OpenDA-NEMO framework for ocean data assimilation

Nils van Velzen[1] · Muhammad Umer Altaf[1] · Martin Verlaan[1]

Received: 29 September 2015 / Accepted: 2 March 2016

Abstract Data assimilation methods provide a means to handle the modeling errors and uncertainties in sophisticated ocean models. In this study, we have created an OpenDA-NEMO framework unlocking the data assimilation tools available in OpenDA for use with NEMO models. This includes data assimilation methods, automatic parallelization, and a recently implemented automatic localization algorithm that removes spurious correlations in the model based on uncertainties in the computed Kalman gain matrix. We have set up a twin experiment where we assimilate sea surface height (SSH) satellite measurements. From the experiments, we can conclude that the OpenDA-NEMO framework performs as expected and that the automatic localization significantly improves the performance of the data assimilation algorithm by successfully removing spurious correlations. Based on these results, it looks promising to extend the framework with new kinds of observations and work on improving the computational speed of the automatic localization technique such that it becomes feasible to include large number of observations.

Keywords NEMO · OpenDA · Data assimilation · Localization techniques · Double-gyre ocean model

Responsible Editor: Lars Nerger

This article is part of the Topical Collection on the *47th International Liège Colloquium on Ocean Dynamics, Liège, Belgium, 4–8 May 2015*

✉ Nils van Velzen
nilsvanvelzen@gmail.com

[1] Delft University of Technology, Delft, The Netherlands

1 Introduction

Due to present day computational resources, it is possible to build a sophisticated ocean models. There are always unavoidable inaccuracies in the observations and in the models, therefore a realistic oceanographic system must be capable of modeling these uncertainties. Data assimilation methods provide a good way of reducing uncertainties in ocean dynamic systems.

Data assimilation (DA) is usually used to enhance model predictions by constraining their outputs with available observations. DA methods are usually divided into two main categories: variational methods based on least-squares fitting (Bennet 1992), and sequential methods based on the so called Kalman filter (Evensen 2003). Recently, sequential ensemble Kalman filter methods became popular in geophysical applications as they are easy to implement, are robust, and computationally reasonable. Different Ensemble Kalman filter (EnKF) variants were developed in recent years depending on whether or not the observations are perturbed before assimilation (Tippett et al. 2003; Burgers et al. 1998; Verlaan and Heemink 1997; Houtekamer and Mitchell 1998; Anderson 2001; Bishop et al. 2001; Whitaker and Hamill 2002; Hoteit et al. 2002).

EnKFs are often used with relatively small ensembles as compared to the numerical dimension of the system state (e.g., ensembles are often in the order of $\mathcal{O}(10-100)$ while the dimension of state vectors in an ocean model can easily be in millions). This means that the estimation error covariance matrix is always singular and can only represent the model errors in a low dimensional space. A small ensemble size could certainly lead to systematically underestimated variances and spuriously large cross covariances in the sample error covariance matrix (Hamill et al. 2001). In order to negate these effects, two auxiliary techniques are used. These are covariance inflation (Anderson and Anderson 1999) and localization (Hamill et al. 2001). Covariance inflation tackles with the underestimation of the variances, while covariance localization addresses the problems of singularity and overestimation of the cross covariances.

Implementing a correct and efficient DA method for a model can be an elaborate task. There are various software packages that provide standard implementations that take away the need to develop custom implementations for each simulation model. All these packages have their strong and weak points. The package that suits best for a particular application depends on many factors. Among these are PDAF (Nerger and Hiller 2013), a toolbox written in FORTRAN 90 that contains highly optimized implementation of LETKF, and LSEIK for usage in strong parallel environments. Sequoia (Le Hénaff and Marsaleix 2009) contains a collection of FORTRAN 95 routines that allow users to build their own data assimilation system. DART (Raeder et al. 2012) is a framework containing ensemble-based data assimilation algorithms and tools. The NERSC repository, containing implementations of various EnKFs. OAK (Barth et al. 2008) is a toolbox containing various filters and tools that interacts with the model using NetCDF files. SESAM (Brasseur 2006) is a set of data assimilation building blocks that are combined together using NetCDF files. OpenDA is a data assimilation framework written in JAVA and some parts in C. Models can be coupled to OpenDA in two ways: in memory coupling through a software library or a black box coupling in which all the interactions between data assimilation method and model goes though input and output files. OpenDA is described in more detail in Section 4. A thorough description of all the frameworks mentioned above can be found in (Heemink et al. 2012).

In this work, we generalize the OpenDA framework to the specific needs for ocean applications. We have coupled the NEMO (Madec 2014) ocean model with OpenDA toolbox using the black box approach. Several synthetic experiments, assimilating the simulated altimetry into a double-gyre NEMO ocean model, are performed with the objective to investigate the impact of different EnKF setups in the quality of analysis. These setups mainly focused on the observation distribution, the ensemble size, and the localization length scale. Furthermore, attention will also be paid to investigate the efficiency and usefulness of localization. Presently, OpenDA only supports distance-based localization for in memory coupling. The use of the automatic localization technique which is part of the OpenDA toolbox is not limited to in memory coupled models and can be easily applied for the NEMO model using the black box coupling. The development of automatic localization techniques is still new and limited to few studies, e.g., (Anderson and Anderson 1999; Zhang and Oliver 2011). From these, only the later is suitable for operational usage due to the relative limited additional computational costs. It is interesting to investigate whether automatic localization is able to improve the performance of EnKF, when it is applied to a medium sized model like the double-gyre NEMO configuration.

The double-gyre NEMO ocean model corresponds to an idealized configuration of the NEMO model: a square and 5000-m-deep flat bottom ocean at mid latitudes (the so called square-box or SQB configuration). The domain extends from 24N to 44N, over 30° in longitude (60–30 W) with 11 vertical levels between 152 and 4613 m in depth. The minimum horizontal resolution of the model is 1/4°. This setup is known as the SEABASS configuration (Bouttier et al. 2012).

The paper is organized as follows. Section 2 presents an overview of the EnKF implemented in this study. Section 3 describes the automatic localization technique. A brief overview of OpenDA data assimilation toolbox with its new features, parallelization of ensembles, and automatic localization is given in Section 4. The ocean model, NEMO, configuration, and the assimilation setup are explained in Section 5. In Section 6, assimilation results of the EnKF filter are presented and discussed. Concluding remarks follow in Section 7.

2 Ensemble Kalman filter

Consider the following discrete dynamical system

$$\mathbf{x}_k^t = \mathbf{M}_{k,k-1}\mathbf{x}_{k-1}^t + \eta_k, \tag{1}$$

where $\mathbf{x}_k^t$ denotes the vector representing the true state of the system at time k, $\mathbf{M}_{k,k-1}$ is the state transition operator that takes as inputs the state at time $k-1$ and outputs the state at time k, and η_k is the system noise with covariance matrix $\mathbf{Q}_k$. At time k, the observed data vector is given by

$$\mathbf{y}_k = \mathbf{H}_k\mathbf{x}_k^t + \epsilon_k. \tag{2}$$

Here, $\mathbf{y}_k$ is the vector with observed values and ϵ_k is the observational noise with covariance matrix $\mathbf{R}_k$. If both the dynamical and observation systems are linear, the state estimation problem is solved by the Kalman filter (Kalman 1960). The main procedures of the Kalman filter involve recursively computing the means and covariance matrices of the system state $\mathbf{x}_k$ (conditioned on available observations), as outlined below.

– Forecast step: At time instant $(k-1)$, propagate the state mean (also called analysis) $\mathbf{x}_{k-1}^a$ and the associated error covariance matrix $\mathbf{P}_{k-1}^a$ to obtain the forecast state (also referred to as background) $\mathbf{x}_k^f$ and the associated covariance $\mathbf{P}_k^f$ at next time k respectively as

$$\mathbf{x}_k^f = \mathbf{M}_{k,k-1}\mathbf{x}_{k-1}^a \, , \tag{3}$$

$$\mathbf{P}_k^f = \mathbf{M}_{k,k-1}\,\mathbf{P}_{k-1}^a\,\mathbf{M}_{k,k-1}^T + \mathbf{Q}_k \, . \tag{4}$$

– Analysis step: When a new observation $\mathbf{y}_k$ becomes available at time k, update $\mathbf{x}_k^f$ and $\mathbf{P}_k^f$ to their analysis counterparts, $\mathbf{x}_k^a$ and $\mathbf{P}_k^a$ with the Kalman update formula:

$$\mathbf{x}_k^a = \mathbf{x}_k^f + \mathbf{K}_k\left(\mathbf{y}_k - \mathbf{H}_k\mathbf{x}_k^f\right) \, , \tag{5}$$

$$\mathbf{P}_k^a = \mathbf{P}_k^f - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_k^f \, , \tag{6}$$

$$\mathbf{K}_k = \mathbf{P}_k^f\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k^f\mathbf{H}_k^T + \mathbf{R}_k)^{-1} \, , \tag{7}$$

with $\mathbf{K}_k$ being Kalman gain.

Geophysical fluid dynamical (and sometimes observational) models are nonlinear, and thus the conventional Kalman filter cannot be directly applied for data assimilation into these systems. To account for nonlinearities, modifications are introduced in the original Kalman filter using a Monte Carlo approach and refers usually as ensemble Kalman filtering (EnKF) (Evensen 1994). The EnKF algorithm uses an ensemble of the system states to represent the model error. The ensemble size n is typically much smaller than the dimension m_x of the state vector. Suppose that an n-member analysis ensemble $\mathbf{X}_{k-1,i}^a : i = 1, 2, ..., n$ is available at the $(k-1)$th filtering step, we take the set $\mathbf{X}_k^f$ as the forecast ensemble at instant k, and the forecast sample mean $\hat{\mathbf{x}}_k^f$ and covariance $\hat{\mathbf{P}}_k^f$ are given by

$$\hat{\mathbf{x}}_k^f = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_{k,i}^f ; \qquad \hat{\mathbf{P}}_k^f = \frac{1}{n-1}\sum_{i=1}^{n}\left(\mathbf{x}_{k,i}^f - \hat{\mathbf{x}}_k^f\right) \times \left(\mathbf{x}_{k,i}^f - \hat{\mathbf{x}}_k^f\right)^T . \tag{8}$$

In practice, $\hat{\mathbf{P}}_k^f$ needs not be calculated. The Kalman gain $\mathbf{K}_k$ is then approximated by

$$\mathbf{K}_k = \hat{\mathbf{P}}_k^f\mathbf{H}_k^T(\mathbf{H}_k\hat{\mathbf{P}}_k^f\mathbf{H}_k^T + \mathbf{R}_k)^{-1} \tag{9}$$

When observation $\mathbf{y}_k$ is made available, one updates each member of the forecast ensemble by

$$\mathbf{x}_{k,i}^a = \mathbf{x}_{k,i}^f + \mathbf{K}_k\left(\mathbf{y}_{k,i}^s - \mathbf{H}_k\mathbf{x}_{k,i}^f\right), i = 1, 2, ..., n, \tag{10}$$

where $\mathbf{y}_{k,i}^s$ are the "surrogate" (i.e., perturbed) observations generated by drawing n samples from the normal distribution of mean $\mathbf{y}_k$ and covariance $\mathbf{R}_k$ (Burgers et al. 1998). Propagating $\mathbf{X}_k^a$ forward to the next time instant, one starts a new assimilation cycle, and so on.

Most of the computational time of an ensemble based data assimilation method is dominated by the forward model simulations. The ensemble members can propagate the model state forward in time in parallel but still due to limited computational resources, we often need to use relatively small ensemble sizes. The use of a small ensemble may lead to filter divergence due to spurious correlations and unrealistic updates of the ensemble members (Hamill et al. 2001). Distance-dependent localization techniques are a popular way to compensate for small ensemble size. The distance-based localization techniques determine the weighting coefficients β between model variables/measurement locations. These weights $\beta \in [0, 1]$ are determined based on the physical distance of the variables. Weights will be close to zero for elements with no correlation and near one for elements we do expect strong correlations. Localization can be applied directly on the covariance matrices or to the computed gain matrix. In this paper, we only consider the latter and update the model using a localized gain matrix according to

$$\mathbf{K}^{Loc} = \mathbf{K} \circ \beta_{xy}, \tag{11}$$

where $\mathbf{K}$ denotes the gain matrix, $\mathbf{K}^{Loc}$ the localized gain matrix, and β_{xy} the weighting coefficients between the observation locations and the model state elements.

The localization weights are often determined using a correlation function with local support that takes the distance as input. This function is non zero for distances up to a given threshold like the fifth-order piecewise rational function as defined in Gaspari and Cohn (1999). Non smooth functions assigning only 0 and 1 weights can also be used (Anderson and Anderson 1999), but these have shown to be inferior (Anderson 2004). Although this approach assumes to be a good measure for correlation, the distances cannot always be properly defined for some sources of observations (Anderson 2004) or the correlation scales can be time dependent. Hence, this method often needs quite some tuning in order to find the optimal distances.

3 Automatic localization

A step forward in fully generic data assimilation methods is to handle the localization in an automatic way avoiding manual configuration, modeling, and tuning of the localization distances. Anderson (2004) suggests a hierarchical ensemble filter. This method tries to tackle the heuristic issues and determine localization weights based on a theoretical basis. The idea is to split all the ensembles into N_g groups of N_e ensembles of ensembles. The gain matrices $K_j, j = 1, .., N_g$ for all the groups are computed. The elements $k_{i,j}$ of the gain matrices are assumed to be randomly drawn from the same distribution that contains the true values of k_i. The localization weights α_i are selected such that it minimizes

$$\sqrt{\sum_{j=1}^{N_g}\sum_{k=1,k\neq j}^{N_g}\left(\alpha_i k_{i,k} - k_{i,j}\right)}. \tag{12}$$

This method requires a large ensemble size and is therefore computationally inefficient compared to a traditional implementation using distance-based localization. However, it can be used to derive localization statistics which can produce high-quality localization strategy for production runs.

Another approach suggested by Zhang and Oliver (2011) uses a bootstrap technique to identify spurious correlation in the gain matrix and then generate weights accordingly. From the original ensemble x_i with $i = 1, .., N_e$, N_B ensembles of size N_e are generated by randomly drawing ensemble members from the original ensemble. The bootstrapped ensemble will contain some of the ensemble members of the original ensemble several times. From each bootstrapped ensembles, we generate gain matrices $\hat{\mathbf{K}}_k, k \in [1, .., N_B]$ for each element $\hat{k}_{i,j}$ of the gain matrix we compute the estimator of the variance as

$$\hat{\sigma}^2_{k_{i,j}} = \frac{\sum_{m=1}^{N_B}\left(\hat{k}^*_{i,j,m} - \bar{k}_{i,j}\right)^2}{N_B}, \tag{13}$$

where $\bar{K}$ denotes the mean of the bootstrapped gain matrices $\hat{K}_k$. The squared ratio between mean and variance is defined as

$$\hat{C}^2_{v_i,j} = \frac{\hat{\sigma}^2_{k_{i,j}}}{\bar{k}^2_{i,j}}. \tag{14}$$

The localization weights are then defined by

$$\beta_{i,j} = \frac{1}{1 + \left(1 + 1/\sigma^2_\alpha \hat{C}^2_{v_i,j}\right)}. \tag{15}$$

The parameter σ^2_α can be used to balance between the deletion of spurious correlations and possibly deleting real correlations. Zhang and Oliver (2010), investigated the best possible value of σ and came up with an optimal value of $\sigma^2_\alpha = 0.36$.

This method can easily be incorporated into an existing implementation of an EnKF. The analysis step will become roughly N_B times more expensive computationally which might be acceptable for assimilating a moderate amount of observations but is not be practical for assimilating large number of observations. A possible solution for assimilating large amounts of observations is to define a large region around each observation isolating it from the surrounding to avoid any correlations from which it is safe to assume there is no correlation between the observation and the points outside this area. The automatic localization can then be applied only to the observations inside this area.

4 OpenDA data assimilation toolbox

OpenDA is an open-source toolbox for developing and applying data assimilation and calibration algorithms to dynamic models (OpenDA 2013). The design philosophy of OpenDA is to breakdown data assimilation into a set of building blocks using the principles of object-oriented programming. The OpenDA toolbox consists of two parts; a description of the building blocks with their interfaces, and a library of default implementations of those building blocks. It stems from the merger of two data assimilation initiatives: COSTA (van Velzen and Segers 2010; Altaf et al. 2009; Heemink et al. 2010; Altaf et al. 2011) and DATools (Weerts et al. 2010).

OpenDA is written in Java. The object oriented nature of java fits well to the modular design of OpenDA and allows easy extension of functionality. The system is set up in such a way that end users can setup experiments without any coding or compiling requirements. Java was historically considered slow, the performance has improved significantly by many improvements in the Java virtual machine a.o. Just-In-Time compilation and Adaptive optimization (Sestoft 2010). In order to get a performance similar to a data assimilation algorithms written purely in languages like C and Fortran, some computationally demanding parts are written in C.

Once a model is prepared for OpenDA, it is possible to try out various off-the-shelf data assimilation algorithms without any further programming. It has been shown that the performance is close to dedicated implementations (van Velzen and Verlaan 2007) and furthermore can automatically propagate the ensemble of models either serially or concurrently.

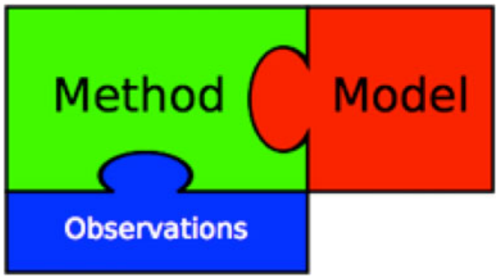


Fig. 1 OpenDA defines three building blocks: *method* (the data assimilation or calibration algorithms), *observations* (the stochastic observer for handling the observations), and the *model*

4.1 OpenDA components

As illustrated in Fig. 1, OpenDA defines three major building blocks: *method* (the data assimilation or calibration algorithms), *observations* (the stochastic observer for handling the observations), and the *model*. The end user can make arbitrary combinations of these building blocks to set up an experiment without requiring any programming. The system is configured using an XML schema where a user defines options regarding: algorithm type (EnSR, EnKF, etc.) and ensemble size; the variables to assimilate and grid location within the model; and associated uncertainties of the model, forcing data, parameters, and observations. Using the default implementations, observations can be stored in various formats like NetCDF, CSV, NOOS, or in an SQL database. OpenDA can produce output in various formats such as ASCII, Matlab, Python, and NetCDF to easily analyze the filtering performance and by plotting useful graphs.

4.2 Coupling a model to OpenDA

The OpenDA model interface has already been implemented for over 20 dynamic model codes (van Velzen et al. 2012). There are two approaches to link a model to OpenDA. The most simple and often the best choice is a so called black box coupling. In case of a black box coupling, OpenDA interacts with the model only by writing model input files, running the model executable and interpretation of the model output files. The coupling can be realized without access and changes to the model code. There are situations where the black box coupling is not the best choice especially when the overhead of restarting the model is high or when the model lacks proper restarting functionality. In these situations, it is possible to setup an "in-memory coupling." The model must be made available as a software library that implements a specified list of routines, called the OpenDA model interface. OpenDA supports in-memory coupling for models written in various programming languages, e.g., Fortran, C, C#, and Python. The in-memory coupling is most efficient computationally and potentially unlock all data. This allows easy creation of the complex interpolation operators and the noise processes. The amount of work needed to create an in memory coupling is very model dependent. The amount of work can be small when the model itself is available as a library with functionality to access the model state, e.g., the models that implement the OpenMI interface (Gregersen et al. 2007; Gijsbers et al. 2010). In such an ideal situation, there is no need to have access to the model source code. Writing some bridging code will be sufficient (Ridler et al. 2014). On the other hand, it can be a serious amount of work to realize an in memory coupling especially for some large legacy simulation codes.

OpenDA contains a black box coupling tool that minimizes the effort needed to create a highly configurable black box coupling as illustrated in Fig. 2. The user must write a wrapper code for reading and writing the input and output files of the model. Depending on the type of file, input, output, or both, the wrapper code will be able to read and/or write the relevant data from/to the model files. This is the only programming effort required to realize the black box

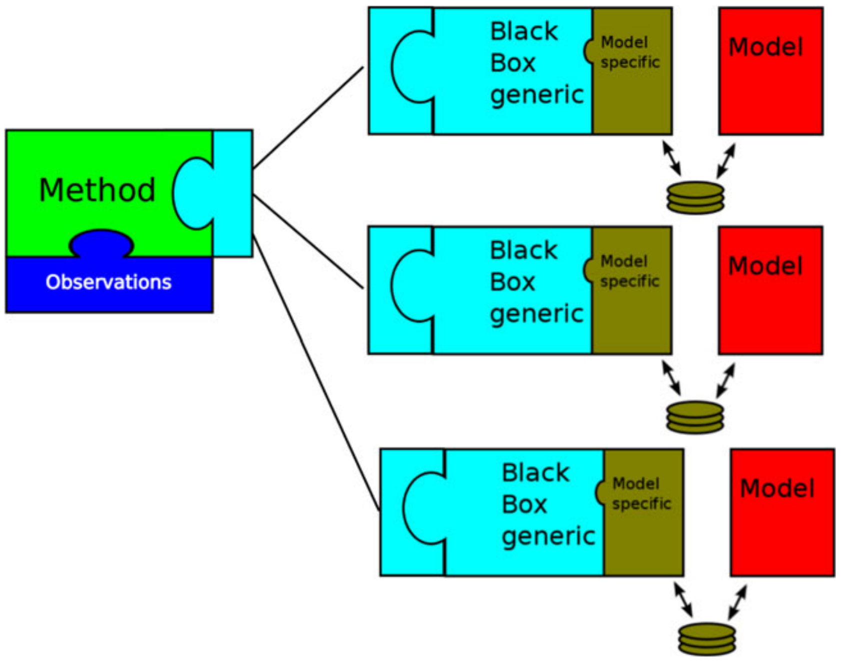


Fig. 2 The OpenDA black box coupling tool simplifies coupling a model and arbitrary can be run in parallel. The user only needs to provide a small part of model specific code to read and write the model output and input files in order to couple the model. The parallelization is handled by in interface model that handles all parallel interaction between the data assimilation method and the models

coupling. All other configuration and specification is done using black box configuration files.

The black box configuration consists of three layers:

- Wrapper layer: contains the specification of location of (template) input files, the names of executables needed to perform a model run and the input and output files that have to be processed using the wrapper code.
- Model layer: contains a list of the variables from the required input and output files. In this layer, it is possible to rename the ID's of the variables in order to ensure uniqueness.
- StochModel layer: The model as presented to the data assimilation method is constructed. The state vector is composed by listing the associated variables and arrays, noise models on model forcings can be specified and information is provided about matching model predictions to observations.

The advantage of this approach is that it offers a lot of flexibility without the need to do any additional programming. The end user can easily add or remove values to the part of the state that is updated by assimilation and experiment with various sources of observations or noise models.

4.3 Parallel computing

To reduce the computational time, the model runs in ensemble-based data assimilation methods can be executed in parallel. The setup of OpenDA allows automatic execution of the models in a natural way. Parallelization is handled using the so called interface model. This is a model that extends the functionality of arbitrary OpenDA model. In this case, it handles and hides the parallelisation of ensemble members. Other examples of interface models in OpenDA are models for automatic bias correction and Kalman smoothing. These interface model create an interface between the actual model and the data assimilation method. It supports both parallelisation of models using threads and running the models on remote machines. The interface model works for arbitrary models since it only makes use of the same methods from the model interface as the algorithm does as illustrated by the shape of the puzzle peaces in Fig. 2.

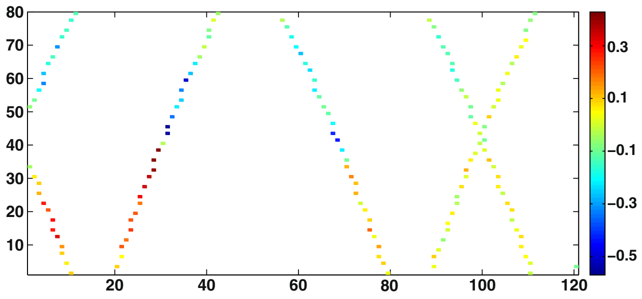


Fig. 4 SSH Observations for assimilation period September 18, 2009

4.4 Localization in OpenDA

The current release of OpenDA (2.1) supports localization using the shur product on the Kalman gain matrix. The user provides a function that takes the meta-information on the observations as input and produces the weights β_{xy}. For numerical models with a fixed, regular computational grid, such an implementation is not difficult. However, OpenDA has been coupled to many simulation suites, e.g., for modeling water quality, rivers, groundwater. The computational grids are often irregular and the state vector contains various quantities depending on the user settings. Writing a generic function for determining the localization weights, based on some parameters, is an elaborate task for these models. Moreover, some of the end users are novice data assimilation users and require a more black box approach.

The method suggested by Zhang and Oliver (2011) is part of OpenDA and is conceptually very suitable. It does not need any configuration except for the bootstrap size N_B and also adds at least some necessary form of localization to all models in OpenDA. Though OpenDA still provides provision for a custom localization function when needed.

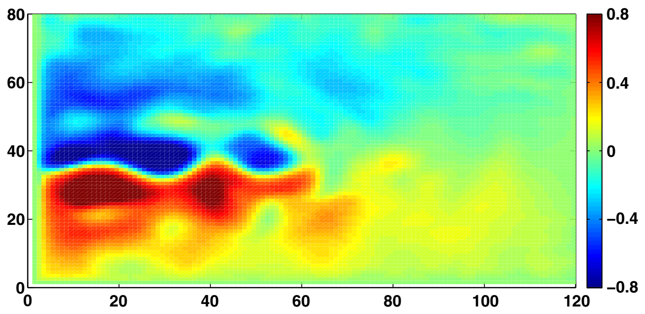


Fig. 3 A snapshot of SSH (m) for the SQB model configuration (April 23, 2009)

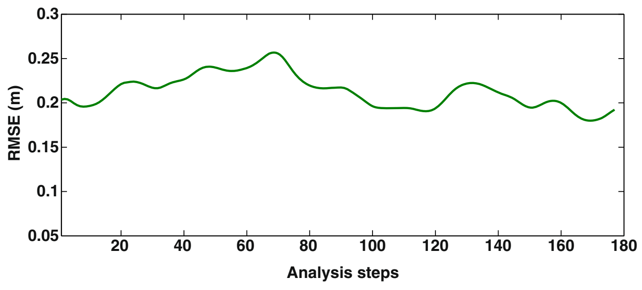


Fig. 5 Root mean square errors (RMSE) in SSH (m) between the free model run and truth taken at every 2 days period for the year 2009

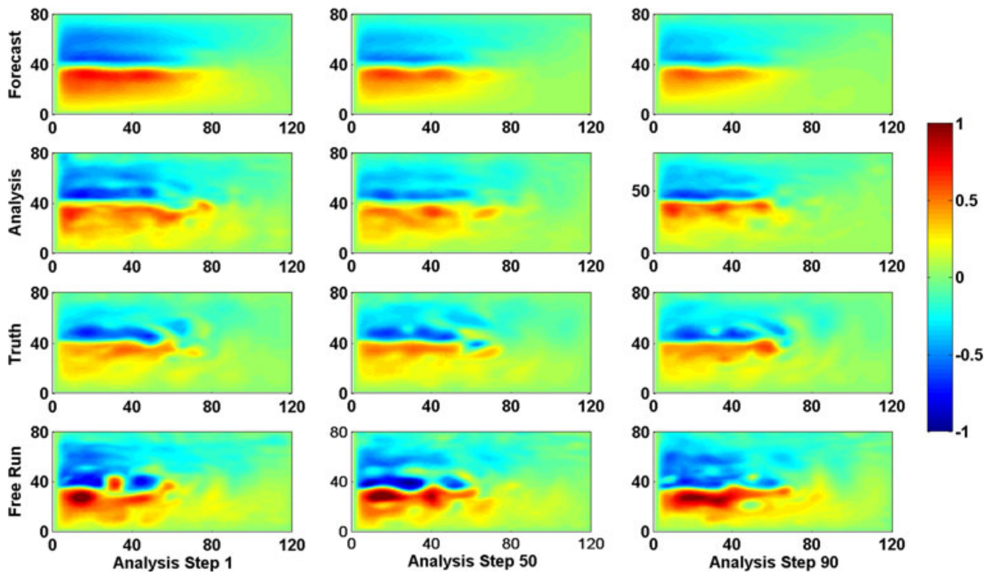


Fig. 6 Snapshots of sea surface height (m) at analysis step 1, 50, and 90 for *top*: forecast using EnKF, *second row*: analysis using EnKF, *third row*: truth, and *bottom*: free run without data assimilation

5 NEMO ocean model

The model setup used in this study is based on an idealized configuration of the NEMO primitive equation ocean model (Cosme et al. 2010). The domain is a square box (between 25 and 45 N), 5000-m-deep flat bottom ocean at mid latitudes and refers as SQB configuration. The model domain is closed with lateral boundaries. A double gyre circulation is created with constant zonal wind forcing blowing westward in the northern and southern parts of the basin and eastward in the middle part of the basin. Figure 3 shows the snapshot of the model sea surface height (SSH). The flow is dominated by chaotic mesoscale dynamics, with largest eddies that are about 100 km wide (Beckers et al. 2012), and to which correspond velocities of about 1 m/s and dynamic height differences of about 1 m. These properties are quite similar in shape and magnitude to the Gulf Stream (North Atlantic).

The primitive equations of the NEMO ocean model are discretized on an Arakawa C grid, with a horizontal resolution of $1/4 \times 1/4 \cos(\phi)$ and 11 z-coordinate levels along the vertical. The model uses a "free surface" formulation, with an additional term in the momentum equation to damp the fastest external gravity waves and the bi-harmonic diffusion operator. The model time integration is based on a leap-frog scheme and an Asselin filter (Cosme et al. 2010).

5.1 Assimilation setup

For the present SQB configuration, a time step is chosen as 15 min. The model is spin-up for a period of 40 years. The model calendar is idealized to 30 days per month. The model state vector is multivariate, containing 2 time instances of 17 variables. The two time instances are due to the leap-frog scheme. The user can select which variables of the state are updated by the data assimilation algorithm by selecting them in the OpenDA configuration file. In order to keep the leap-frog scheme in tact, the user should always select both time instances of a variable to be updated.

The ensemble Kalman filter (EnKF) is used for all the experiments. The assimilation is performed during 1 year (360 days calendar) with analysis frequency of 2 days. The model output of the year 75 is taken as the truth. Observations of sea surface heights (SSH) are generated from the truth by adding spatially uncorrelated Gaussian noise with standard deviation of 6 cm. The observation grid is generated from the ENVISAT and Jason-1 flight parameters: 35 days and 1002 passes one cycle for ENVISAT and 10 days and 254 passes one cycle for Jason-1. Figure 4 shows SSH observation for one assimilation period. In our experiments, we update the sea surface height variables of the state vector.

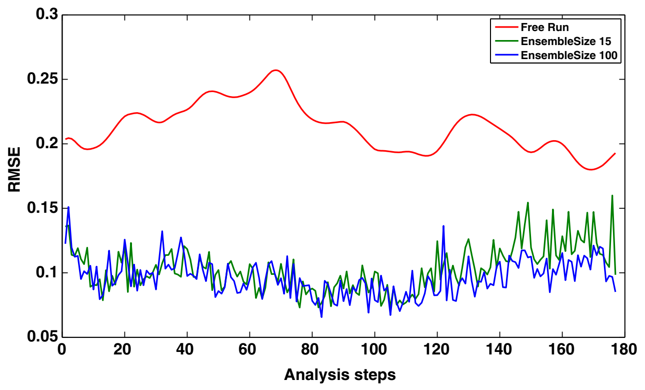


Fig. 7 RMSE in SSH (m) between EnKF and truth at analysis steps using ensemble members of sizes 15 and 100

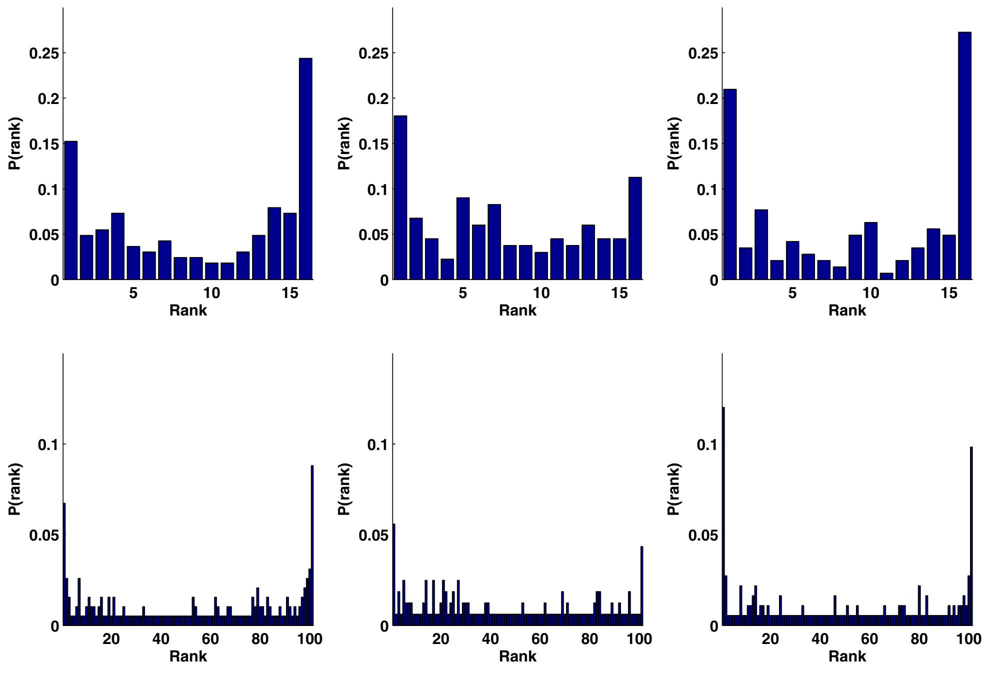


Fig. 8 Rank histogram of the ensembles forecast step 20, 70, and 140 (from *left* to *right*) using EnKF. The *top row* corresponds to an ensemble size of 15 and the *bottom row* to an ensemble size of 100. The U shape can be an indication that the ensembles suffer from conditional biases or lack of variability

The beginning pass numbers for both satellites of the year 2009 are taken. Since the observation grid is different from the model grid, while generating the SSH observations, bilinear interpolation is used to generate SSH values from model grid to observation grid (Yan et al. 2014). Figure 5 presents the root mean square errors (RMSE) in SSH (m), between the truth and the free model run with an interval of every 2 days (analysis steps). The errors are in the range between 0.15 and 0.30 m.

The initial ensemble is generated with 100 members from a 10-year interval (years 41–50) of free model simulation with outputs every 30 days are used. The last 100 outputs are used to initialize the ensemble. The 10-year interval is selected far from the true state year, for better evaluation of the assimilation performance. From this 100 ensemble members, 4 sets of different ensemble members are picked (15, 30, 50, 100). This is helpful in evaluating the results with respect to ensemble members and localization.

The computational time is dominated by the NEMO model runs. Depending on the number of ensemble members, 1–3 % is consumed by the classical EnKF implementation in OpenDA when 8 NEMO models are run in parallel.

6 Results and discussion

As a first step, a set of four assimilation experiments are performed with different set of ensemble members (15, 30, 50, 100). Figure 6 presents the snapshots of SSH for an

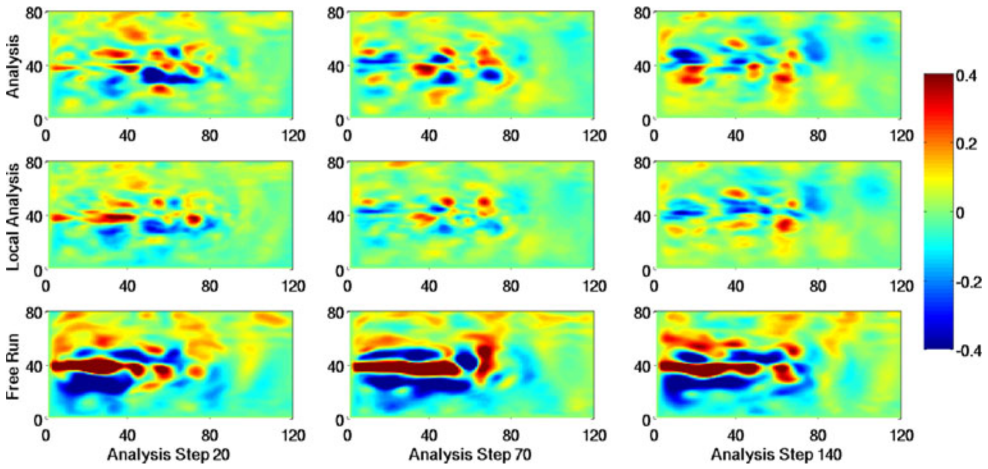
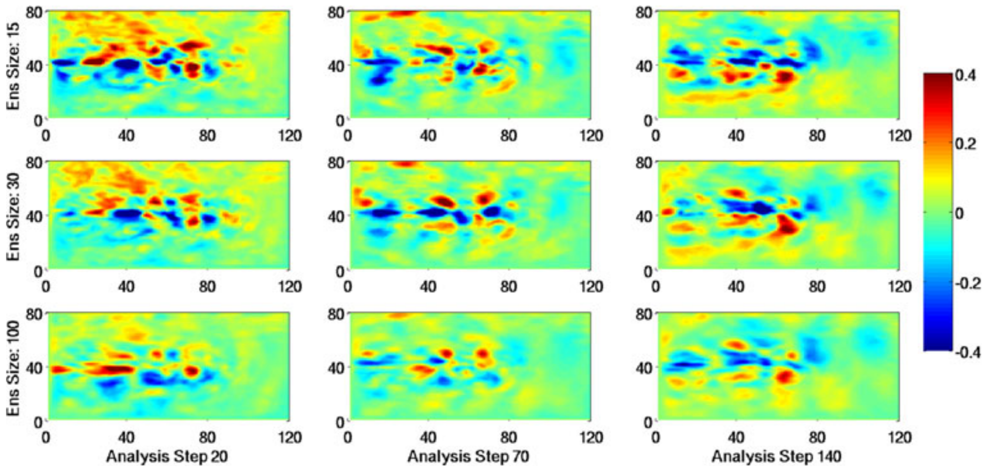


Fig. 9 Snapshots of sea surface height errors from truth at analysis step 20, 70, and 140 for *top*: using EnKF, *middle*: EnKF with localization, and *bottom*: free run without data assimilation

Fig. 10 Snapshots of errors in SSH from truth at analysis step 20, 70, and 140 using Localized EnKF with *top*: ensemble size: 15, *middle*: ensemble size: 30, and *bottom*: ensemble size: 100



assimilation experiment with an ensemble of 100 members. Comparison of forecast and analysis with the truth and free run demonstrates that the figure clearly shows that the assimilation experiments have improved the model results significantly and the analysis at each step brings the model close to the truth. Keeping in mind, we are only assimilating and updating SSH, the forecast results look reasonable as well.

To see the filter performance with lower ensemble member, the RMSE in SSH over the whole model domain (with respect to each analysis step) are compared for different ensemble members. We have plotted these results in Fig. 7 for 15 and 100 ensemble members. Decreasing the number of ensemble members has a negative effect on the performance of the EnKF. The results demonstrate that the RMSE are lower for 100 ensemble case specially for later part of the assimilation. The results also demonstrate that a small ensemble size also gives reasonable assimilation results. One should also note that the data assimilation process has improved the model results significantly. The range of errors with no data assimilation is between 0.15 and 0.30 m which is decreased to 0.05 and 0.15 with data assimilation (Fig. 7).

So far, we have examined the EnKF performance based on RMSE which is the most commonly used measure to evaluate the filter performance. A rank histogram is another common diagnostic to measure the ensemble reliability by repeatedly tallying the rank of the verifying observations at each assimilation step (Hamill 2000). A reliable rank histogram will show up a flat rank histogram. Figure 8 presents the rank histogram of the ensembles forecast at assimilation step 20, 70, and 140 for an ensemble size of 15 and an ensemble size of 100. The U-shape histograms are observed for both the ensemble sizes and can be an indication that the ensembles suffer from conditional biases or lack of variability (Hamill 2000).

The EnKF showed reasonable improvements in assimilation results and showed that the black box coupling between the NEMO model and OpenDA works quite well. The next step is to apply and test the automatic localization strategy explained in Section 2 with the NEMO-OpenDA framework using the same experimental setup. We have selected the bootstrap size $N_B = 50$ which is approximately the average ensemble size in our experiments.

Figure 9 shows the error results in the snapshots of the SSH taken at 20th, 70th, and 140th analysis steps with an ensemble of 100 members. These error plots are shown for EnKF, EnKF with localization and free run without data assimilation. It is evident from these plots that the localization has improved the EnKF results. Figure 10 presents the differences in the snapshots between the truth and localized EnKF using different ensemble sizes (15, 30, and 100). These error snapshots are taken at 20th, 70th, and 140th analysis steps. Increasing the ensemble size greatly influences the simulation results as evident from Fig. 10. As an example, consider the first column of the Fig. 10 (analysis step 20), it can be seen that the scale of these differences has notably reduced and large differences in the upper domain have almost vanished as we increase the ensemble size from 15 to 100. This is true for all three analysis steps shown here.

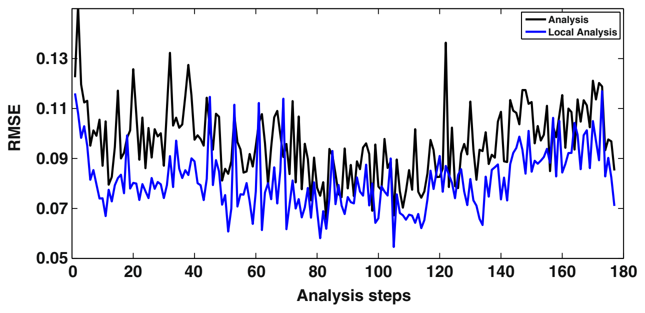


Fig. 11 RMSE in SSH (m) between EnKF and truth at analysis steps with and without auto localization using ensemble of 100 members

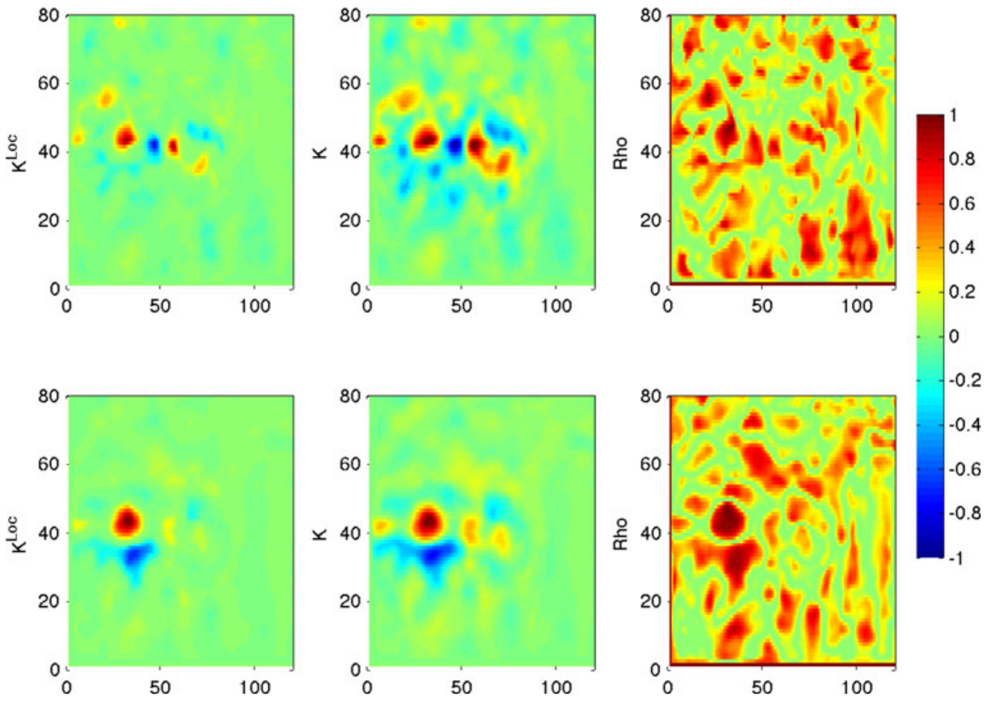


Fig. 12 Illustration of the impact of automatic localization for a single row of the **K** matrix for SSH for various ensemble sizes. The *top row* is created for an ensemble size of 15 and the *lower row* with an ensemble size of 100. The **K** matrices are generated for the same observation on analysis step 60. The first column shows the localized **K** matrix, the second column the **K** matrix without localization, and the last column shows the automatically computed correction factors ρ. The correction factors are not distance based but, they are a measure for uncertainty in the computed elements of K. This is different from factors which are determined using classical localization techniques. However, the method is well capable of removing structures in the **K** matrix which are expected to spurious, sharpening the important structures in the **K** matrix

To see the overall improvements at different analysis steps, Fig. 11 presents the RMSE of SSH over the whole model domain with and without localization for 100 ensemble cases. The RMSE clearly demonstrate that filter performance has improved with the introduction of localization. From the experimental results, we see that the automatic localization has a positive impact on the performance of the filter. To get some idea on what the automatic localization does, we have made plots of the localized and non localized gain matrix and the correction factor ρ for the SSH correction. Figure 12 shows these plots for an arbitrary observation at analysis step 60 created with ensemble size 15 and 100. Note that we cannot do a direct comparison between the gain of the two ensemble sizes since they are created with a different ensemble.

We can clearly see that the localization enhances the structure in the gain matrix by removing some of the small structures which are likely to be created by spurious correlations. At the same time, we see that the non localized gain matrix created with 15 ensembles shows more spurious correlations than the gain created with 100 ensembles. The correction factors are very different from what we would expect from classical localization techniques. The corrections are not location based but are based on the uncertainty of gain elements. As a result, it shows large values as well on locations where the gain has a small value with high probability.

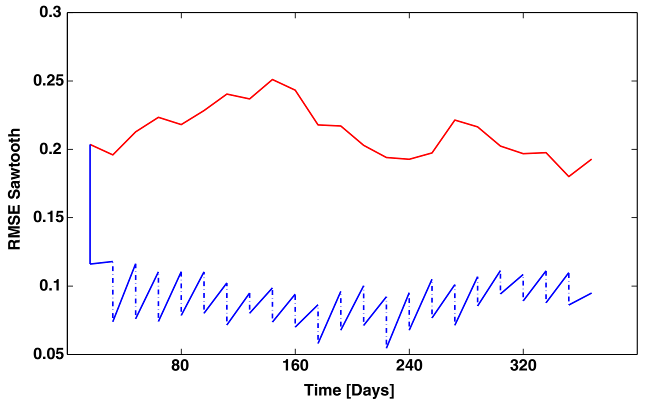


Fig. 13 RMSE in SSH (m) between the truth and free model run (*red line*), RMSE sawtooth patterns with with localized EnKF—100 ensemble members (*blue line*), forecast with localized EnKF—100 ensemble members (*red line*)

Finally, RMSE are computed for both forecast (before assimilation) and analysis (after assimilation). Both forecast and analysis RMSE are plotted to produce sawtooth diagrams. Figure 13 presents RMSE in SSH (m) between the truth and free model run and RMSE sawtooth patterns of the Forecast-Analysis with localized EnKF as we progress in time using 100 ensemble members. It is evident that not only analysis but the forecasts results are also improved with significant margin. One important point to notice here is that we are only assimilating SSH observations, and the model consists of 17 variables including temperature, salinity, etc. So for, a realistic case we expect different sources of observations to be part of assimilation process and that will further improve our analysis and forecasts results.

7 Conclusions and future work

We have created an OpenDA-NEMO ocean data assimilation framework using OpenDA black box coupling. With limited programming effort, we have managed to read and write the NEMO netcdf restart file and configuration file. With the present framework, a set of twin experiments was performed for the NEMO model where we assimilate satellite SSH values using EnKF. We have used the automatic parallelization functionality of OpenDA to run the ensemble members in parallel.

From the results of these experiments, we see that a run without data assimilation has an RMSE in SSH that varies around 23 cm which is quite large since the SSH varies between -1 and 1 m. The EnKF, assimilating SSH improves the model predictions over the whole domain. Increasing the ensemble size does improve the performance of the EnKF filter but only to some extent. The experiments were repeated using the auto localization technique which is now a part of OpenDA. We have used the auto localization technique as the distance based localization technique was not yet available for black box coupling, however, will be present in the future version of OpenDA. The auto localization method does not determine weights based on the distances but based on the uncertainty of elements in the gain matrix. The introduction of automatic localization significantly improved the performance of EnKF for all ensemble sizes. A visual inspection shows that this method yields a very different localization weights as we are used to see in distance based localization. The method is, however, capable of keeping the important structures in the gain matrix intact while reducing artifacts that are likely caused by spurious correlations. The performance of the method justifies more research. In future work, we should compare the automatic localization technique to the distance-based localization and find out which method performs best for various types of observations and situations.

The automatic localization technique is very easy to use but increases the computational time of the analysis significantly. Using a bootstrap size of 50, the computational time of the analysis increased with a factor of 10 in our experiments. This makes the method only feasible to use for a moderate number of observations. We have to look for ways to reduce the computational time of the bootstrapping techniques. Possible solutions are to perform the bootstrapping in parallel and to combine the automatic localization method with a local analysis strategy such that the auto localization only needs to be computed for a limited computational domain.

Acknowledgments This work is supported by SANGOMA a European FP7-SPACE-2011 project, Grant 283580.

References

Altaf MU, Heemink AW, Verlaan M (2009) Inverse shallow-water flow modelling using model reduction. Int J Multiscale Comput Eng 7:577–596

Altaf MU, Verlaan M, Heemink AW (2011) Efficient identification of uncertain parameters in a large scale tidal model of European continental shelf by proper orthogonal decomposition. Int J Numer Meth Fluids. doi:10.1002/fld.2511

Anderson JL (2001) An ensemble adjustment Kalman filter for data assimilation. Mon Wea Rev 129:2884–2903

Anderson JL (2004) A hierarchical ensemble filter for data assimilation. http://www.image.ucar.edu/staff/jla/filter.pdf

Anderson JL, Anderson SL (1999) A Monte Carlo implementation of the nonlinear filtering problem to produce ensemble assimilations and forecasts. Mon Wea Rev 127:2741–2758

Barth A, Alvera-Azcárate A, Weisberg RH (2008) Assimilation of high-frequency radar currents in a nested model of the West Florida shelf. Geophys Res 113:C08,033

Beckers JM, Barth A, Heemink A, Velzen N, Verlaan M, Altaf MU, Van Leeuwen PJ, Nerger L, Brasseur P, Brankart JM, de Mey P, Bertino L (2012) Deliverable 4.1: Benchmark definitions. http://www.data-assimilation.net/Documents/sangomaDL4.1.pdf

Bennet A (1992) Inverse methods in physical oceanography. Campbridge Univ. Press

Bishop CH, Etherton BJ, Majumdar SJ (2001) Adaptive sampling with ensemble transform Kalman filter. Part I: theoretical aspects. Mon Wea Rev 129:420–436

Bouttier P-A, Blayo E, Brankart JM, Brasseur P, Cosme E, Verron J, Vidard A (2012) Towards variational assimilation of altimetric data in a high resolution ocean model. Newsletter Mercator Océan:2012

Brasseur PVJ (2006) The seek filter method for data assimilation in oceanography: a synthesis. Ocean Dyn 56(5-6):650–661

Burgers G, van Leeuwen PJ, Evensen G (1998) On the analysis scheme in the ensemble Kalman filter. Mon Wea Rev 126:1719–1724

Cosme E, Brankart J, Verron J, Brasseur P, Krysta A (2010) Implementation of a reduced rank square root smoother for high resolution ocean data assimilation. Ocean Modell 33:87–100

Evensen G (1994) Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. J Geophys Res 162(10):143–10

Evensen G (2003) The ensemble Kalman filter: theoretical formulation and practical implementation. Ocean Dyn 53:343–367

Gaspari G, Cohn SE (1999) Construction of correlation functions in two and three dimensions. Q J R Meteorol Soc 125:723–757

Gijsbers P, Hummel S, Vaneçek S, Groos J, Harper A, Knapen R, Gregersen J, Schade P, Antonello A, Donchyts G (2010) From OpenMI 1.4 to 2.0. In: Swayne DA, Yang W, Voinov AA, Rizzoli A, Filatova T (eds) International environmental modelling and software society (iEMSs) 2010 international congress on environmental modelling and software modelling for environments sake, fifth biennial meeting. Ottawa, Canada. http://www.iemss.org/iemss2010/index.php?n=Main.Proceedings

Gregersen J, Gijsbers P, Westen S (2007) Openmi: open modelling interface. J Hydroinformatics 9(3):175–191

Hamill TM (2000) Interpretation of rank histograms for verifying ensemble forecasts. Mon Wea Rev 129:550–560

Hamill TM, Whitaker JS, Snyder C (2001) Distance-dependent filtering of background error covariance estimates in an ensemble Kalman filter. Mon Wea Rev 129:2776–2790

Heemink AW, Hanea RG, Sumihar J, Roest M, Velzen N, Verlaan M (2010) Data assimilation algorithms for numerical models. Advanced computational methods in science and engineering. In: Koren B, Vuik K (eds) Lecture notes in computational science and engineering. Springer, pp 107–142. ISBN 978-3-642-03343-8

Heemink A, velzen N, Verlaan M, Altaf MU, Beckers JM, Barth A, Van Leeuwen PJ, Nerger L, Brasseur P, Brankart JM, de Mey P, Bertino L (2012) Deliverable 1.1: first identification of common tools; SANGOMA: stochastic assimilation for the next generation ocean model applications. http://www.data-assimilation.net/Documents/sangomaDL1.1.pdf

Hoteit I, Pham DT, Blum J (2002) A simplified reduced order Kalman filtering and application to altimetric data assimilation in Tropical Pacific. J Mar Syst 36:101–127

Houtekamer PL, Mitchell HL (1998) Data assimilation using an ensemble Kalman filter technique. Mon Wea Rev 126:796–811

Kalman R (1960) A new approach to linear filtering and prediction problems. Trans ASME, Ser D J Basic Eng 82:35–45

Le Hénaff PDMM, Marsaleix P (2009) Assessment of observational networks with the representer matrix spectra method—application to a 3-d coastal model of the bay of biscay. Ocean Dyn 59:3–20

Madec G (2014) NEMO ocean engine. Note du Ple de modlisation, Institut Pierre-Simon Laplace (IPSL), France, No 27 ISSN No, pp 1288–1619

Nerger L, Hiller W (2013) Software for ensemble-based data assimilation systems—implementation strategies and scalability. Comput Geosci 55:110–118

OpenDA (2013) The OpenDA data-assimilation toolbox. http://www.openda.org

Raeder K, Anderson JL, Collins N, Hoar TJ, Kay JE, Lauritzen PH, Pincus R (2012) Dart/cam: an ensemble data assimilation for cesm atmospheric models. J Clim 25:6304–6317

Ridler ME, Velzen Nv, Sandholt I, Falk AK, Heemink A, Madsen H (2014) Data assimilation framework: linking an open data assimilation library (openda) to a widely adopted model interface (openmi). Environ Model Softw 57:76–89

Sestoft P (2010) Numeric performance in C, C and Java. http://www.itu.dk/sestoft/papers/numericperformance.pdf

Tippett MK, Anderson JL, Bishop CH, Hamill TM, Whitaker JS (2003) Ensemble square root filters. Mon Wea Rev 131:1485–1490

van Velzen N, Segers aJ (2010) A problem-solving environment for data assimilation in air quality modelling. Environ Model Softw 25(3):277–288. doi:10.1016/j.envsoft.2009.08.008. http://linkinghub.elsevier.com/retrieve/pii/S136481520900231X

van Velzen N, Verlaan M (2007) COSTA a problem solving environment for data assimilation applied for hydrodynamical modelling. Meteorol Z 16(6):777–793

van Velzen N, Verlaan M, Bos E (2012) The OpenDA association annual report. http://www.openda.org/association/Annual_report_2012_v6.pdf

Verlaan M, Heemink AW (1997) Tidal flow forecasting using reduced rank square root filters. Stoch Hydrol Hydraul 11:349–368

Weerts AH, El Serafy GY, Hummel S, Dhondia J, Gerritsen H (2010) Application of generic data assimilation tools (datools) for flood forecasting purposes. Comput Geosci 36(4):453–463

Whitaker JS, Hamill TM (2002) Ensemble data assimilation without perturbed observations. Mon Wea Rev 130:1913–1924

Yan Y, Barth A, Beckers JM (2014) Comparison of different assimilation schemes in a sequential Kalman filter assimilation system. Ocean Modell 73:123–137

Zhang Y, Oliver DS (2010) Improving the ensemble estimate of the Kalman gain by bootstrap sampling. Math Geosci 42(3):327–345

Zhang Y, Oliver DS (2011) Evaluation and error analysis: Kalman gain regularisation versus covariance regularisation. Comput Geosci 15:489–508