Data Assimilation for Geophysical Fluids: The Diffusive Back and Forth Nudging

Didier Auroux, Jacques Blum, and Giovanni Ruggiero

Abstract Data assimilation is the domain at the interface between observations and models, which makes it possible to identify the global structure of a geophysical system from a set of discrete space-time data. After recalling state-of-the-art data assimilation methods, the variational 4DVAR algorithm and sequential Kalman filters, but also the Back and Forth Nudging (BFN) algorithm, we present the Diffusive Back and Forth Nudging (DBFN) algorithm, which is a natural extension of the BFN to some particular diffusive models. We numerically test the DBFN and compare it with other methods on both a 2D shallow water model, and a 3D full primitive ocean model. These numerical results show that the DBFN converges very quickly. Its main advantage is to filter the observation noise, unlike the BFN algorithm, particularly when the model diffusion is high.

Keywords Data assimilation • Geophysics • Nudging • Oceanography • Shallow water • Primitive equations • Back and forth nudging

1 Introduction

It is well established that the quality of weather and ocean circulation forecasts is highly dependent on the quality of the initial conditions. Geophysical fluids (air, atmospheric, oceanic, surface or underground water) are governed by the general equations of fluid dynamics. Geophysical processes are hence non linear because of their fluid component. Such non linearities impose a huge sensitivity to the initial conditions, and then an ultimate limit to deterministic prediction (estimated to be about 2 weeks for weather prediction for example). This limit is still far from being reached, and substantial gain can still be obtained in the quality of forecasts.

D. Auroux (✉) • J. Blum
Laboratoire J.A. Dieudonné, Université de Nice Sophia Antipolis, Parc Valrose, 06108 Nice cedex 2, France
e-mail: auroux@unice.fr; jblum@unice.fr

G. Ruggiero
Mercator Océan, 10 Rue Hermès, 31520 Ramonville-Saint-Agne, France
e-mail: giovanni.ruggiero@mercator-ocean.fr

F. Ancona et al. (eds.), *Mathematical Paradigms of Climate Science*, Springer INdAM Series 15, DOI 10.1007/978-3-319-39092-5_8

This can be obtained through improvement of the observing system itself, but also through improvement of the geophysical models used to modelize the geophysical processes. For example, a major problem comes from the fact that sub-scaled processes could be associated with extremely large fluxes of energy. Seeking a numerical solution to the equations requires discretizing the equations, and therefore cutting off in the scales. It will be crucial to represent the fluxes of energy associated to sub-grid processes by some additional terms in the equations [36, 50].

Over the past 20 years, observations of ocean and atmosphere circulation have become much more readily available [13], as a result of new satellite techniques and international field programs (MERCATOR, CLIPPER, GODAE, ARGO, ...). In the case of the ocean modelling, the use of altimeter measurements has provided extremely valuable information about the sea-surface height, and then has allowed the oceanographic community to study more precisely both the general circulation of the ocean and the local dynamics of some particular regions (the Gulf Stream area, for example, but also the Kuroshio extension, the Antarctic circumpolar current and the tropical oceans). Geostationnary satellites also provide information on the wind by estimating the shifting of clouds considered as lagrangian tracers. Polar orbiting satellites are used for the estimation of the atmospheric vertical temperature profiles. Generally, radiances are measured and then temperatures are estimated as the solution of an inverse problem.

Meteorologic and oceanographic data are currently extremely heterogeneous, both in nature, density and quality, but their number is still smaller than the degree of freedom of the models. The growth of the available computing ressources indeed allows refinements of the grid size of general circulation models.

Environmental scientists are increasingly turning to inverse methods for combining in an optimal manner all the sources of information coming from theory, numerical models and data. Data assimilation (DA) is precisely the domain at the interface between observations and models which makes it possible to identify the global structure of a system from a set of discrete space-time data. DA covers all the mathematical and numerical techniques in which the observed information is accumulated into the model state by taking advantage of consistency constraints with laws of time evolution and physical properties, and which allow us to blend as optimally as possible all the sources of information coming from theory, models and other types of data.

There are two main categories of data assimilation techniques [58], variational methods based on the optimal control theory [42] and statistical methods based on the theory of optimal statistical estimation (see, for example, [12, 13, 39] for an overview of inverse methods, both for oceanography and meteorology). The first class of methods (3D-VAR, 4D-VAR, 4D-PSAS, ...) was first introduced in meteorology [19, 41, 59] and more recently for oceanic data [45, 48, 49, 54, 55, 60]. The statistical (or sequential) methods (optimal interpolation, Kalman filter, SEEK filter, ...) were introduced in oceanography roughly 15 years ago [30, 32]. The Kalman filter was extended to nonlinear cases [29, 38] but it has been mostly applied in oceanography to quasi-linear situations, in particular tropical oceans [16, 26, 27, 34, 62].

In practice, all data assimilation techniques encounter major difficulties due to computational reasons. The full Kalman filter would, in principle, require the manipulation of matrices with a dimension of typically 10^7 or 10^8 in an oceanic problem. The optimal control adjoint method often requires several hundred iterations of the minimization process to converge, thus implying an equivalent number of model runs. In this context, it is important to find new data assimilation algorithms allowing in particular a reduction of the problem dimension.

In this paper, we focus our interest on various data assimilation algorithms in order to identify the initial condition of a geophysical system and reconstruct its evolution in time and space.

We first study in Sect. 2 the four dimensional variational adjoint method (named 4D-VAR), using a strong constraint hypothesis (the ocean circulation model is assumed to be exact). The use of a cost function, measuring the mean-square difference between the observations and the corresponding model variables, allows us to carry out the assimilation process by an identification of the initial state of the ocean which minimizes the cost function.

Sequential methods are mostly based on the Kalman filtering theory, which consists in a forecast step and an analysis (or correction) step. In Sect. 3, we present the extended Kalman filter (EKF), for nonlinear models. A main drawback of the (extended) Kalman filter is the computational cost of propagating in time the error covariance matrices. We present then the ensemble Kalman filter (EnKF), for which an ensemble of states (members) is used to compute actual covariance matrices at a lower computational cost.

We recall in Sect. 4 the Back and Forth Nudging (BFN) algorithm, which is the prototype of a new class of data assimilation methods, although the standard nudging algorithm is known for a couple of decades. It consists in adding a feedback term in the model equations, measuring the difference between the observations and the corresponding space states. The idea is to apply the standard nudging algorithm to the backward (in time) nonlinear model in order to stabilize it. The BFN algorithm is an iterative sequence of forward and backward resolutions, all of them being performed with an additional nudging feedback term in the model equations. We also present the Diffusive Back and Forth Nudging (DBFN) algorithm, which is a natural extension of the BFN to some particular diffusive models. This section ends with theoretical considerations on both BFN and DBFN algorithms.

In Sect. 5, we study the behaviour of the DBFN algorithm on a 2D shallow water model. We report the results of numerical experiments, showing the convergence of the DBFN algorithm, a comparison between BFN, DBFN and 4DVAR algorithms, and sensitivity studies with respect to observation noise, and to the model diffusion coefficient. Note that we compare the BFN and DBFN algorithms with the 4DVAR algorithm, as they all consist in iterative resolutions of forward and backward (or adjoint) models, and they also all update the initial condition at each iteration. Kalman filter methods are sequential and do not update the initial condition. This is why we decided not to compare numerically BFN and DBFN algorithms with a Kalman filter approach.

In Sect. 6, we present numerical results on a full primitive ocean model (NEMO). After showing the convergence of the DBFN algorithm, we compare the BFN and DBFN algorithms.

Finally, some concluding remarks and perspectives are shown in Sect. 7.

2 Variational Method: 4D-VAR

Variational methods consider the equations governing the geophysical flow as constraints, and the problem is closed by using a variational principle, e.g. the minimization of the discrepancy between the model and the observations.

2.1 Model and Observations

Every DA method needs both a model describing the evolution of the fluid, basically a system of non linear partial differential equations (PDE), and a set of discrete observations. Firstly, we assume that the model can be written, after discretization in space of the set of PDE:

$$\begin{cases} \dfrac{dX}{dt} = F(X, U), \quad 0 < t < T, \\ X(0) = V, \end{cases} \tag{1}$$

where X is the state variable which describes the evolution of the system at each grid point. X depends on time, and is for operational models of large dimension (10^7–10^8). F is a non linear differential operator, describing the dynamics of the system. U corresponds to some internal variables of the model (parameters or boundary conditions) and may be time dependent. Finally, V is the initial condition of the system state, which is unknown. In order to use optimal control techniques, we have to define a control variable that should be identified. Most of the time, the control is (U, V), the initial condition and the model parameters.

Secondly, we suppose that we have an observation vector X_{obs} which gathers all the data we want to assimilate. These observations are discrete in time and space, distributed all over the assimilation period $[0, T]$, and are not in the same space as the state variable, from a geographical or a physical point of view. Therefore, we will need an observation operator C mapping the space of state into the space of observations. This operator can be non linear in some cases.

2.2 Cost Function

It is now possible to define a cost function $\mathscr{J}$ measuring the discrepancy of the solution of the model associated to the control vector (U, V) and the observations X_{obs}:

$$\mathscr{J}(U,V) = \frac{1}{2}\int_0^T \langle R^{-1}(CX - X_{obs}), CX - X_{obs}\rangle dt + \frac{1}{2}\langle P_0^{-1}V, V\rangle + \frac{1}{2}\int_0^T \langle Q^{-1}U, U\rangle dt \tag{2}$$

where X is the solution of (1). R, P_0 and Q are covariance matrices, allowing us to introduce some a priori information about the statistics of the fields X_{obs}, V and U respectively. $\langle ., . \rangle$ is most of the time the canonical real scalar product.

The first part of the cost function quantifies the difference between the observations and the state function, and the two others act like a regularization term in the sense of Tykhonov. It is sometimes replaced by the so-called background term, which is the quadratic (with respect to the covariance matrix norm) difference between the initial optimal variable and the last prediction [19].

The inverse problem which consists in the minimization of the cost function $\mathscr{J}$ is then generally well-posed. The variational formulation of our DA problem can then be written as:

$$\begin{cases} \text{Find}(U^*, V^*) \text{ such that} \\ \mathscr{J}(U^*, V^*) = \inf\limits_{(U,V)} \mathscr{J}(U, V). \end{cases} \tag{3}$$

2.3 Gradient Step

In order to minimize the cost function, we need its gradient $\nabla\mathscr{J}$. Because of the large dimension of the model state vector (usually more than 10^7), it is not possible to compute directly the gradient by using finite difference methods. The gradient vector of the functional is then obtained by the adjoint method [18, 19]. Let $\hat{X}$ be the derivative of X with respect to (U, V) in the direction (u, v). Then $\hat{X}$ is solution of the following set of discretized partial differential equations, known as the linear tangent model:

$$\begin{cases} \dfrac{d\hat{X}}{dt} = \dfrac{\partial F}{\partial X}\hat{X} + \dfrac{\partial F}{\partial U}u, \\ \hat{X}(0) = v, \end{cases} \tag{4}$$

where $\frac{\partial F}{\partial X}$ and $\frac{\partial F}{\partial U}$ represent the jacobian of the model with respect to the state variable and the model parameters respectively.

If we assume that the operator C is linear (otherwise, we have to linearize it), the derivative of $\mathscr{J}$ with respect to (U, V) in the direction (u, v) is then

$$\langle \hat{\mathscr{J}}(U,V),(u,v)\rangle = \int_0^T \langle R^{-1}(CX - X_{obs}), C\hat{X}\rangle dt$$
$$+\langle P_0^{-1}V, v\rangle + \int_0^T \langle Q^{-1}U, u\rangle dt.$$

We can introduce the so-called adjoint state P (which lives in the same space as X), solution of the adjoint model [19]:

$$\begin{cases} -\frac{dP}{dt} = \left(\frac{\partial F}{\partial X}\right)^T P - C^T R^{-1}(CX - X_{obs}), \\ P(T) = 0. \end{cases} \tag{5}$$

We have then:

$$\langle \hat{\mathscr{J}}(U,V),(u,v)\rangle = \int_0^T \langle \frac{dP}{dt} + \left(\frac{\partial F}{\partial X}\right)^T P, \hat{X}\rangle dt$$
$$+\langle P_0^{-1}V, v\rangle + \int_0^T \langle Q^{-1}U, u\rangle dt$$

and an integration by part shows that, using (4):

$$\langle \hat{\mathscr{J}}(U,V),(u,v)\rangle = \int_0^T \langle -P, \frac{\partial F}{\partial U}u\rangle dt - \langle P(0), v\rangle$$
$$+\langle P_0^{-1}V, v\rangle + \int_0^T \langle Q^{-1}U, u\rangle dt.$$

Finally, the gradient of $\mathscr{J}$ is given by:

$$\nabla \mathscr{J}(U,V) = \begin{pmatrix} -\left(\frac{\partial F}{\partial U}\right)^T P + Q^{-1}U \\ -P(0) + P_0^{-1}V \end{pmatrix}. \tag{6}$$

Therefore, the gradient is obtained by a backward integration of the adjoint model (5), which has the same computational cost as one evaluation of $\mathscr{J}$.

2.4 Optimality System

The minimization problem (3) is then equivalent to the following optimality system:

$$\begin{cases} \dfrac{dX}{dt} = F(X, U^*), \\ X(0) = V^*, \end{cases}$$

$$\begin{cases} -\dfrac{dP}{dt} = \left(\dfrac{\partial F}{\partial X}\right)^T P - C^T R^{-1}(CX - X_{obs}), \\ P(T) = 0, \end{cases} \tag{7}$$

$$\left(\frac{\partial F}{\partial U}\right)^T P = Q^{-1}U^*,$$

$$P(0) = P_0^{-1}V^*.$$

2.5 4D-Var Algorithm Computation

The determination of (U^*, V^*), solution of (3) and (7), is carried out by running a descent-type optimization method. We may use as a first guess (U_0, V_0) the result of the minimization process at the last prediction. Then, given the first guess, we use an iterative algorithm [33]:

$$(U_n, V_n) = (U_{n-1}, V_{n-1}) - \rho_n D_n$$

where D_n is a descent direction, and ρ_n is the step size.

The knowledge of (U_{n-1}, V_{n-1}) allows us to compute the corresponding solution X_{n-1} of the direct model (1), and consequently to evaluate the cost function $\mathscr{J}(U_{n-1}, V_{n-1})$. Then we solve the adjoint model (5) and compute the adjoint solution P_{n-1}, and using (6), the gradient of the cost function $\nabla\mathscr{J}(U_{n-1}, V_{n-1})$. The computation of the descent direction D_n is usually performed using conjugate gradient or Newton type methods. Finally, the step size ρ_n is chosen to be the step size which minimizes

$$\mathscr{J}((U_{n-1}, V_{n-1}) - \rho D_n)$$

with respect to ρ. This is a one-dimensional minimization, but in case the problem is non linear, we can get a high computational cost because it will require several evaluations of $\mathscr{J}$, and hence several integrations of the model (1) [15, 33, 43, 61].

2.6 Computational Issues

One of the most difficult steps in the 4D-Var algorithm is the implementation of the adjoint model. Numerically, the goal is to solve the discrete optimality system, which gives the solution of the discrete direct problem, and the discrete gradient is given by the discrete adjoint model, which has to be derived from the discrete direct model, and not from the continuous adjoint model. A bad solution would be to derive the adjoint model from the continuous direct model, and then to discretize it. The good solution is to first derive the tangent linear model from the direct model. This can be done by differentiating the direct code line by line. And then one has to transpose the linear tangent model in order to get the adjoint of the discrete direct model. To carry out the transposition, one should start from the last statement of the linear tangent code and transpose each statement. The derivation of the adjoint model can be long. Sometimes, it is possible to use some automatic differentiation codes (the direct differentiation gives the tangent linear model, and the inverse differentiation provides the adjoint model) [47, 52].

Another issue is the relative ill-posedness of the problem when the model is non linear. The cost function $\mathscr{J}$ is hence non convex, and may have plenty of local minima. The optimization algorithm may then converge toward a local minimum and not the global minimum. For this reason, the choice of the initial guess is extremely important, because if it is located in the vicinity of the global minimum, one can expect a convergence toward the global minimum. Another solution is to increase the weight of the two last terms of $\mathscr{J}$ in (2), which correspond to two regularization terms with respect to the two control variables. This has to be done carefully because it can provide a physically incorrect solution: if P_0 and Q are too small, the regularization of $\mathscr{J}$ is indeed a penalization. But usually, these regularization terms are used to force the model to verify some additional physical constraints or/and to take into account some statistical information on model/observation/background errors.

3 Sequential Methods: Kalman Filter

In this section, we will study data assimilation methods based on the statistical estimation theory, in which the Kalman filtering theory is the primary framework. But the application of this theory encounters enormous difficulties due to the huge dimension of the state vector of the considered system. A further major difficulty is caused by its non linear nature. To deal with this, one usually linearizes the ordinary Kalman filter (KF) leading to the so-called extended Kalman filter (EKF) [22, 28, 31, 62]. We will also present the ensemble Kalman filter (EnKF), which allows one to get rid of too expensive computations of covariance matrices.

3.1 The Extended Kalman Filter

Consider a physical system described by

$$X(t_i) = \mathscr{M}(t_{i-1}, t_i)X(t_{i-1}) + U_i \tag{8}$$

where $\mathscr{M}(t_{i-1}, t_i)$ is an operator describing the system transition from time t_{i-1} to t_i, usually obtained from the integration of a partial differential system, and U_i is an unknown term of the model (it can be a noise term, used to modelize the unknown parameters of the model [17]). We suppose that at each time t_i, we have an observation vector $X_{obs}(t_i)$. Let us denote by ε_i the observation error, i.e. the difference between the observation vector and the corresponding state vector:

$$\varepsilon_i = X_{obs}(t_i) - C_i X(t_i), \tag{9}$$

where C_i is the observation operator at time t_i, mapping the state space into the space of observations. Q_i and R_i will be the covariance matrices of the model error (U_i) and the observation error (ε_i) respectively.

The extended Kalman filter operates sequentially: from an analysis state vector $X_a(t_{i-1})$ and its error covariance matrix $P^a(t_{i-1})$, it constructs the next analysis state vector $X_a(t_i)$ and $P^a(t_i)$ in two steps, a forecasting step and a correction step.

The first step is used to forecast the state at time t_i:

$$X^f(t_i) = M(t_{i-1}, t_i)X^a(t_{i-1}), \tag{10}$$

where $M(t_{i-1}, t_i)$ is the linearized model around $X^a(t_{i-1})$. The forecast error covariance matrix is then approximately

$$P^f(t_i) = M(t_{i-1}, t_i)P^a(t_i)M(t_{i-1}, t_i)^T + Q_i. \tag{11}$$

The second step is an analysis step, the newly available observation $X_{obs}(t_i)$ is used to correct the forecast state vector $X^f(t_i)$ in order to define a new analysis vector:

$$X^a(t_i) = X^f(t_i) + K_i(X_{obs}(t_i) - C_i X^f(t_i)), \tag{12}$$

where K_i is a gain matrix, called the Kalman matrix. The optimal gain is given by

$$K_i = P^f(t_i)C_i^T \left(C_i P^f(t_i) C_i^T + R_i\right)^{-1}. \tag{13}$$

The corresponding analysis error covariance matrix is given by

$$P^a(t_i) = P^f(t_i) - P^f(t_i)C_i^T \left(C_i P^f(t_i) C_i^T + R_i\right)^{-1} C_i P^f(t_i). \tag{14}$$

One main issue of the EKF is that the covariance matrices R_i, Q_i and P_0^a have to be known. Some statistical information can be obtained for observation error from the knowledge of the instrumental error variances in situations such as altimetric observations from satellites over the ocean, for which the error estimates have become fairly solidly established. But it is not clear how the correlations of these errors can be obtained. The covariances matrices Q_i and P_0^a are much more difficult to obtain, because very little is known concerning the true initial state of the system. These matrices are of very large dimension, and usually have a quite large number of independent elements. Is it really useful to estimate such a huge number of parameters? The theory for such equations (Eqs. (11) and (14)) state that for linear autonomous systems, even if P_0^a is poorly specified, one may hopefully still have a good approximation to P_i^a in the long term. The Kalman filter is optimal only if the covariance matrices R_i and Q_i are correctly specified. Thus, in practice, the Kalman filter is suboptimal.

3.2 Ensemble Kalman Filter

A main issue of the (extended) Kalman filter is the computational cost of propagating the covariance matrices in time. The dimension of P^f and P^a matrices is usually too large in geophysical problems, and there are several ways to avoid this point. One interesting approach is the ensemble Kalman filter (EnKF). The EnKF can be seen as a Monte Carlo approximation of the KF, avoiding evolving the full covariance matrix of the probability density function of the state vector [13, 23–25, 37].

As error statistics of background errors are not very well known, the idea is to generate a set of perturbed background states, with small perturbations around the background state with the same probability distribution. For $1 \leq j \leq M$, M being the size of the ensemble, we define the background ensemble members:

$$X_j(t_0) = X^b(t_0) + \varepsilon_j, \tag{15}$$

where X^b is the background state, and ε_j is the statistical perturbation, with a probability distribution consistent with the background error covariance matrix.

Then, we obtain an ensemble of forecast states with

$$X_j^f(t_i) = \mathscr{M}(t_{i-1}, t_i) X_j^a(t_{i-1}), \tag{16}$$

where we use the nonlinear model $\mathscr{M}(t_{i-1}, t_i)$. The correlation between these states gives some information about the forecast error statistics. The forecast error covariance matrix is then the actual covariance of the ensemble of states.

Then, the analysis state step is similar to the standard Kalman filter, with the computation of the Kalman gain matrix, and the correction is applied to each member:

$$X_j^a(t_i) = X_j^f(t_i) + K_i(X_{obs}(t_i) - C_i X_j^f(t_i)), \tag{17}$$

where K_i is the Kalman gain matrix computed with the actual covariance matrices of the ensemble. Then, the analysis error covariance matrix is computed from the ensemble of analysis states.

The EnKF is then *simply* a Kalman filter, applied to a discrete set (ensemble) of states (members). The covariance matrices are computed from this set, without using the linear or adjoint model. There are two main advantages: first, the computational cost is much lower, as there are no costly computations for the covariance matrices; and second, the covariance matrices represent the actual covariance statistics of the members in the ensemble.

4 Back and Forth Nudging Schemes: BFN and Diffusive BFN (DBFN)

The main issues of data assimilation for geophysical systems are the huge dimension of the control vectors (and hence of the covariance matrices) and the non linearities (most of the time, one has to linearize the model and/or some operators). The computation of the adjoint model is for example a difficult step in the variational algorithms. To get rid of these difficulties, we have very recently introduced a new algorithm, based on the nudging technique.

4.1 The Nudging Algorithm

The standard nudging algorithm consists in adding to the state equations a feedback term, which is proportional to the difference between the observation and its equivalent quantity computed by the resolution of the state equations. The model appears then as a weak constraint, and the nudging term forces the state variables to fit as well as possible to the observations.

Let us remind the model

$$\begin{cases} \dfrac{dX}{dt} = F(X, U), \quad 0 < t < T, \\ X(0) = V. \end{cases} \tag{18}$$

We still suppose that we have an observation $X_{obs}(t)$ of the state variable $X(t)$. The nudging algorithm simply gives

$$\begin{cases} \dfrac{dX}{dt} = F(X, U) + K(X_{obs} - CX), \quad 0 < t < T, \\ X(0) = V, \end{cases} \tag{19}$$

where C is still the observation operator, and K is the nudging matrix. It is quite easy to understand that if K is large enough, then the state vector transposed into the observation space (through the observation operator) $CX(t)$ will tend towards the observation vector $X_{obs}(t)$. In the linear case (where F and C are linear operators), the forward nudging method is nothing else than the Luenberger observer, also called asymptotic observer, where the operator K can be chosen so that the error goes to zero when time goes to infinity [44].

This algorithm was first used in meteorology [35], and then has been used with success in oceanography [63] and applied to a mesoscale model of the atmosphere [57]. Many results have also been carried out on the optimal determination of the nudging coefficients K [56, 64, 65].

The nudging algorithm is usually considered as a sequential data assimilation method. If one solves Eq. (19) with a numerical scheme, then it is equivalent with the following algorithm:

$$\begin{cases} X_n^f = X_{n-1} + dt \times F(X_{n-1}, U), \\ X_n = X_n^f + K_n(X_{obs}(t_n) - C_n X_n^f), \end{cases} \tag{20}$$

which is exactly the Kalman filter's algorithm. Then, if at any time the nudging matrix K is set in an optimal way, it is quite easy to see that K will be exactly the Kalman gain matrix. It is also possible to consider suboptimal K matrices, that still correct all variables, and not only the observed ones [10].

4.2 Backward Nudging

The backward nudging algorithm consists in solving the state equations of the model backwards in time, starting from the observation of the state of the system at the final instant. A nudging term, with the opposite sign compared to the standard nudging algorithm, is added to the state equations, and the final obtained state is in fact the initial state of the system [4, 7].

We now assume that we have a final condition in (18) instead of an initial condition. This leads to the following backward equation

$$\begin{cases} \dfrac{d\tilde{X}}{dt} = F(\tilde{X}, U), \quad T > t > 0, \\ \tilde{X}(T) = \tilde{V}. \end{cases} \tag{21}$$

If we apply nudging to this backward model with the opposite sign of the feedback term (in order to have a well posed problem), we obtain

$$\begin{cases} \dfrac{d\tilde{X}}{dt} = F(\tilde{X}, U) - K(X_{obs} - C\tilde{X}), \quad T > t > 0, \\ \tilde{X}(T) = \tilde{V}. \end{cases} \tag{22}$$

Once again, it is easy to see that if K is large enough, the vector $CX(t)$ will tend (through the observation operator) towards the observation vector $X_{obs}(t)$.

4.3 The BFN Algorithm

The back and forth nudging (BFN) algorithm consists in solving first the forward (standard) nudging equation, and then the direct system backwards in time with a feedback term. After resolution of this backward equation, one obtains an estimate of the initial state of the system. We repeat these forward and backward resolutions with the feedback terms until convergence of the algorithm [7].

The BFN algorithm is then the following:

$$\begin{cases} \dfrac{dX_k}{dt} = F(X_k, U) + K(X_{obs} - CX_k), \\ X_k(0) = \tilde{X}_{k-1}(0), \end{cases}$$
$$\begin{cases} \dfrac{d\tilde{X}_k}{dt} = F(\tilde{X}_k, U) - K'(X_{obs} - C\tilde{X}_k), \\ \tilde{X}_k(T) = X_k(T), \end{cases} \tag{23}$$

with $\tilde{X}_{-1}(0) = V$. Then, $X_0(0) = V$, and a resolution of the direct model gives $X_0(T)$ and hence $\tilde{X}_0(T)$. A resolution of the backward model provides $\tilde{X}_0(0)$, which is equal to $X_1(0)$, and so on.

This algorithm can be compared to the 4D-Var algorithm, which also consists in a sequence of forward and backward resolutions. In the BFN algorithm, even for nonlinear problems, it is useless to linearize the system and the backward system is not the adjoint equation but the direct system, with an extra feedback term that stabilizes the resolution of this ill-posed backward resolution.

The BFN algorithm has been tested successfully for the system of Lorenz equations, Burgers equation and a quasi-geostrophic ocean model in [8], for a shallow-water model in [5] and compared with a variational approach for all these models. It has been used to assimilate the wind data in a mesoscale model [14] and for the reconstruction of quantum states in [40].

4.4 DBFN: Diffusive Back and Forth Nudging Algorithm

In the framework of oceanographic and meteorological problems, there is usually no diffusion in the model equations. However, the numerical equations that are solved contain some diffusion terms in order to both stabilize the numerical integration (or the numerical scheme is set to be slightly diffusive) and model some subscale

turbulence processes. We can then separate the diffusion term from the rest of the model terms, and assume that the partial differential equations read:

$$\partial_t X = F(X) + \nu \Delta X, \quad 0 < t < T, \tag{24}$$

where F has no diffusive terms, ν is the diffusion coefficient, and we assume that the diffusion is a standard second-order Laplacian (note that it could be a fourth or sixth order derivative in some oceanographic models, but for clarity, we assume here that it is a Laplacian operator).

We introduce the D-BFN algorithm in this framework, for $k \geq 1$:

$$\begin{cases} \partial_t X_k = F(X_k) + \nu \Delta X_k + K(X_{obs} - C(X_k)), \\ X_k(0) = \tilde{X}_{k-1}(0), \quad 0 < t < T, \end{cases} \quad \begin{cases} \partial_t \tilde{X}_k = F(\tilde{X}_k) - \nu \Delta \tilde{X}_k - K'(X_{obs} - C(\tilde{X}_k)), \\ \tilde{X}_k(T) = X_k(T), \quad T > t > 0. \end{cases} \tag{25}$$

It is straightforward to see that the backward equation can be rewritten, using $t' = T - t$:

$$\partial_{t'} \tilde{X}_k = -F(\tilde{X}_k) + \nu \Delta \tilde{X}_k + K'(X_{obs} - C(\tilde{X}_k)), \quad \tilde{X}_k(t' = 0) = X_k(T), \tag{26}$$

where $\tilde{X}$ is evaluated at time t'. Then the backward equation is well-posed, with an initial condition and the same diffusion operator as in the forward equation. The diffusion term both takes into account the subscale processes and stabilizes the numerical backward integrations, and the feedback term still controls the trajectory with the observations.

The main interest of this new algorithm is that for many geophysical applications, the non diffusive part of the model is reversible, and the backward model is then stable. Moreover, the forward and backward equations are now consistent in the sense that they will be both diffusive in the same way (as if the numerical schemes were the same in forward and backward integrations), and only the non-diffusive part of the physical model is solved backwards. Note that in this case, it is reasonable to set $K' = K$.

The DBFN algorithm has been tested successfully for a linear transport equation in [9], for non-linear Burgers equation in [6], and for full primitive equations in [53].

4.5 Theoretical Considerations

The convergence of the BFN algorithm has been proved by Auroux and Blum in [7] for linear systems of ordinary differential equations and full observations, by Ramdani et al. [51] for reversible linear partial differential equations (wave and Schrödinger equations), by Donovan et al. [20] for the reconstruction of quantum states. In [11], the authors consider the BFN algorithm on transport equations. They show that for non viscous equations (both linear transport and Burgers), the

convergence of the algorithm holds under observability conditions. Convergence can also be proven for viscous linear transport equations under some strong hypothesis, but not for viscous Burgers' equation. Moreover, the authors show that the convergence rate is always exponential in time [11]. In [9], the authors prove the theoretical convergence of DBFN algorithm for linear transport equations.

Data Assimilation is the ensemble of techniques combining the mathematical information provided by the equations of the model and the physical information given by the observations in order to retrieve the state of a flow. In order to show that both BFN and DBFN algorithms achieve this double objective, let us give a formal explanation of the way these algorithms proceed.

If $K' = K$ and the forward and backward limit trajectory are equal, i.e. $\tilde{X}_\infty = X_\infty$, then taking the sum of the two equations in (23) shows that the limit trajectory X_∞ satisfies the model Eq. (18) (including possible model viscosity). Moreover, the difference between the two equations in (23) shows that the limit trajectory is solution of the following equation:

$$K(X_{obs} - C(X_\infty)) = 0. \tag{27}$$

Equation (27) shows that the limit trajectory perfectly fits the observations (through the observation operator, and the gain matrix).

In a similar way, for the DBFN algorithm, taking the sum of the two equations in (25) shows that the limit trajectory X_∞ satisfies the model equations without diffusion:

$$\partial_t X_\infty = F(X_\infty) \tag{28}$$

while taking the difference between the two same equations shows that X_∞ satisfies the Poisson equation:

$$\Delta X_\infty = -\frac{K}{\nu}(X_{obs} - C(X_\infty)) \tag{29}$$

which represents a smoothing process on the observations for which the degree of smoothness is given by the ratio $\frac{\nu}{K}$ [9]. Equation (29) corresponds, in the case where C is a matrix and $K = kC^T R^{-1}$, to the Euler equation of the minimization of the following cost function

$$J(X) = k\langle R^{-1}(X_{obs} - CX), (X_{obs} - CX)\rangle + \nu \int_\Omega \|\nabla X\|^2 \tag{30}$$

where the first term represents the quadratic difference to the observations and the second one is a first order Tikhonov regularisation term over the domain of resolution Ω. The vector X_∞, solution of (29), is the point where the minimum of this cost function is reached. This is a nice increment to the BFN algorithm, in which the limit trajectory fits the observations, while in the DBFN algorithm, the

limit trajectory is the result of a smoothing process on the observations (which are often very noisy).

It is shown in the next section (numerical results) that at convergence the forward and backward trajectories are very close, which justifies this qualitative justification of the algorithm.

5 Numerical Results on a 2D Shallow-Water Model

5.1 Description of the Model

The shallow water model (or Saint-Venant's equations) is a basic model, representing quite well the temporal evolution of geophysical flows. This model is usually considered for simple numerical experiments in oceanography, meteorology or hydrology. The shallow water equations are a set of three equations, describing the evolution of a two-dimensional horizontal flow. These equations are derived from a vertical integration of the three-dimensional fields, assuming the hydrostatic approximation, i.e. neglecting the vertical acceleration. There are several ways to write the shallow water equations, considering either the geopotential or height or pressure variables. We consider here the following configuration:

$$\begin{cases} \partial_t u - (f+\zeta)v + \partial_x B = \dfrac{\tau}{\rho_0 h} - ru + \nu\Delta u, \\ \partial_t v + (f+\zeta)u + \partial_y B = \dfrac{\tau}{\rho_0 h} - rv + \nu\Delta v, \\ \partial_t h + \partial_x(hu) + \partial_y(hv) = 0, \end{cases} \tag{31}$$

where the unknowns are u and v the horizontal components of the velocity, and h the geopotential height (see e.g. [1, 21]). The initial condition $(u(0), v(0), h(0))$ and no-slip lateral boundary conditions complete the system. The other parameters are the following:

- $\zeta = \partial_x v - \partial_y u$ is the relative vorticity;
- $B = g^* h + \dfrac{1}{2}(u^2 + v^2)$ is the Bernoulli potential;
- g^* is the reduced gravity;
- $f = f_0 + \beta y$ is the Coriolis parameter (in the β-plane approximation);
- $\tau = (\tau_x, \tau_y)$ is the forcing term of the model (e.g. the wind stress);
- ρ_0 is the water density; r is the friction coefficient; ν is the viscosity (or dissipation) coefficient.

We consider a numerical configuration in which the domain is a square of 2000×2000 km, with a rigid boundary, and no-slip boundary conditions. The time step is 1800 s (half an hour), and we consider an assimilation period $[0; T]$ of 720 time steps (i.e. 15 days). The forecast period is $[T; 4T]$, corresponding to 45 prediction

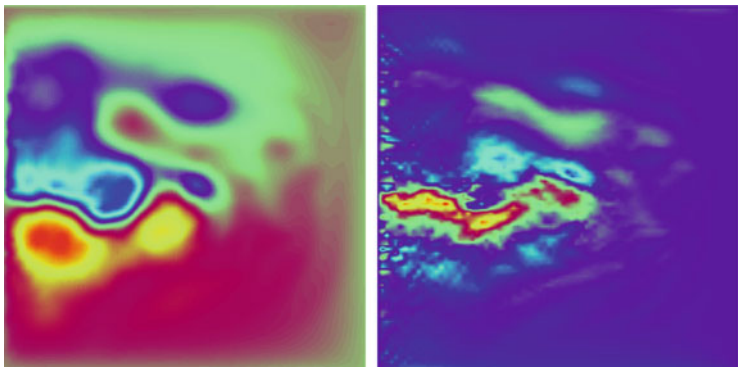

Fig. 1 Sea surface height h (*left*, in meters) and longitudinal velocity u (*right*, in ms^{-1}) of the ocean at a reference state. The ranges are the following: for h, from 250 m (*blue*) to 700 m (*red*), 500 being between *green* and *magenta*; for u, from -0.55 (*blue*) to 1 m per second (*red*), 0 being in *purple*

days. The spatial resolution is 25 km. The wind forcing is chosen constant in time, and set to a sine function which induces a standard double gyre circulation. This numerical model has been developed by the MOISE research team of INRIA Rhône-Alpes [21].

We briefly describe the numerical schemes used for the resolution of Eqs. (31) and we refer to [21] for more details. We consider a leap-frog method for time discretization of Eqs. (31), controlled by an Asselin time filter [3]. The equations are then discretized on an Arakawa C grid [2], with $N \times N$ points ($N = 81$ in our experiments): the velocity components u and v are defined at the center of the edges, and the height is defined at the center of the grid cells. Then, the vorticity and Bernoulli potential are computed at the nodes and center of the cells respectively.

The spin-up phase lasts nearly 6 years, starting from $u = v = 0$ and $h = 500$ meters, after which the model simulates a double-gyre wind-driven oceanic circulation. This approximate model reproduces quite well the surface circulation at mid-latitudes, including the jet stream and ocean boundary currents. In our experiments, the water depth varies from roughly 265–690 m, its mean being 500 m, and the maximum velocity (in the jet stream) is roughly 1.1 ms^{-1}, the mean velocity being 0.1 ms^{-1}. Figure 1 shows the height h and longitudinal velocity u at the reference state (true initial condition of the data assimilation period).

5.2 Experimental Setup

In all the following numerical experiments, we will consider twin experiments: a reference initial condition is set, and some data are extracted from the corresponding trajectory (see the model Eqs. (31) with $\nu = 0$). These data are then noised for some

experiments (and not for some others), and provided as observations to the 4D-VAR, BFN and DBFN data assimilation algorithms.

We assume that some observations h_{obs} of only the height h are available, every n_t time steps and every n_x grid points (in both longitudinal and transversal directions). We can then easily define an observation operator C, from the model space to the observation space, as an extraction of one every n_x values of the variable h in each direction. This operator is clearly linear, and allows us to compare a model solution with the observations. Unless some other values are given, the considered values of n_x and n_t are respectively 5 and 24. Unless said otherwise (for instance in Sect. 5.6), the observations are unnoisy.

In such a configuration, the model space (state variables (u, v, h)) is of dimension 19,683, and the observation space (data variable h_{obs}) is of dimension 289. The corresponding total number of observations all over the assimilation period is 8959.

5.3 Convergence of DBFN

Figure 2 shows the relative RMS (root mean square) difference between the height of the DBFN iterates and the true height, versus time, for the 10 first iterations. The RMS error is computed in the following way:

$$Relative\ RMS = \frac{\|h_{DBFN} - h_{true}\|}{\|h_{true}\|}, \tag{32}$$

where we use the standard L^2 norm over the space domain.

The height of the background state (used for the initialization of the algorithm) has a relative error of 45 % with the true height at initial time. The error decreases during the first forward resolution, down to 28 %, and again during the first backward iteration, reaching approximately 22 % after one DBFN iteration. After 5 iterations, the algorithm almost converged, and the relative RMS error is close to 10 % (slightly more at the beginning of the assimilation window, and slightly less at the end).

In the same figure, one can see the relative RMS errors for the two other variables of the model, the longitudinal and transversal components of the velocity. The error on u decreases from 61 % to less than 30 % in 5 iterations, and the error on v decreases from more than 80 % to approximately 40 %. So we can consider that in 5 (or 6) iterations, the DBFN algorithm converged. Note that there are no correction terms in the DBFN velocity equations, as there are no observations on the velocity, but the model coupling between the three variables allows the DBFN to correct all variables and not only the sea surface height.

Figure 3 shows the evolution of the relative RMS error of the DBFN solution, as a function of time. Between day 0 and day 15, the assimilation window, one can see the DBFN iterations with the decrease of the error during forward and backward iterations. Then, once the algorithm reaches convergence, we use the state identified

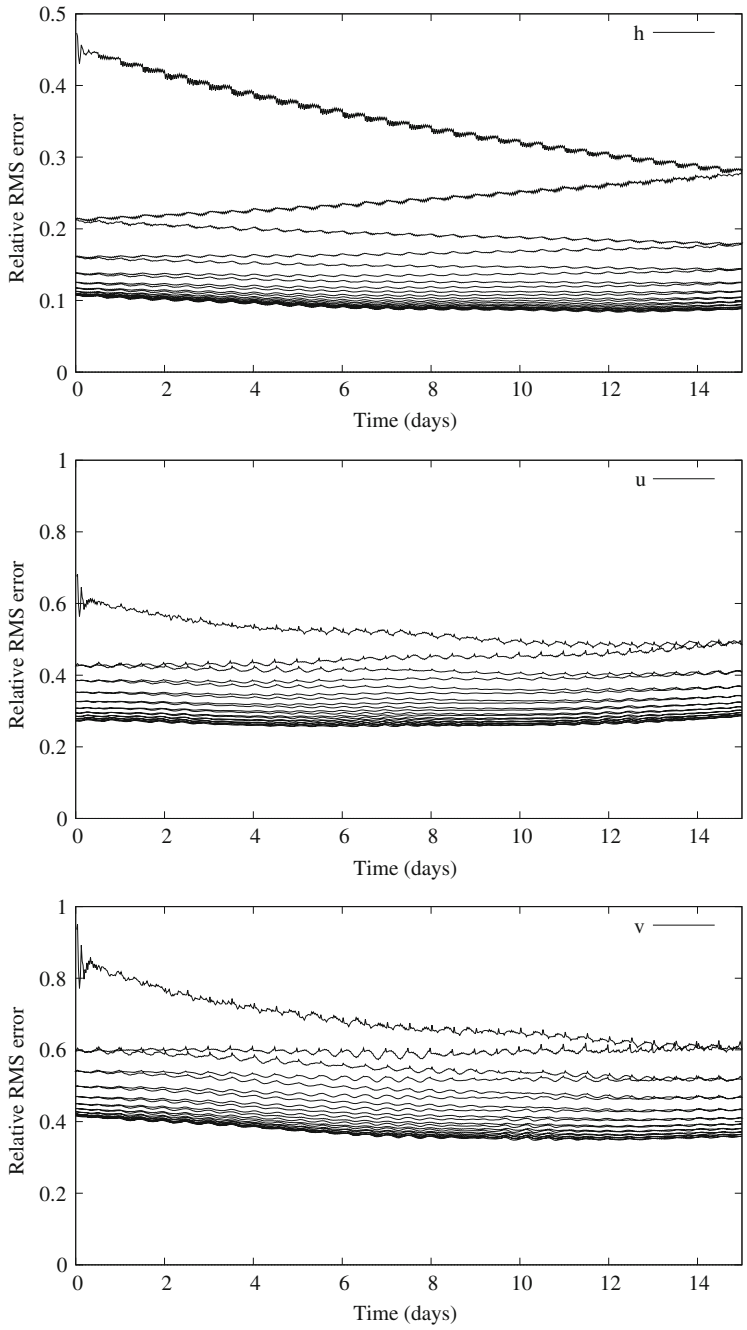

Fig. 2 Relative RMS error between the DBFN iterates (10 first iterations) and the true state, versus time, for the sea surface height h, and the velocity u and v

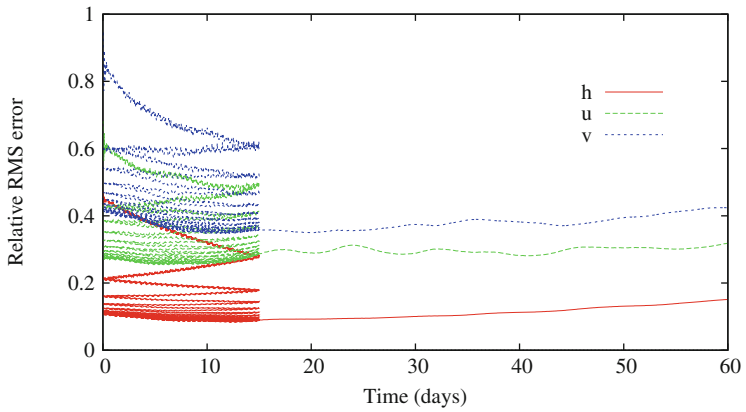

Fig. 3 Evolution of relative RMS error of the DBFN solution during the iterations (15 first days, corresponding to the assimilation window), and the forecast (from day 15 to day 60, corresponding to the forecast window)

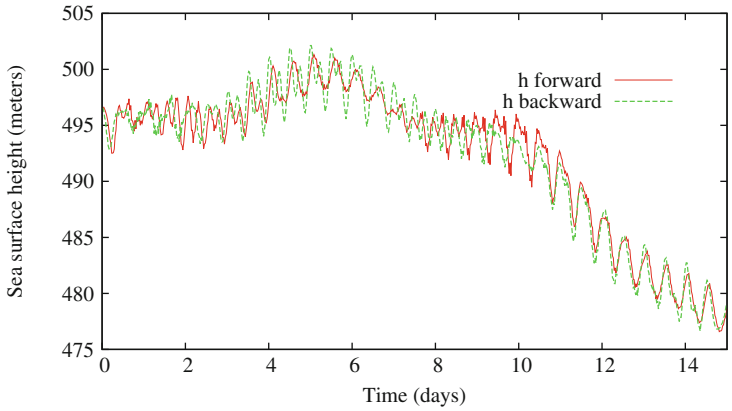

Fig. 4 Trajectory of the sea surface height at one particular point in the space domain during one back and forth iteration

by the DBFN algorithm as the initial condition for a forecast experiment: the model, without any feedback term, is used to propagate the solution in time, and predict the evolution of the solution.

Note that on the first 15 days, Fig. 3 shows the same results as in the three plots of Fig. 2. The interesting part is on the next 45 days, corresponding to the forecast period. The error remains quite stable and the final forecasted state has less than 20 % error on the sea surface height, and approximately 40 % on the velocity.

Figure 4 shows the evolution of the sea surface height at a given point in the domain (close to the main jet in the west part of the domain) during one back and forth iteration (iteration 5, when the DBFN converged). One can see that forward and backward solutions are very close, and this confirms that the algorithm

converged, as the new initial value (at time 0) after the backward iteration is similar to the previous initial value before the forward iteration.

5.4 Comparison Between BFN, DBFN and 4DVAR

Figure 5 shows the sea surface height h of several initial conditions (at time 0): on the top left, the initial condition identified by the BFN after 5 iterations (converged); on the top right, the initial condition identified after 5 iterations of DBFN (converged); on the bottom left, the initial condition identified by the 4DVAR after 50 iterations (converged); and on the bottom right, the true initial condition.

Note that we looked at the solution of the 4DVAR after 5 iterations. The computational costs of one (D)BFN iteration and of one 4DVAR iteration are more or less comparable, as they both consist in one resolution of the direct model, and one resolution of either the backward or the adjoint model. But the identified solution after 5 iterations of 4DVAR is not good at all and still too close to the

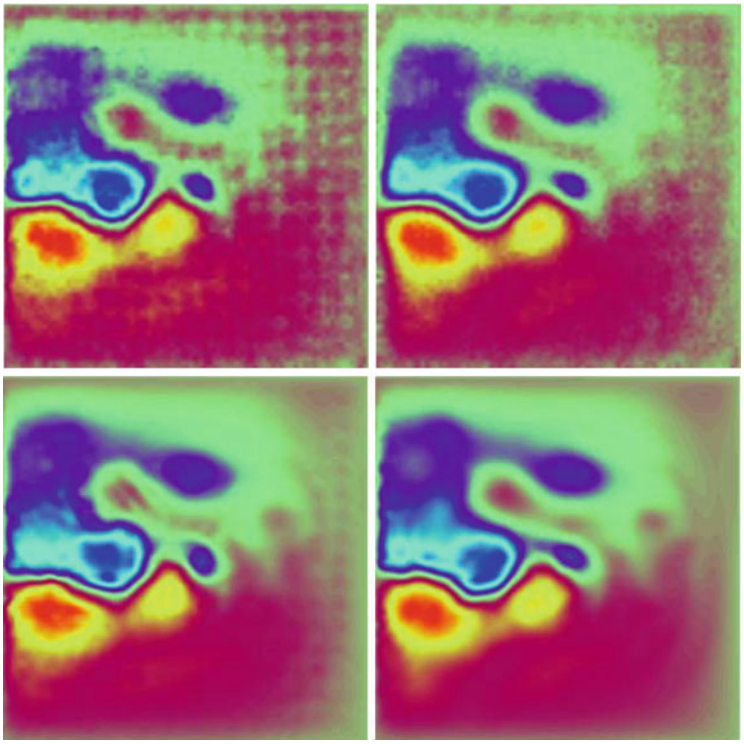

Fig. 5 Initial condition (sea surface height) identified by: BFN (5 iterations, converged), DBFN (5 iterations, converged), 4DVAR (50 iterations, converged), true solution

background state. So we decided to show only the solution after 50 iterations, when the algorithm converged.

We can see on Fig. 5 that the DBFN solution looks nicer than the BFN solution. In the BFN algorithm, the identified initial condition is the result of the last backward integration, where diffusion has the wrong sign. This partly explains why the BFN solution is not so smooth. The DBFN solution is better than the BFN one, as the diffusion helps for a better filtering of noise in the background state (and also a better filtering of the feedback term that is added at only some locations). All solutions show a pretty good agreement with the true state, with a good location of the Gulf stream and of the main vortices, but (D)BFN solutions are obtained in a much smaller time, as they require 10 times less iterations.

Figure 6 shows similar results as in Fig. 5, with one component u of the velocity instead of the sea surface height h. The color scale goes from -0.6 (blue) to 1.3 ms^{-1} (red). We can see that BFN, DBFN and 4DVAR identified solutions are close to the true state. But they all present some small oscillations and waves outside the Gulf stream zone. However, these small effects are negligible and do not induce big differences in the result. As previously said, it is noticeable that the velocity is

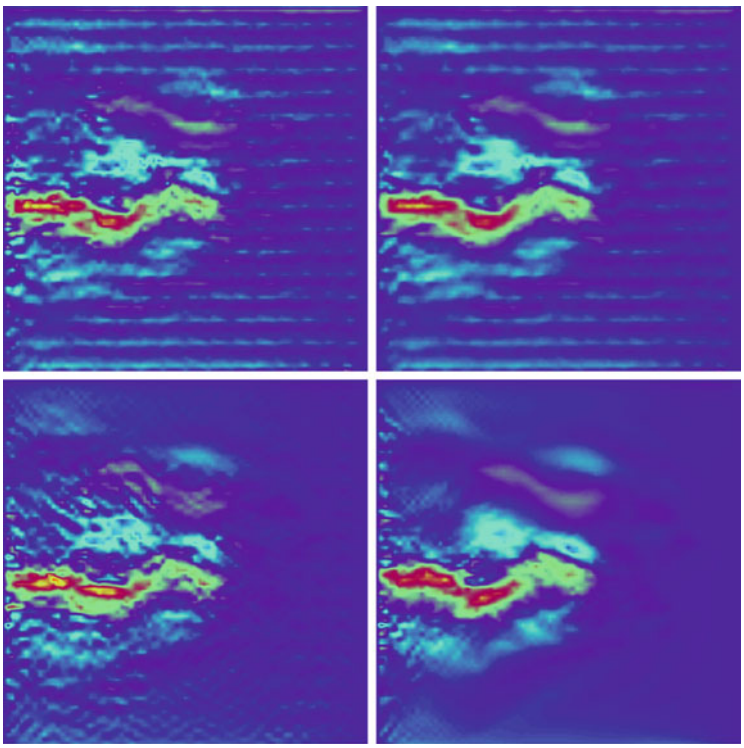

Fig. 6 Initial condition (longitudinal velocity u) identified by: BFN (5 iterations, converged), DBFN (5 iterations, converged), 4DVAR (50 iterations, converged), true solution

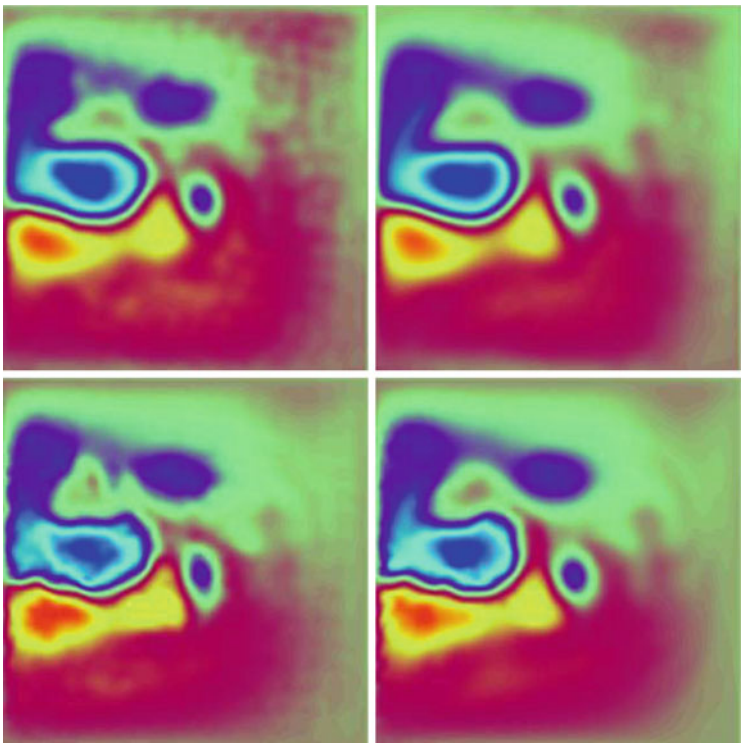

Fig. 7 Final state (sea surface height at the end of the forecast period) identified by: BFN (5 iterations, converged), DBFN (5 iterations, converged), 4DVAR (50 iterations, converged), true solution

quite well identified while there are no observations on velocity, only on sea surface height.

Finally, Fig. 7 shows similar results as in the previous figures, for the sea surface height h, at the end of the forecast period (60 days). From the identified initial conditions (Figs. 5 and 6), we used the model to propagate the solution in time, without any assimilation, up to 60 days.

We can see that all solutions are still close to the true state. The DBFN solution looks smoother than the BFN one (as the identified initial condition was also smoother). The DBFN solution has been obtained with a much smaller computational cost (10 times less) than the 4DVAR solution, as it required only 5 iterations instead of 50. As for the initial condition, if we stop the 4DVAR after 5 iterations, the solution is not good at all.

We can conclude that the DBFN gives smoother solutions than the BFN, which is interesting for noise filtering. This is indeed due to the reversed diffusion in the backward resolutions. And the DBFN gives similar results to the 4DVAR, but much faster as it requires much less iterations to achieve convergence.

5.5 Sensitivity of the BFN and DBFN with Respect to the Diffusion Coefficient

Table 1 gives the relative RMS error of the solution identified by the BFN and DBFN, for several viscosity coefficients. For each case, the error is measured at both time $t = 0$ (initial condition identified by the algorithm at convergence), and at time $t = 60$ days (corresponding final forecast obtained from the integration of the model).

When the diffusion is very small, BFN and DBFN give very similar results. As the diffusion has a very limited role in this case, BFN and DBFN algorithms are very close (and equal in the limit case where $\nu = 0$). When the diffusion increases, the DBFN algorithms gives better results, particularly on the velocity components. This is due to the fact that the DBFN has a better filtering role than the BFN algorithm, as there is also diffusion (with the good sign) in the backward integrations. As we only have observations on a sparse grid, and as the feedback term is added only at observation locations, the filtering effect helps the feedback to be smoothed over neighboring grid points. Finally, if the diffusion is very high, both methods give degraded results: the BFN has strong instabilities in the backward resolutions, and the DBFN has a too strong diffusive effect. However, the errors on the forecast remain very reasonable.

Table 1 Relative RMS error (in %) of the solution identified by the (D)BFN for several viscosity coefficients (in $m^2.s^{-1}$): error on the identified initial condition after 5 iterations (converged), and on the corresponding final forecast at time $t = 60$ days

		h init	u init	v init	h forecast	u forecast	v forecast
$\nu = 0.5$	BFN	14.9	45.6	65.1	22.6	46.7	66.5
	DBFN	14.7	44.6	63.6	21.9	45.5	64.0
$\nu = 1.5$	BFN	14.9	45.6	65.2	22.5	46.4	65.8
	DBFN	14.3	42.8	61.1	20.9	43.6	59.6
$\nu = 5$	BFN	14.9	45.7	65.4	22.4	45.6	64.1
	DBFN	13.0	38.8	54.6	18.5	37.3	48.7
$\nu = 15$	BFN	14.9	46.0	65.9	21.9	43.6	59.9
	DBFN	10.9	32.8	45.4	16.4	33.2	44.6
$\nu = 50$	BFN	15.0	47.7	68.3	21.0	39.2	52.3
	DBFN	10.8	27.1	41.6	15.1	31.9	42.4
$\nu = 150$	BFN	15.6	57.7	82.0	20.1	36.2	46.1
	DBFN	14.6	30.1	49.7	16.8	36.9	46.4

Table 2 Relative RMS error (in %) of the solution identified by the (D)BFN for several observation noise levels: error on the identified initial condition after 5 iterations (converged), and on the corresponding final forecast at time $t = 60$ days

		h init	u init	v init	h forecast	u forecast	v forecast
Noise = 5 %	BFN	15.3	49.6	69.6	20.7	38.6	51.5
	DBFN	11.2	28.0	42.6	15.2	32.0	42.8
Noise = 10 %	BFN	16.1	53.0	74.4	22.0	39.7	51.5
	DBFN	12.3	30.6	46.0	15.4	32.2	43.4
Noise = 20 %	BFN	19.3	64.1	91.3	23.9	42.7	55.8
	DBFN	15.0	39.7	58.2	15.7	33.2	45.0
Noise = 40 %	BFN	29.6	98.1	136.5	31.3	58.2	74.5
	DBFN	23.3	65.6	93.7	17.3	36.8	50.1

5.6 Sensitivity of the BFN and DBFN with Respect to Observation Noise

In this section, we now consider noisy observations. Table 2 gives the relative RMS error of the solution identified by the BFN and DBFN, for several observation noise levels. In this experiment, we set the diffusion coefficient to $5\,\mathrm{m}^2.\mathrm{s}^{-1}$. We added Gaussian white noise to the observations, with several relative levels. For each case, the error is measured at both time $t = 0$ (initial condition identified by the algorithm at convergence), and at time $t = 60$ days (corresponding final forecast obtained from the integration of the model).

When the noise level increases, of course all initial conditions identified are degraded, but the DBFN is less sensitive to noise on the observations, particularly on the velocity components. For all noise levels, the DBFN gives better results than the BFN on all components. If we look at the results on the final forecast state, we can see that the DBFN is much less sensitive to observation noise than the BFN, and it gives much better results for high levels (though realistic) of noise.

6 Numerical Results on a 3D Primitive Equation Ocean Model

6.1 Description of the Model

In this section the BFN and DBFN are tested on a Primitive Equation (PE) ocean model. The ocean model used in this study is the ocean component of NEMO (Nucleus for European Modeling of the Ocean; [46]). This model is able to represent a wide range of ocean motions, from basin scale up to regional scale. Currently, it has been used in operational mode at the French Mercator Océan project (http://www.mercator-ocean.fr).

The model solves six prognostic equations for zonal velocity (u) and meridional velocity (v), pressure (P), sea surface height (η), temperature (T), salinity (S) and finally a state equation which connects the mass field to the dynamical field. Two physically-based hypotheses are made: Boussinesq and incompressible ocean, and the hydrostatic approximation.

The conservation of momentum is expressed as the Navier-Stokes equations for a fluid element located at (x, y, z) on the surface of our rotating planet and moving at velocity (u, v, w) relative to that surface:

$$\frac{DU}{Dt} = -\frac{\nabla P}{\rho} + g - 2\Omega \times U + D + F, \tag{33}$$

where t is the time, $U = U_h + w\mathbf{k}$ is the velocity vector composed by its horizontal components $U_h = (u, v)$ and vertical component $w\mathbf{k}$, where $\mathbf{k}$ is the upward unit vector perpendicular to the earth surface, g is the gravitational acceleration, Ω is the rotational vector, ρ is the density, and D and F are the dissipation and forcing terms. The operator $\dfrac{D}{Dt}$ is the total derivative which includes time and local variations. It is given by:

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + U.\nabla.$$

The conservation of mass is given by the continuity equation:

$$\frac{\partial \rho}{\partial t} + \nabla (\rho U) = 0 \, . \tag{34}$$

The Boussinesq approximation considers that density variations are much smaller than the mean density

$$\frac{\delta\rho}{\rho} \ll 1 \, .$$

Therefore, the ocean density is considered constant, i.e. a mean density ρ_0 replaces ρ in the equations, except in the buoyancy term for which the density is multiplied by the gravity. Considering the aspect ratio $\dfrac{\tilde{H}}{\tilde{L}} \ll 1$, where $\tilde{L}$ is the horizontal scale and $\tilde{H}$ is the vertical scale, the equation for the evolution of w is resumed to the hydrostatic equilibrium:

$$\frac{\partial P}{\partial z} = -\rho g \tag{35}$$

With this approximation the acceleration term, $\frac{\partial w}{\partial t}$, is neglected implying a misrepresentation of gravitational flows as well as of vertical convection processes. Indeed, the vertical component of the velocity field is a diagnostic variable calculated thanks to the continuity equation under the Boussinesq approximation:

$$\frac{\partial w}{\partial z} = -\nabla_h . U, \tag{36}$$

where ∇_h is the restriction of the gradient operator to the horizontal plane.

The equation for the evolution of the free surface, i.e. for the sea surface height η, is derived from the surface kinetic boundary condition and may be written as:

$$\begin{aligned} \frac{\partial \eta}{\partial t} &= -\nabla_h (D\bar{U}_h) \\ \bar{U}_h &= \frac{1}{D}\int_{-D}^{0} U_h dz, \end{aligned} \tag{37}$$

where D is the water depth.

The equations for the conservation of salt and potential temperature are derived from the first thermodynamical law. The final equations can be written as:

$$\frac{\partial T}{\partial t} = -\nabla (UT) + D^T + F^T \tag{38}$$

$$\frac{\partial S}{\partial t} = -\nabla (US) + D^S + F^S . \tag{39}$$

The system is closed by a state equation linking density, temperature, salinity and pressure:

$$\rho = \rho(T, S, P) . \tag{40}$$

The system composed by Eqs. (33),(37),(38), (39) and (40) are discretized on a spherical mesh using a traditional centered second-order finite difference approximation. The non-diffusive terms evolves in time using a leap-frog method, controlled by an Asselin time filter [3], and the diffusion terms are discretized using a forward Euler scheme, for the horizontal diffusion, and an implicity backward Euler, for the vertical diffusion terms, in order to assure numerical stability.

Special attention has been given to the homogeneity of the solution in the three space directions. The arrangement of variables is the same in all directions. It consists of cells centered on scalar points (T, S, P, ρ) with vector points (u, v, w) defined in the center of each face of the cells. This is the well-known C grid in Arakawa's classification generalized to the three dimensional space. The relative and planetary vorticity, ζ and f , are defined in the center of each vertical edge and the barotropic stream function ϕ is defined at the horizontal points overlying the ζ

and f points. More details on the model formulation and on the numerical methods are given by [46].

6.2 Model Configuration

NEMO model is configured to simulate the double gyre circulation as the SW model used in the Sect. 5. Two experiments using different horizontal resolution are configured: one at 9.25 km resolution, which is used as representing the true ocean state, and another one with 25.5 km resolution that will assimilate the observations from the higher resolution model. For both configurations 11 vertical levels are considered with resolution ranging from 100 m near the upper surface up to 500 m near the bottom. The bottom topography is flat and the lateral boundaries are closed and frictionless. The model depth is set to $H = 5000$ m, in this case the variable h described in the Sect. 5 corresponds to $h = H + \eta$. The only forcing term considered is a constant wind stress of the form $\tau = \left(\tau_0 cos\left(\frac{2\pi(y-y_0)}{L}\right), 0\right)$, where $L = 2000$ km and $\tau_0 = 0.1\,\mathrm{N/m^2}$. The diffusion and viscosity terms D, D^T and D^S in the Eqs. (33), (38) and (39) are decomposed into a horizontal component, which is modeled by a bilaplacian operator $\left(D_h = \nu_h \nabla_h^4\right)$, and a vertical component, which is modeled by a laplacian operator $\left(D_v = \nu_v \frac{\partial^2}{\partial z^2}\right)$. They all use constant coefficients in time and space. Table 3 summarizes the characteristics of each experiment in terms of number of grid points, time step and horizontal and vertical diffusion/viscosity coefficients.

The initial condition consists of a homogeneous salinity field of 35 psu and a temperature field created to provide a stratification which has a first baroclinic deformation radius of 44.7 km. Velocity and pressure fields are initially set to zero. For both experiments the model was integrated for 70 years, in order to reach the statistical steady state. The final condition of the simulation at 25.5 km resolution is used as the first guess in the assimilation experiments. The 9.25 km resolution experiment is further integrated for 1 year to generate the true trajectory.

Table 3 Characteristics of each experiment in terms of horizontal spatial resolution, number of grid points (N_x, N_y, N_z), time step and horizontal (ν_h) and vertical (ν_v) diffusion/viscosity coefficients

Resolution	N_x	N_y	N_z	time step (s)	ν_h ($\mathrm{m^4/s}$)	ν_v ($\mathrm{m^2/s}$)
9.25 km	363	243	11	300	-8×10^9	1.2×10^{-5}
25.5 km	121	81	11	900	-8×10^{10}	1.2×10^{-5}

6.3 Experimental Setup

6.3.1 Model Aspects

The assimilation period $[0; T]$ is made of 960 time steps (i.e. 10 days). Each dataset (see Sect. 6.3.2) is assimilated with the BFN and DBFN using a reduced horizontal diffusion coefficient, $\nu_h = -8 \times 10^8\,\mathrm{m}^4/\mathrm{s}$, which is different from the diffusion coefficient used to generate the background field (see Table 3). The vertical diffusion coefficient is the same as in Table 3.

6.3.2 Observation Network

The observation network is composed by observations of the Sea Surface Height (η) sampled in order to mimic the spatial sample density of the Jason-1 satellite, but with higher time sampling frequency (here we considered the satellite period as 1 day in contrast to the 10 day period of Jason-1), complete daily fields of Sea Surface Temperature (SST), which may be seen as the merged products of SST derived from satellite observations, and vertical profiles of temperature and salinity available every $3° \times 3°$ and every day, similar to the ARGO buoys sampling. This observation network is quite close to a real ocean observation system in terms of number of available observations within the given model domain.

The true state, used to generate the observations, comes from a much higher resolution experiment ($\Delta x = 9.25\,\mathrm{km}$). As a consequence, the observations contain two types of error: representation errors that simulate the fact that the observations are sampled from a system containing process and scales that are not modeled by the model used in the assimilation step, and white Gaussian errors that simulate instrumental errors.

For each DBFN and BFN configuration two datasets are assimilated: one without white noise and another one with a noise level of 100 %.

6.4 Convergence of DBFN

Our reference experiment assimilates observations without noise with the DBFN. Figure 8 shows the evolution of the relative error during the forward and backward integrations for the SSH (η) and for the zonal velocity, which is a non observed variable. The DBFN needs 108 iterations to converge, which is much larger than the number of iterations needed by the DBFN to converge in the SW experiments. This is due to the fact that the observation information takes time to be propagated downwards to the deep ocean, and for the temperature increments, calculated from the vertical profiles, to be smoothed.

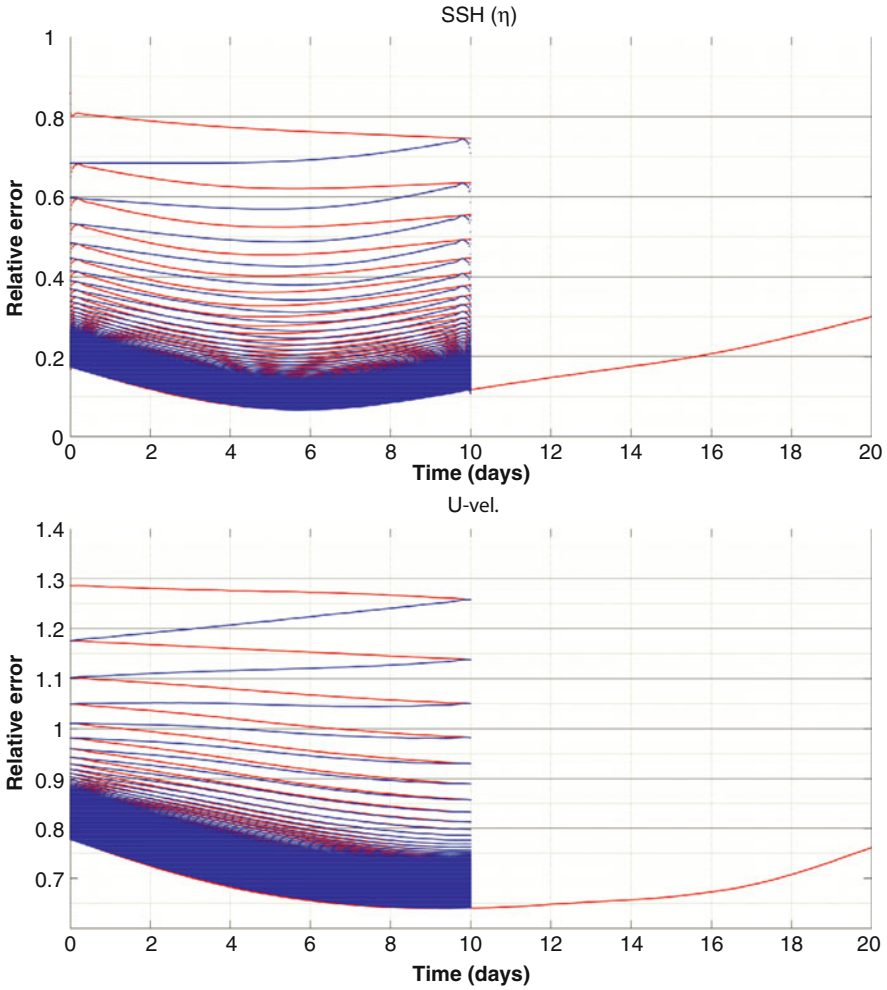

Fig. 8 Evolution of the relative error during the forward and backward integrations for the (*top*) SSH and (*bottom*) zonal velocity for the experiment assimilating observations without noise with the DBFN. *Red curves* represent the forward integration and *blue curves* represent the backward integration. The *red curve* for the days 10–20 represents a forecast initialized with the initial condition identified by the DBFN

6.5 Comparison Between DBFN and BFN

Table 4 summarizes the relative error on the initial condition obtained for both algorithms in the case of the assimilation of observations without random noise and with 100 % of noise. The DBFN produces the best results for both cases: assimilating observations with and without noise. Indeed, the DBFN performance

Table 4 Summary of the relative error on the initial condition obtained after 108 iterations, which is the number of iterations needed by the DBFN-0 to converge, for both algorithms in the case of the assimilation of observations without random noise (BFN-0 and DBFN-0) and with 100 % of noise (BFN-1 and DBFN-1)

	SSH	u	v	T	S
BFN-0	0.1790	0.8226	0.7130	0.1195	0.0143
DBFN-0	0.1754	0.7790	0.6440	0.1193	0.0143
BFN-1	0.1926	0.8928	0.7932	0.1195	0.0143
DBFN-1	0.1872	0.8053	0.6710	0.1194	0.0143

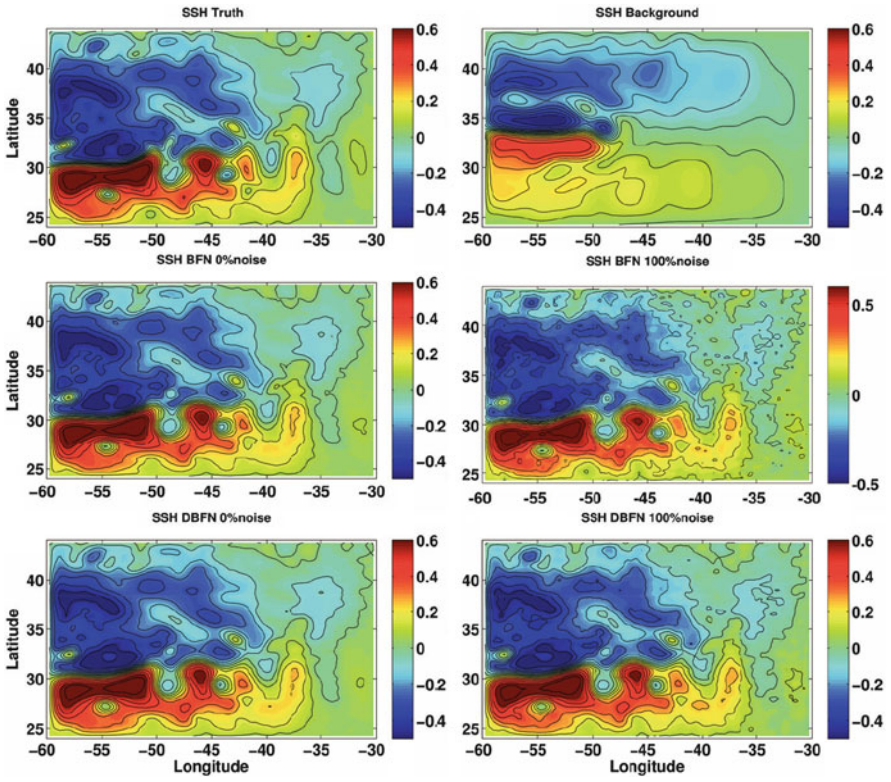

Fig. 9 SSH fields for the true state (*top left*) and the first guess (*top right*), and the analyzed state produced by the BFN (*middle panels*) and DBFN (*bottom panels*) assimilating non perturbed (*left panels*) and perturbed (*right panels*) observations

is improved relative to the BFN performance when the observations are noisy. This is the same result observed for the SW model and is related to the diffusive aspect of the algorithm which helps the method to filter out the observation noise.

The Figs. 9 and 10 show the SSH and surface zonal velocity fields for the first guess, the true state and the analyzed states produced by the BFN and DBFN assimilating perturbed and non perturbed observations. The first important remark

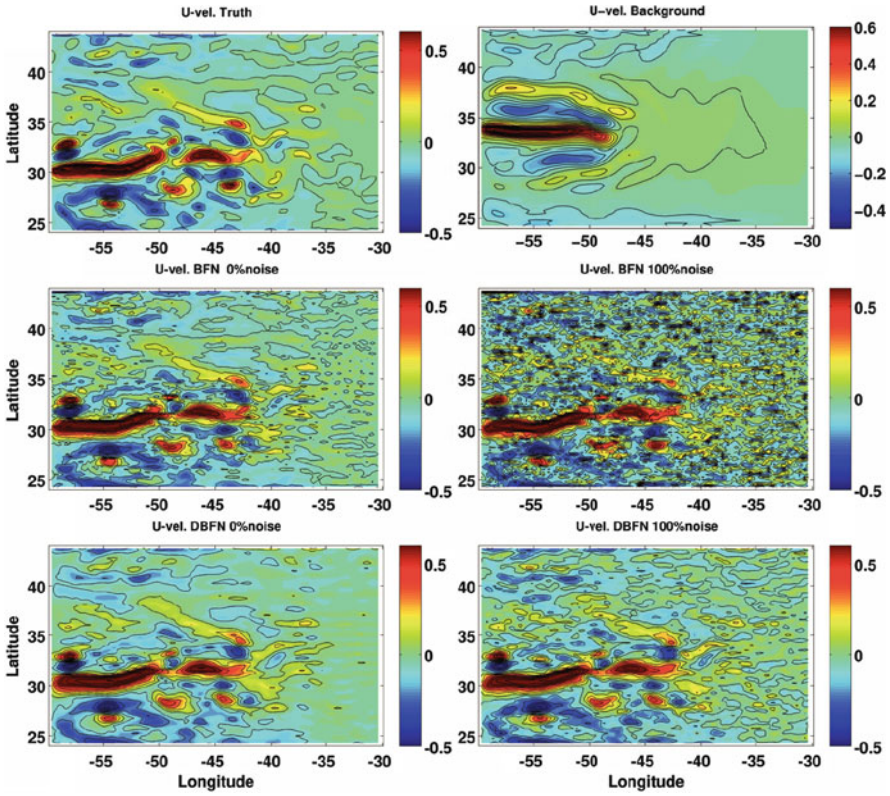

Fig. 10 Surface zonal velocity fields for the true state (*top left*) and the first guess (*top right*), and the analyzed state produced by the BFN (*middle panels*) and DBFN (*bottom panels*) assimilating non perturbed (*left panels*) and perturbed (*right panels*) observations

is that the front position in the true state is displaced southward with respect to the background. This is observed for the entire free run and is the effect of non-linear interactions between the small scales, not modeled by the 25.5 km resolution, and the large scale. All experiments were able to replace the front at the right position. Moreover, all experiments were able to reproduce the presence of eddies, especially the three eddies observed southward of the main front.

The results also show that even in the absence of noise in the observations, the velocity field produced by the BFN is quite noisy due to the diffusion in the backward integration. In the presence of noise, the obtained BFN solution has an increased level of noise. The same is also valid for the DBFN, but the observation noise has a much smaller impact in this case. In other words, the DBFN has a double advantage: first it ensures the stability of the backward integration, and second it is more efficient in filtering out the observation errors.

6.6 Concluding Remarks

In this section we have seen that the BFN and the DBFN can be used to assimilate oceanic observations into a PE ocean model. Indeed, for a PE ocean model, the DBFN should be preferably used due to its diffusive aspect that has a double role: stabilize the backward integration and filter out the noise on the observations.

7 Conclusion

The diffusive back and forth nudging algorithm has a very easy implementation, which does not impose any linearization of the model, which does not require neither the construction of an adjoint model, nor an optimization procedure as in the 4DVAR method. Moreover it is a very fast algorithm, which converges in less iterations than the 4DVAR. Therefore it has a lower computational and memory cost than the variational method. The progress of the DBFN, compared to the BFN method, is a better smoothing of the noise on the observations, which results from a more diffusive process (forward and backward). The distance from the reconstructed solution to the true one is also smaller with the DBFN algorithm than the one obtained with the BFN method. These characteristics have been observed, both on 2D shallow water model and on the 3D ocean primitive equation model.

The perspective is to achieve an optimization of the gain matrix, which can be the same in the forward and in the backward resolution, which is not the case in the BFN method. And the final goal is of course to apply this method to real data in order to confirm the conclusions obtained here on realistic models and with synthetic data corresponding to realistic observation systems.

Another interesting point would be to obtain a proof of the convergence of the method on these sophisticated non-linear geophysical models.

Acknowledgements The authors would like to thank Yann Brenier (CMLS, École Polytechnique), Emmanuel Cosme (MEOM, LGGE Grenoble) and Maëlle Nodet (MOISE, LJK Grenoble) for fruitful discussions. This work was supported by a CNRS-INSU LEFE-MANU grant.

References

1. Adcroft, A., Marshall, D.: How slippery are piecewise-constant coastlines in numerical ocean models? Tellus **50**(1), 95–108 (1998)
2. Arakawa, A., Lamb, V.: Computational design of the basic dynamical processes of the UCLA general circulation model. Methods Comput. Phys. **17**, 174–267 (1977)
3. Asselin, R.: Frequency filter for time integrations. Mon. Weather Rev. **100**, 487–490 (1972)
4. Auroux, D.: Étude de différentes méthodes d'assimilation de données pour l'environnement. PhD thesis, University of Nice Sophia-Antipolis, France (2003)

5. Auroux, D.: The back and forth nudging algorithm applied to a shallow water model, comparison and hybridization with the 4D-VAR. Int. J. Numer. Methods Fluids **61**(8), 911–929 (2009)
6. Auroux, D., Bansart, P., Blum, J.: An evolution of the Back and Forth Nudging for geophysical data assimila tion: application to burgers equation and comparisons. Inverse Probl. Sci. Eng. **21**(3), 399–419 (2013)
7. Auroux, D., Blum, J.: Back and forth nudging algorithm for data assimilation problems. C. R. Acad. Sci. Paris Ser. I **340**, 873–878 (2005)
8. Auroux, D., Blum, J.: A nudging-based data assimilation method for oceanographic problems: the back and forth nudging (BFN) algorithm. Nonlinear Proc. Geophys. **15**, 305–319 (2008)
9. Auroux, D., Blum, J., Nodet, M.: Diffusive Back and Forth Nudging algorithm for data assimilation. C. R. Acad. Sci. Paris Ser. I **349**(15–16), 849–854 (2011)
10. Auroux, D., Bonnabel, S.: Symmetry-based observers for some water-tank problems. IEEE Trans. Autom. Control **56**(5), 1046–1058 (2011)
11. Auroux, D., Nodet, M.: The Back and Forth nudging algorithm for data assimilation problems: theoretical results on transport equations. ESAIM Control Optim. Calc. Var. **18**(2), 318–342 (2012)
12. Bennett, A.F.: Inverse Modeling of the Ocean and Atmosphere. Cambridge University Press, Cambridge (2002)
13. Blum, J., Le Dimet, F.-X., Navon, I.M.: Data assimilation for geophysical fluids. In: Temam, R., Tribbia, J.J. (eds.) Computational Methods for the Atmosphere and the Oceans. Handbook of Numerical Analysis, vol. XIV, pp. 385–441. Elsevier, Amsterdam/London (2009)
14. Boilley, A., Mahfouf, J.-F.: Assimilation of low-level wind in a high resolution mesoscale model using the Back and Forth nudging algorithm. Tellus A **64**, 18697 (2012)
15. Broyden, C.G.: A new double-rank minimization algorithm. Not. Am. Math. Soc. **16**, 670 (1969)
16. Cane, M.A., Kaplan, A., Miller, R.N., Tang, B., Hackert, E.C., Busalacchi, A.J.: Mapping tropical Pacific Sea level: data assimilation via a reduced state Kalman filter. J. Geophys. Res. **101**(C10), 22599–22617 (1996)
17. Carrassi, A., Vannitsem, S.: Deterministic treatment of model error in geophysical data assimilation. In: Ancona, F., et al. (eds.) Mathematical Paradigms of Climate Science. Springer (2016)
18. Courtier, P., Talagrand, O.: Variational assimilation of meteorological observations with the adjoint equations Part 2. Numerical results. Q. J. R. Meteor. Soc. **113**, 1329–1347 (1987)
19. Le Dimet, F.-X., Talagrand, O.: Variational algorithms for analysis and assimilation of meteorogical observations: theoretical aspects. Tellus **38A**, 97–110 (1986)
20. Donovan, A., Mirrahimi, M., Rouchon, P.: Back and Forth nudging for quantum state reconstruction. In: 4th International Symposium Communications Control Signal Proceedings, Limassol, pp. 1–5, Mar 2010
21. Durbiano, S.: Vecteurs caractéristiques de modèles océaniques pour la réduction d'ordre en assimilation de données. PhD thesis, University of Grenoble I, France (2001)
22. Evensen, G.: Using the extended Kalman filter with a multilayer quasi-geostrophic ocean model. J. Geophys. Res. **97**, 17905–17924 (1992)
23. Evensen, G.: Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. J. Geophys. Res. **99**(C5), 10143–10162 (1994)
24. Evensen, G.: The ensemble Kalman filter: theoretical formulation and practical implementation. Ocean Dyn **53**, 343–367 (2003)
25. Evensen, G.: Data Assimilation: The Ensemble Kalman Filter. Springer, Berlin/Heidelberg (2009)
26. Fukumori, I.: Assimilation of topex sea level measurements with a reduced-gravity, shallow water model of the tropical Pacific Ocean. J. Geophys. Res. **100**(C12), 25027–25039 (1995)
27. Fukumori, I., Benveniste, J., Wunsch, C., Haidvogel, D.B.: Assimilation of sea surface topography into an ocean circulation model using a steady state smoother. J. Phys. Oceanogr. **23**, 1831–1855 (1993)

28. Gauthier, P., Courtier, P., Moll, P.: Assimilation of simulated wind lidar data with a Kalman filter. Mon. Weather Rev. **121**, 1803–1820 (1993)
29. Gelb, A.: Applied Optimal Estimation. MIT, Cambridge (1974)
30. Ghil, M.: Meteorological data assimilation for oceanographers. Part 1: description and theoretical framework. Dyn. Atmos. Oceans **13**, 171–218 (1989)
31. Ghil, M., Cohn, S.E., Dalcher, A.: Sequential estimation, data assimilation and initialization. In: Williamson, D. (ed.) The Interaction Between Objective Analysis and Initialization. Publication in Meteorology, vol. 127. McGill University, Montréal (1982)
32. Ghil, M., Manalotte-Rizzoli, P.: Data assimilation in meteorology and oceanography. Adv. Geophys. **23**, 141–265 (1991)
33. Gilbert, J.-Ch., Lemaréchal, C.: Some numerical experiments with variable storage quasi-Newton algorithms. Math. Prog. **45**, 407–435 (1989)
34. Gourdeau, L., Arnault, S., Ménard, Y., Merle, J.: Geosat sea-level assimilation in a tropical Atlantic model using Kalman filter. Ocean. Acta **15**, 567–574 (1992)
35. Hoke, J., Anthes, R.A.: The initialization of numerical models by a dynamic initialization technique. Mon. Weaver Rev. **104**, 1551–1556 (1976)
36. Holland, W.R.: The role of mesoscale eddies in the general circulation of the ocean. J. Phys. Ocean **8**(3), 363–392 (1978)
37. Houtekamer, P., Mitchell, H.: Data assimilation using an ensemble Kalman filter technique. Mon. Weather Rev. **126**, 796–811 (1998)
38. Jazwinski, A.H.: Stochastic Processes and Filtering Theory. Academic, New York (1970)
39. Kalnay, E.: Atmospheric Modeling, Data Assimilation and Predictability. Cambridge University Press, Cambridge/New York (2003)
40. Leghtas, Z., Mirrahimi, M., Rouchon, P.: Observer-based quantum state estimation by continuous weak measurement. In: American Control Conference (ACC), pp. 4334–4339 (2011)
41. Lewis, J.M., Derber, J.C.: The use of adjoint equations to solve a variational adjustment problem with convective constraints. Tellus **37A**, 309–322 (1985)
42. Lions, J.L.: Contrôle optimal de systèmes gouvernés par des équations aux dérivées partielles. Dunod (1968)
43. Liu, D.C., Nocedal, J.: On the limited memory BFGS method for large scale optimization. Math. Prog. **45**, 503–528 (1989)
44. Luenberger, D.: Observers for multivariable systems. IEEE Trans. Autom. Control **11**, 190–197 (1966)
45. Luong, B., Blum, J., Verron, J.: A variational method for the resolution of a data assimilation problem in oceanography. Inverse Probl. **14**, 979–997 (1998)
46. Madec, G.: NEMO ocean engine. Note du Pôle de modélisation, Institut Pierre-Simon Laplace (IPSL) No 27, France (2008)
47. Mohammadi, B., Pironneau, O.: Applied Shape Optimization for Fluids. Clarendon Press, Oxford (2001)
48. Moore, A.M.: Data assimilation in a quasigeostrophic open-ocean model of the Gulf-stream region using the adjoint model. J. Phys. Oceanogr. **21**, 398–427 (1991)
49. Nechaev, V., Yaremchuk, M.I.: Application of the adjoint technique to processing of a standard section data set: World Ocean circulation experiment section s4 along 67°s in the Pacific Ocean. J. Geophys. Res. **100**(C1), 865–879 (1994)
50. Pedlosky, J.: Geophysical Fluid Dynamics. Springer, New-York (1979)
51. Ramdani, K., Tucsnak, M., Weiss, G.: Recovering the initial state of an infinite-dimensional system using observers. Automatica **46**(10), 1616–1625 (2010)
52. Rostaing-Schmidt, N., Hassold, E.: Basic function representation of programs for automatic differentiation in the Odyssée system. In: Le Dimet, F.-X. (ed.) High Performance Computing in the Geosciences, pp. 207–222. Kluwer Academic (1994)
53. Ruggiero, G.A., Ourmières, Y., Cosme, E., Blum, J., Auroux, D., Verron, J.: Data assimilation experiments using diffusive Back-and-Forth nudging for the nemo ocean model. Nonlinear Process. Geophys. **22**(2), 233–248 (2015)

54. Schröter, J., Seiler, U., Wenzel, M.: Variational assimilation of geosat data into an eddy-resolving model of the Gulf stream area. J. Phys. Oceanogr. **23**, 925–953 (1993)
55. Sheinbaum, J., Anderson, D.L.T.: Variational assimilation of XBT data. Part 1. J. Phys. Oceanogr. **20**, 672–688 (1990)
56. Stauffer, D.R., Bao, J.W.: Optimal determination of nudging coefficients using the adjoint equations. Tellus A **45**, 358–369 (1993)
57. Stauffer, D.R., Seaman, N.L.: Use of four dimensional data assimilation in a limited area mesoscale model – Part 1: experiments with synoptic-scale data. Mon. Weather Rev. **118**, 1250–1277 (1990)
58. Talagrand, O.: Assimilation of observations, an introduction. J. Meteorol. Soc. Jpn. **75**(1B), 191–209 (1997)
59. Talagrand, O., Courtier, P.: Variational assimilation of meteorological observations with the adjoint vorticity equation. Part 1: theory. Q. J. R. Meteorol. Soc. **113**, 1311–1328 (1987)
60. Thacker, W.C., Long, R.B.: Fitting dynamics to data. J. Geophys. Res. **93**, 1227–1240 (1988)
61. Veersé, F., Auroux, D., Fisher, M.: Limited-memory BFGS diagonal preconditioners for a data assimilation problem in meteorology. Optim. Eng. **1**(3), 323–339 (2000)
62. Verron, J., Gourdeau, L., Pham, D.T., Murtugudde, R., Busalacchi, A.J.: An extended Kalman filter to assimilate satellite altimeter data into a non-linear numerical model of the tropical pacific ocean: method and validation. J. Geophys. Res. **104**, 5441–5458 (1999)
63. Verron, J., Holland, W.R.: Impact de données d'altimétrie satellitaire sur les simulations numériques des circulations générales océaniques aux latitudes moyennes. Ann. Geophys. **7**(1), 31–46 (1989)
64. Vidard, A., Le Dimet, F.-X., Piacentini, A.: Determination of optimal nudging coefficients. Tellus A **55**, 1–15 (2003)
65. Zou, X., Navon, I.M., Le Dimet, F.-X.: An optimal nudging data assimilation scheme using parameter estimation. Q. J. R. Meteorol. Soc. **118**, 1163–1186 (1992)