet & Sakov (2014) where we consider a perfect model which relates the steady state of the system ($\mathbf{X}_{\mathrm{s}}$) to the boundary conditions ($\mathbf{z}$):

$$\mathbf{X}_{\mathrm{s}} = \mathcal{M}(\mathbf{z}) = \mathcal{M}\left(u_{\mathrm{L}}, h_{\mathrm{R}}\right). \tag{8}$$

The control vector, i.e. the vector of boundary conditions, is assumed to follow a Gaussian distribution with mean $\mathbf{z}^{\mathrm{b}}$, which is the first guess – or background state –, and the background-error covariance matrix is $\mathbf{B}$. The observations are available at the end of the data assimilation window, i.e. when the steady state is reached, and are given by the vector $\mathbf{y}$. The observation likelihood is assumed to follow a Gaussian distribution:

$$p\left(\mathbf{y}|\mathbf{X}_{\mathrm{s}}\right) = n\left(\mathbf{y} - \mathcal{H}(\mathbf{X}_{\mathrm{s}})|\mathbf{0}, \mathbf{R}\right). \tag{9}$$

Bayes' rule combined with the aforementioned assumptions give the expression of the cost function:

$$\mathcal{J}(\mathbf{z}) = \frac{1}{2}\|\mathbf{z} - \mathbf{z}^{\mathrm{b}}\|^2_{\mathbf{B}^{-1}} + \frac{1}{2}\|\mathbf{y} - \mathcal{H} \circ \mathcal{M}(\mathbf{z})\|^2_{\mathbf{R}^{-1}}. \tag{10}$$

As previously noticed, this cost function is identical to the one used in 3D-Var (Eq. 3). The difference between these two methods lies in the fact that for the IEnKS the minimisation of the cost function is performed within the ensemble space instead of the control space for 3D-Var. The advantages of this approach are that, in general, the ensemble space is substantially smaller than the control space and the analysis in the ensemble space does not require the use of the tangent linear nor the adjoint model. The tangent linear model of $\mathcal{M}$ is a first-order approximation of the evolution, by $\mathcal{M}$, of a perturbation of the control vector. The adjoint of a linear operator $\mathbf{A}$ (typically the tangent linear model) is the linear operator $\mathbf{A}^*$ such that for any x, y:

$$\langle \mathbf{A}x, y\rangle = \langle x, \mathbf{A}^* y\rangle, \tag{11}$$

where $\langle ., .\rangle$ represents the inner product.

We consider an ensemble of N boundary condition vectors, that represents the background distribution: $\mathbf{E}^{\mathrm{b}} = \{\mathbf{z}_{[i]}\}_{i=1\ldots N}$, where subscript $[i]$ refers to the member index in the ensemble. The ensemble is centred on the background $\mathbf{z}^{\mathrm{b}}$ and the departure of each member from this mean is given by the (normalised) anomaly matrix:

$$\mathbf{A} = \left(1/\sqrt{N-1}\right)\left[\mathbf{z}_{[1]} - \mathbf{z}^{\mathrm{b}}, \mathbf{z}_{[2]} - \mathbf{z}^{\mathrm{b}}, ..., \mathbf{z}_{[N]} - \mathbf{z}^{\mathrm{b}}\right]. \tag{12}$$

The background error covariance matrix can be estimated from the ensemble:

$$\mathbf{B} = \mathbf{A}\mathbf{A}^{\mathrm{T}}. \tag{13}$$

We seek the combination of the ensemble members that gives the best estimate of $\mathbf{z}$, i.e. we look for the weight vector $\mathbf{w}^{\mathrm{a}}$ such that:

$$\mathbf{z}^{\mathrm{a}} = \mathbf{z}^{\mathrm{b}} + \mathbf{A}\mathbf{w}^{\mathrm{a}}, \tag{14}$$

where the 'a' superscript is used to refer to all the variables obtained at the end of the analysis cycle.

Replacing (13) and (14) in (10), the cost function in the ensemble space reads:

$$\widetilde{\mathcal{J}}(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{2}\|\mathbf{y} - \mathcal{H} \circ \mathcal{M}\left(\mathbf{z}^{\mathrm{b}} + \mathbf{A}\mathbf{w}\right)\|^2_{\mathbf{R}^{-1}}, \tag{15}$$

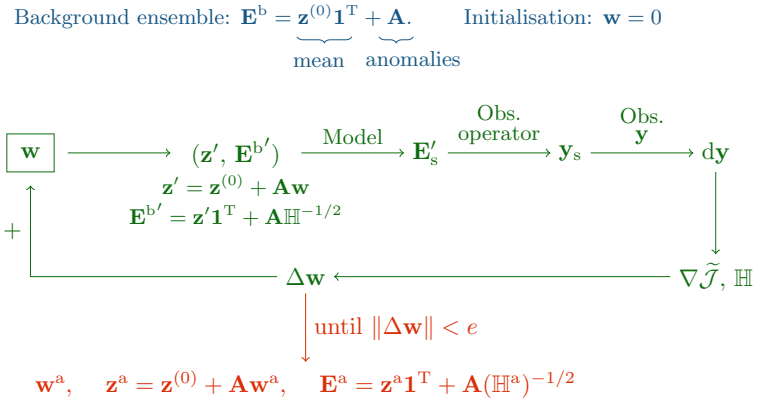


Figure 4: One analysis cycle of the IEnKS. The background ensemble $\mathbf{E}^{\mathrm{b}}$ is either obtained by a forecast ensemble from the previous analysis cycle or given as an input of the method. The best estimate of the weight vector $\mathbf{w}^{\mathrm{a}}$ is obtained by minimising the cost function $\widetilde{\mathcal{J}}$ as shown by the cycle on the figure: for each value of $\mathbf{w}$, a new ensemble of boundary conditions $\mathbf{E}^{\mathrm{b}'}$, centred on $\mathbf{z}'$, is generated using the *transform* method. The model and the observation operator are applied to this ensemble, which gives an ensemble of simulated observations with mean $\mathbf{y}_{\mathrm{s}}$ that can be compared to the observations $\mathbf{y}$. The increment $\mathrm{d}\mathbf{y} = \mathbf{y} - \mathbf{y}_s$ is used in the estimation of the gradient and Hessian of the cost function. The weight $\mathbf{w}$ is thus updated following Gauss-Newton algorithm until the convergence criterion is reached. At the end of the analysis, the best estimate of the control vector $\mathbf{z}^{\mathrm{a}}$ and the analysis ensemble $\mathbf{E}^{\mathrm{a}}$ can be used as a first guess for the next analysis cycle.

where $\|\mathbf{x}\|^2 = \mathbf{x}^{\mathrm{T}}\mathbf{x}$. The minimum of this cost function is reached for the most likely $\mathbf{w}$, referred to as $\mathbf{w}^{\mathrm{a}}$. Due to pre-conditioning by $\mathbf{A}$, as in Liu et al. (2008) and Gu & Oliver (2007), the calculation of the gradient of the cost function does not require the adjoint nor the tangent linear of $\mathcal{H} \circ \mathcal{M}$. It only involves the tangent linear of the operator transporting from the ensemble space to the observation space: $\mathbf{Y} = [\mathcal{H} \circ \mathcal{M}]'_{|\mathbf{z}}\,\mathbf{A}$, where $[\mathcal{H} \circ \mathcal{M}]'_{|\mathbf{z}}$ represents the tangent linear model of $\mathcal{H} \circ \mathcal{M}$ estimated by perturbations around $\mathbf{z}$. The tangent linear operator $\mathbf{Y}$, the gradient $\nabla\widetilde{\mathcal{J}}$ and the Hessian $\mathbb{H}$ of the cost function are estimated using the *transform* method as in Bocquet & Sakov (2012). A schematic representation of one analysis cycle of the IEnKS is shown on Figure 4.

It is worth noting that in our case with a single analysis cycle, the background ensemble has to be pre-constructed, satisfying the conditions of being centred on the background $\mathbf{z}^{\mathrm{b}}$ and corresponding to the background error covariance matrix $\mathbf{B}$. In cases with several IEnKS analysis cycles, the background ensemble, and thus $\mathbf{B}$, are given by the analysis ensemble from the previous analysis cycle.

3 Results

3.1 Experimental setup

3D-Var, the BFN algorithm and the IEnKS are tested on a shallow layer system with one-dimensional twin experiments (synthetic observations are extracted from the reference

simulation). The experiments correspond to a channel of length $L = 2500\mathrm{m}$ with a realistic topography profile but without diffusion nor ground friction. The reduced gravity is equal to $g' = 0.5g = 4.905\mathrm{m/s^2}$, which corresponds to a boundary layer twice as dense as the free atmosphere above. The true state – or reference simulation – is obtained with an upstream boundary condition equal to $u_{\mathrm{L}}^{\mathrm{t}} = 5.5\mathrm{m/s}$ and the first guess (or background) corresponds to the boundary value $u_{\mathrm{L}}^{\mathrm{b}} = 4.4\mathrm{m/s}$, i.e. a background error of 20% (Fig. 2). The value of the downstream boundary condition ($h_{\mathrm{R}} = 154\mathrm{m}$) is unchanged between the reference simulation and the background. In all the cases, we consider the steady state obtained with constant boundary conditions. The initial state for the data assimilation experiments is the steady state obtained with the *a priori* boundary conditions: in Sections 3.2 and 3.3 the same initial state is used for all the data assimilation experiments ($u_{\mathrm{L}}^{\mathrm{b}} = 4.4\mathrm{m/s}$ and $h_{\mathrm{R}} = 154\mathrm{m}$), while in Sections 3.2 the background $u_{\mathrm{L}}^{\mathrm{b}}$ differs between the experiments and the initial state varies accordingly. It is noteworthy that for simulations performed until convergence, the initial conditions does not influence the results, i.e. the steady state.

Two observations of the velocity, located at $x_1^o = 625\mathrm{m}$ and $x_1^o = 1875\mathrm{m}$ (see Fig. 2), are extracted from the reference simulation and are kept exact, i.e. noise-free, in Section 3.2 while they are noisy in Sections 3.3 and 3.4.

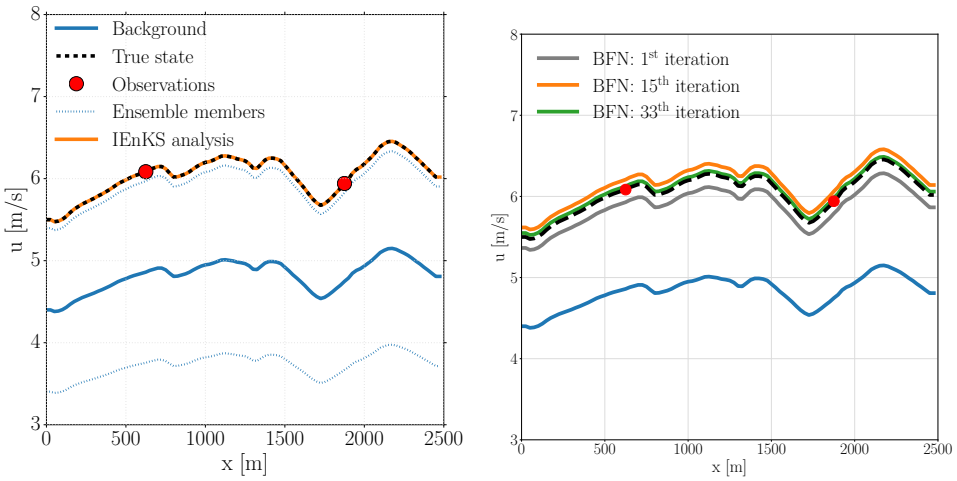
As noted in Section 2.3, the BFN does not have the same objective as 3D-Var and the IEnKS. To allow a quite fair comparison between these methods, we consider cases for which the uncertainty on the background is substantially larger than on observations. For 3D-Var and the IEnKS, the background error covariance matrix (**B**) and the observation error covariance matrix (**R**) have been chosen diagonal with variances of $1\mathrm{m^2/s^2}$ and $10^{-6}\mathrm{m^2/s^2}$ respectively. For each value of the left boundary condition, the steady state is obtained analytically by the resolution of the Bernoulli equation (Eq. 2). Equivalently, we could have integrated the numerical model during a long enough time until reaching the steady state. For the IEnKS, we take a background ensemble of 2 members corresponding to a background error covariance matrix equal to identity, centred on u^{b}, and the convergence criteria is set to $e = 10^{-3}$.

For the BFN algorithm, the forward and backward integrations with nudging are performed over a time period T, which corresponds to the time needed for the information coming from the observations to reach the boundary conditions. Using the method of characteristics we have estimated this time period to be $T = 30\mathrm{s}$. The nudging is applied every time step ($\Delta t = 0.5\mathrm{s}$) and the nudging matrices are the same in the forward and backward integrations: $\mathbf{K} = k\mathbf{H}^{\mathrm{T}}$ where $k = 1\mathrm{s}^{-1}$. This definition of nudging matrices is equivalent to $\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}$, often used in nudging studies. In fact, here the observations are independently perturbed by noise such that $\mathbf{R}$ is here proportional to the identity matrix: $\mathbf{R} = r\mathbf{I}$. Moreover the numerical scheme used to solve the shallow water equations is explicit such that the nudging coefficient must be smaller than one. In the present study, $r = 10^{-6}$ and 0.25 (without and with noise), in which case r^{-1} is too large. Consequently we take the largest possible value for k, which is 1.

We assume that the BFN algorithm has converged when the relative variation of the control vector (here the boundary condition u_{L}) between two consecutive iterations is smaller than 0.05% during 5 consecutive iterations.

3.2 Results with perfect observations

Figures 5a and 5b show the background in blue, the reference simulation in dashed black, and the observations as red dots.



(a) Results of the IEnKS with perfect observations

(b) Results of the BFN algorithm with perfect observations

Figure 5: Profiles of the velocity obtained: (a) at the end of the IEnKS analysis and (b) after 1, 15, and 33 BFN iterations for the experiment with perfect observations. The simulations are performed with a shallow layer model without diffusion nor ground friction.

Figures 5a shows the steady state corresponding to the boundary conditions of the 2 ensemble members (dotted, blue curves) and of the one calculated by the IEnKS analysis (orange curve). In this simple case the IEnKS perfectly recovers the value of the boundary condition, so does 3D-Var (not shown on the figure).

Figure 5b shows the steady state corresponding to the boundary conditions obtained after 1, 15, and 33 BFN iterations in the experiment with perfect observations. A BFN iteration consists of one forward and one backward integrations. The BFN algorithm has converged after 33 iterations, i.e. 66 model integrations over a period of $T = 30\text{s}$, and the boundary condition obtained is $u_{\text{L}}^{\text{a}} = 5.54\text{m/s}$. The relative RMSE between the true state and the state obtained with this value of boundary condition is equal to 0.68%, when evaluated over the whole domain.

The results of both the IEnKS and 3D-Var are very good, though these methods have a slightly higher computational cost. In fact, here the resolution of the Bernoulli equation (2) is not costly but in general cases where no analytical solution is available, the model should be integrated over a time window long enough to reach the convergence, and consequently longer than the period T used in the BFN algorithm. In fact, each iteration of the BFN algorithm (i.e. one successive forward and backward integrations) requires nearly 30 times less computational resources than one analysis cycle of IEnKS with two members or one finite-difference calculation for 3D-Var minimisation. However, the IEnKS and 3D-Var can be partially parallelised, unlike the BFN algorithm. The IEnKS is here still more efficient than 3D-Var as the minimisation of the cost function only requires 2 iterations of the Gauss-Newton algorithm, i.e. this method requires 4 model integrations as we consider 2 ensemble members. In comparison, 3D-Var in ADAO estimates the gradient by finite-differences such that each iteration of the optimisation algorithm requires 2 model integrations. Here,

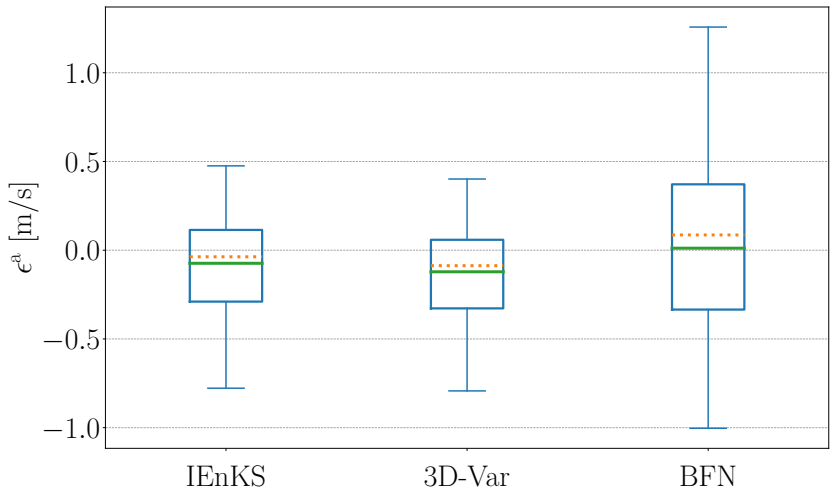


Figure 6: Boxplots corresponding to the analysis errors obtained with 50 pairs of noisy observations using the IEnKS, 3D-Var, and the BFN algorithm. The bottom and top of the boxes represent the 1st and 3rd quartiles, the green line corresponds to the mean, the dashed orange line to the median, and the ends of the whiskers represent the minimum and maximum of all the data.

6 iterations are necessary to reach the minimum of the cost function, which corresponds to 12 model integrations.

3.3 Results with noisy observations

In real studies, the observations are never perfect. Consequently, we have analysed the ability of the data assimilation methods to assimilate noisy observations.

We have generated an ensemble of 50 pairs of observation errors following a Gaussian distribution with zero mean and a covariance matrix equal to $0.25\mathbf{I}$, where $\mathbf{I}$ is the 2×2 identity matrix. For 3D-Var and the IEnKS, we thus set the observation error covariance matrix to $\mathbf{R} = 0.25\mathbf{I}$ in what follows.

Figure 6 shows boxplots corresponding to the distribution of the analysis errors for the 50 cases with noisy observations, for each data assimilation method. The analysis error is defined as the difference between the value of the boundary condition given by the analysis and the true value of the boundary condition:

$$\epsilon^{\mathrm{a}} = u_{\mathrm{L}}^{\mathrm{a}} - u_{\mathrm{L}}^{\mathrm{t}}. \tag{16}$$

We observe that the three methods have small mean analysis errors (green line) and quite small dispersions of the errors (blue boxes). The IEnKS and 3D-Var give good results with small mean analysis errors (respectively $-0.07\mathrm{m/s}$ and $-0.12\mathrm{m/s}$) and small standard deviation (both $0.27\mathrm{m/s}$). The BFN algorithm gives the smallest mean analysis error ($0.01\mathrm{m/s}$) though the standard deviation around this mean is larger than for the two other methods ($0.5\mathrm{m/s}$). This larger variability shows that the BFN algorithm is more sensitive

to observation errors. It is consistent with the definition of the BFN, which gives a large importance to observations. If observations are not perfect, the BFN algorithm nudges the system toward a state that is close to these observations and which might be different from the true state.

A useful measure of the uncertainty on the analysis is the precision, which is defined as the inverse of the analysis error variance:

$$p^{\mathrm{a}} = \frac{1}{\sigma_{\mathrm{a}}^2}. \tag{17}$$

Theoretically, the analysis precision is the sum of the precisions of the background and of the observations (Section 5.4.1 in Kalnay, 2003). The precision of the observations is calculated using the adjoint of the forward operator $\widetilde{\mathbf{H}}$, estimated by the IEnKS and 3D-Var using either the ensemble or finite differences. We deduce the precision of the observations: $p^{\mathrm{o}} = \widetilde{\mathbf{H}}^{\mathrm{T}}\mathbf{R}^{-1}\widetilde{\mathbf{H}} \approx 39\mathrm{s}^2/\mathrm{m}^2$. In this section, we analyse the sensitivity to the observation errors only and the background is the same for the 50 experiments. Consequently, we compare the analysis precisions to the observation precision. Both the IEnKS and 3D-Var give precisions of $14\mathrm{s}^2/\mathrm{m}^2$ which is smaller than the theoretical value, especially because the statistical assumptions are not perfectly satisfied. However, the precisions are not too low neither, indicating that the methods are not very sensitive to the observation errors. In particular, for these two methods the absolute value of the analysis error is smaller than the background error ($|\epsilon^{\mathrm{b}}| = |u_{\mathrm{L}}^{\mathrm{b}} - u_{\mathrm{L}}^{\mathrm{t}}| = 1.1\mathrm{m/s}$), showing that the data assimilation methods always help correct the boundary condition. As mentioned above, the BFN algorithm is intrinsically more sensitive to observation errors, which is consistent with the smaller analysis precision of $4\mathrm{s}^2/\mathrm{m}^{-2}$.

3.4 Analysis of sensitivity to the background error

We have also analysed the sensitivity of the data assimilation methods to the first guess $u_{\mathrm{L}}^{\mathrm{b}}$. Similar to the sensitivity analysis to observation error, we have generated an ensemble of 20 background errors, following a Gaussian distribution with zero mean and a variance of $\sigma_{\mathrm{b}}^2 = 1\mathrm{m}^2/\mathrm{s}^2$, symmetrical with respect to zero. We have also generated 10 pairs of observation errors, as in Section 3.3. The three data assimilation methods have been tested on these 200 cases.

Figure 7 shows the analysis error against the background error for the IEnKS (blue squares), 3D-Var (orange triangles), and the BFN algorithm (green dots). The markers represent the absolute analysis error, averaged over the 10 pairs of noisy observations for a same background error, and the errorbars indicate the standard deviation around these mean values. The absolute values of the background error are represented by the grey line ($y = |x|$): in the dotted areas the analysis error is greater than the background error, indicating that the data assimilation method does not help correct the simulations.

We observe in Figure 7 that the mean analysis errors of 3D-Var and the IEnKS (blue squares and orange triangles) are proportional to the background error. This is consistent with the Bayesian framework of these methods. We can also verify that the sensitivity of the IEnKS and 3D-Var to the background error is quite small here, which is in agreement with the fact that the background precision is here substantially smaller than the observation precision. The BFN algorithm does not depend on the background and we can verify that the mean analysis error (green circles) is nearly constant. For neither of the three methods

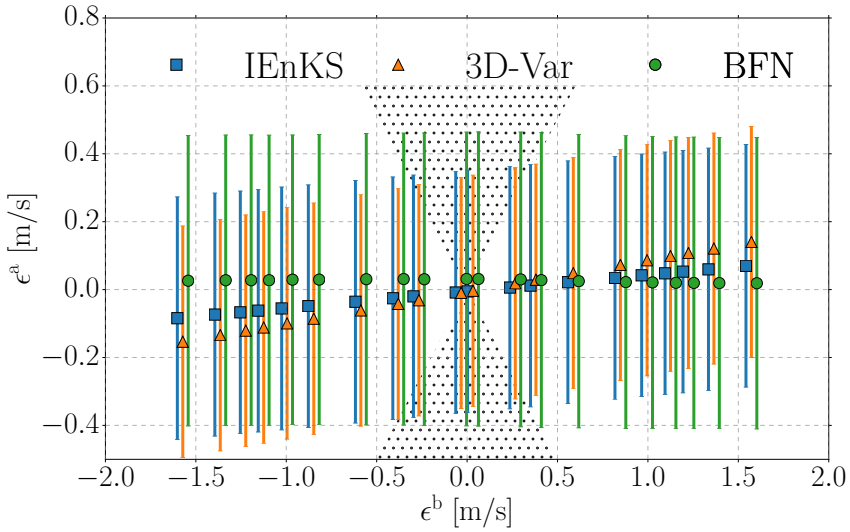


Figure 7: Analysis error against background error for 20 values of background error and for the three data assimilation methods: the IEnKS (blue squares), 3D-Var (orange triangles), and the BFN algorithm (green circles). The errorbars are centred on the mean analysis error and show the standard deviation, calculated over 10 experiments with different pairs of observation errors and a given background error. The dotted areas correspond to absolute values of analysis error larger than the background error, i.e. the data assimilation method has increased the error.

does the sensitivity to observation error – represented by the length of the blue, orange, and green errorbars – depend on the background error. We can verify here again that the green error bars are longer than the blue and orange ones, indicating that the BFN algorithm is more sensitive to observation errors than 3D-Var and the IEnKS.

Except for very small absolute background errors ($< 0.4\mathrm{m/s}$), all the errorbars are below the grey line (Fig. 7), indicating that in nearly all the cases, the three data assimilation methods help reduce the error on the boundary condition. The cases with very small $|\epsilon^{\mathrm{b}}|$ correspond to cases for which the background are very close to the true state. However, the background error variance is still larger than the observation error variance ($\sigma_{\mathrm{b}}^2 = 1\mathrm{m}^2/\mathrm{s}^2 > \sigma_{\mathrm{o}} = 0.5\mathrm{m}^2/\mathrm{s}^2$), which means that the confidence placed in the background is smaller than the confidence in the noisy observations. Consequently, in 3D-Var and the IEnKS a smaller weight is given to the background compared to the one given to the observations. As the observations are noisy, the value of the control vector that is the closest to the observations might be further away from the true state than the background. Consequently, the analysis error is somewhat larger than the background error and on Figure 7 the errorbars go over the edges of the dotted region.

The results with the BFN algorithm are consistent with the independence of the analysis to the background error and with the quite large sensitivity to observation error, explained in the previous section.

Another way to measure the sensitivity of the methods to background error is to look at the mean analysis error and the standard deviation in the ensemble of 20 experiments for each pair of noisy observations. The IEnKS analysis error depends little on the background

value: for each pair of noisy observations the mean analysis error is smaller than $0.5\mathrm{m/s}$ and the standard deviation around this average value is smaller than $0.05\mathrm{m/s}$. 3D-Var gives similar results to the IEnKS, even though the standard deviations are slightly larger, indicating that the method is somehow more sensitive to the background error. This is consistent with Figure 7 in which the orange triangles follow a line with a larger slope than the blue squares.

With this measure we can also confirm that the BFN algorithm is more sensitive to observation error but less to background error than the two variational methods. In fact, depending on the pair of observations considered, the mean analysis error - calculated over the 20 analyses obtained with different background errors - vary from $0.12\mathrm{m/s}$ to $1\mathrm{m/s}$. The standard deviations around theses means are all around $0.005\mathrm{m/s}$.

The analysis precision (eq. 17) is a measure of the confidence in the analysis: the higher the precision, the lower the uncertainty on the analysis. In this section, both the background and the observations vary, thus the analysis precision should be the sum of the precisions of the background ($p^{\mathrm{b}} = \frac{1}{\sigma_{\mathrm{b}}^2} = 1\mathrm{s}^2/\mathrm{m}^2$) and the precision of the observations ($p^{\mathrm{o}} = 39\mathrm{s}^2/\mathrm{m}^{-2}$, Section 3.3). Consequently the analysis precision should theoretically be equal to $p^{\mathrm{a}} = p^{\mathrm{b}} + p^{\mathrm{o}} = 40\mathrm{s}^2/\mathrm{m}^2$ which gives an estimation of the standard deviation for the analysis velocity at boundary: $\sigma_{\mathrm{a}} = \sqrt{\frac{1}{40}} = 0.16\mathrm{m/s}$. Overall, considering all the 200 simulations for each method, the mean absolute analysis error is equal to $-0.007\mathrm{m/s}$ for the IEnKS and 3D-Var and $0.026\mathrm{m/s}$ for the BFN algorithm with precisions of $8\mathrm{s}^2/\mathrm{m}^2$ for the IEnKS and 3D-Var and $5\mathrm{s}^2/\mathrm{m}^2$ for the BFN algorithm. The analysis precisions of all the three methods are greater than the background precision, which confirms that data assimilation methods improve the estimation of the control vector.

Regarding computational efficiency, the BFN algorithm is always the most cost-effective because it is not integrated until convergence. The IEnKS is more effective than 3D-Var in these cases since only two or three iterations of the Gauss-Newton algorithm are necessary to minimise the cost function whereas for 3D-Var the minimisation by finite differences requires at least 5 iterations.

4 Conclusion

Variants of the BFN algorithm, 3D-Var, and the IEnKS have been derived for local flows to take boundary conditions into account instead of the usual initial conditions. The derivations of 3D-Var and the IEnKS have been performed in a general framework whereas the BFN derivation is specific to the 1D shallow water equations. In fact, the shallow layer model offers a quite good representation of geophysical flows and is often used for simple numerical experiments in meteorology and hydrology. In the present study, we consider a 1D shallow layer model in order to validate the numerical behaviour of the modified data assimilation methods before applying them to more complex model and cases.

The first results presented here show the efficiency of these methods to assimilate a few observations on cases with a complex one-dimensional geometry. The three methods help correct the boundary conditions, and thus obtain a state that is closer to the true state than the first guess. It has been shown that even if the observations are noisy and for a quite large range of background errors, the three methods are nearly always able to improve the estimation of the boundary conditions.

The IEnKS and 3D-Var depend little on observation error but depend linearly on background error. The BFN algorithm is more sensitive to the observation errors but not to the background. The behaviour of the three methods is in good agreement with theory and highlights the fundamentally different data assimilation problem solved by 3D-Var and IEnKS on one hand and the BFN algorithm on the other hand. Furthermore, the BFN algorithm has the disadvantage of being sensitive to parameters such as the nudging matrix, the length of the integration time window, and the location of the observations. Here we have used a very simple nudging matrix. Further investigation could be performed to develop more complex observers, as in Auroux & Bonnabel (2011), Apte et al. (2018) or Krstic et al. (2009), in order to correct fluid height with velocity observations.

In this simple example, the IEnKS is the most efficient data assimilation method as it gives the smallest analysis error, with the greatest precision, and it requires less model integrations than 3D-Var. However, the 1D case analysed in the present study is quite simple and the control vector considered is a singleton. Further investigation on more complex cases, especially with two horizontal dimensions and vertical profiles of velocity will be performed to analyse the performances of the methods on larger control vectors. Eventually, we will apply these methods to more realistic cases with the CFD model *Code_Saturne*.

References

Abbott, M. (1966). *An introduction to the Method of Characteristics*. American Elsevier, New York, NY.

ADAO, a module for Data Assimilation and Optimization, http://www.salome-platform.org/.

Alcrudo, F. & Garcia-Navarro, P. (1993). A high-resolution Godunov-type scheme in finite volumes for the 2D shallow-water equations. *International Journal for Numerical Methods in Fluids*, *16*(6), 489–505.

Apte, A., Auroux, D., & Ramaswamy, M. (2018). Observers for compressible Navier-Stokes equation. *SIAM Journal on Control and Optimization*, *56*(2), 1081–1104.

Archambeau, F., Méchitoua, N., & Sakiz, M. (2004). Code Saturne: A Finite Volume Code for the computation of turbulent incompressible flows - Industrial Applications. *International Journal on Finite Volumes*, *1*.

Asch, M., Bocquet, M., & Nodet, M. (2016). *Data assimilation: methods, algorithms, and applications*. Society for Industrial and Applied Mathematics.

Audusse, E., Bristeau, M. O., & Decoene, A. (2006). 3D Free Surface Flows Simulations Using a Multilayer Saint-Venant Model. Comparisons with Navier-Stokes Solutions. In *Numerical Mathematics and Advanced Applications: Proceedings of ENUMATH 2005, the 6th European Conference on Numerical Mathematics and Advanced Applications Santiago de Compostela, Spain, July 2005* (pp. 181–189). Berlin, Heidelberg: Springer Berlin Heidelberg.

Auroux, D. (2008). The back and forth nudging algorithm applied to a shallow water model, comparison and hybridization with the 4D-VAR. *International Journal for Numerical Methods in Fluids*, *61*, 911–929.

Auroux, D., Bansart, P., & Blum, J. (2013). An evolution of the back and forth nudging for geophysical data assimilation: application to Burgers equation and comparisons. *Inverse Problems in Science and Engineering*, *21*(3), 399–419.

Auroux, D. & Blum, J. (2005). Back and forth nudging algorithm for data assimilation problems. *Comptes Rendus Mathematique*, *340*, 873–878.

Auroux, D. & Blum, J. (2008). A nudging-based data assimilation method: the Back and Forth Nudging (BFN) algorithm. *Nonlinear Processes in Geophysics*, *15*, 305–319.

Auroux, D., Blum, J., & Nodet, M. (2011). Diffusive Back and Forth Nudging algorithm for data assimilation. *Comptes Rendus Mathematique*, *349*, 849–854.

Auroux, D. & Bonnabel, S. (2011). Symmetry-based observers for some water-tank problems. *IEEE Trans. Automat. Control*, *56*(5), 1046–1058.

Auroux, D. & Nodet, M. (2012). The Back and Forth Nudging algorithm for data assimilation problems : theoretical results on transport equations. *ESAIM: Control, Optimisation and Calculus of Variations*, *18*, 318–342.

Bocquet, M. & Sakov, P. (2012). Combining inflation-free and iterative ensemble Kalman filters for strongly nonlinear systems. *Nonlinear Processes in Geophysics*, *19*(3), 383–399.

Bocquet, M. & Sakov, P. (2014). An iterative ensemble Kalman smoother. *Quarterly Journal of the Royal Meteorological Society*, *140*, 1521–1535.

Byrd, R. H., Lu, P., Nocedal, J., & Zhu, C. (1995). A Limited Memory Algorithm for Bound Constrained Optimization. *SIAM Journal on Scientific Computing*, *16*(5), 1190–1208.

Daley, R. (1991). *Atmospheric data analysis*. Cambridge University Press.

Goutal, N. & Maurel, F. (1997). *Proceedings of the 2nd workshop on dam-break wave simulation*. Electricité de France. Direction des études et recherches.

Gu, Y. & Oliver, D. S. (2007). An Iterative Ensemble Kalman Filter for Multiphase Fluid Flow Data Assimilation. *SPE Journal*, *12*, 438–446.

Kalnay, E. (2003). *Atmospheric modeling, data assimilation, and predictability*. Cambridge university press.

Krstic, M., Magnis, L., & Vazquez, R. (2009). Nonlinear Control of the Viscous Burgers Equation: Trajectory Generation, Tracking, and Observer Design. *Journal of Dynamic Systems, Measurement, and Control*, *131*.

Liu, C., Xiao, Q., & Wang, B. (2008). An Ensemble-Based Four-Dimensional Variational Data Assimilation Scheme. Part I: Technical Formulation and Preliminary Test. *Monthly Weather Review*, *136*, 3363–3373.

Pedlosky, J. (2013). *Geophysical fluid dynamics*. Springer Science & Business Media.

Pielke, R. A. (2013). *Mesoscale meteorological modeling*. Academic Press.

Sakov, P., Oliver, D. S., & Bertino, L. (2012). An Iterative EnKF for Strongly Nonlinear Systems. *Monthly Weather Review*, *140*, 1988–2004.

SALOME. The Open Source Integration Platform for Numerical Simulation, http://www.salome-platform.org/.