J Sci Comput (2014) 61:239–257
DOI 10.1007/s10915-014-9824-2

A Scalable Approach for Variational Data Assimilation

Luisa D'Amore · Rossella Arcucci · Luisa Carracciuolo · Almerico Murli

Received: 9 December 2013 / Revised: 19 January 2014 / Accepted: 25 January 2014 /
Published online: 8 February 2014

Abstract Data assimilation (DA) is a methodology for combining mathematical models simulating complex systems (the background knowledge) and measurements (the reality or observational data) in order to improve the estimate of the system state (the forecast). The DA is an inverse and ill posed problem usually used to handle a huge amount of data, so, it is a large and computationally expensive problem. Here we focus on scalable methods that makes DA applications feasible for a huge number of background data and observations. We present a scalable algorithm for solving variational DA which is highly parallel. We provide a mathematical formalization of this approach and we also study the performance of the resulted algorithm.

Keywords Data assimilation · Inverse problem · Parallel computing · Scalable algorithm

Mathematics Subject Classification 65Y05 · 65J22 · 68W10 · 68U20

L. D'Amore (✉) · R. Arcucci
Department of Mathematics and Applications,
University of Naples Federico II, Naples, Italy
e-mail: luisa.damore@unina.it

R. Arcucci
e-mail: rossella.arcucci@unina.it

L. Carracciuolo
Istituto di Chimica e Tecnologia dei Polimeri, CNR, Naples, Italy
e-mail: luisa.carracciuolo@na.icar.cnr.it

A. Murli
SPACI c/o Department of Mathematics and Applications,
University of Naples Federico II, Naples, Italy
e-mail: almerico.murli@unina.it

1 Introduction and Motivations

Data assimilation (DA) has long been playing a crucial role in numerical weather prediction and oceanography [15,22,38] and more in general, in climate science; recently, DA is also applied to numerical simulations of geophysical applications [3], medicine and biological science [9,42] for improving the reliability of the numerical simulations.

Data assimilation (DA) is an ill posed inverse problem [31] and, today, there are a lot of DA algorithms. Two main methods gained acceptance as powerful methods for data assimilation in the last decennium: the variational approach and the Kalman Filter. The variational approach [29] is based on the minimization of a functional which estimate the discrepancy between numerical results and measures. The Kalman Filter [21] is a recursive filtering instead. Recently, were developed algorithms (see [2]) which are an intermediate between the two approaches, i.e. the back and forth nudging algorithm, consisting in adding to the equation of the model a relaxation term that fits the model to the observations. Most significant feature of such approaches is the very large computational burden required in concrete scenarios [32]. The parallel processing is necessary for the numerical solution of these problems, but it is not sufficient. These large-scale problems are computationally difficult and their solution requires designing of *scalable* approaches. Many factors contribute to scalability, including the architecture of the parallel computer and the parallel implementation of the algorithm. However, one important issue is the scalability of the algorithm itself. Here, scalability refers to the capability of the algorithm to [23]:

1. exploit performance of emerging computing architectures in order to get a solution in a real time (strong scaling),
2. use additional computational resources effectively to solve increasingly larger problems (weak scaling).

As claimed in [13], problem partitioning (decomposability: to break the problem into small enough independent less complex subproblems) is a universal source of scalable parallelism; the approach we introduce here meets the following demand: *parallelization should be considered from the beginning* [1,36].

1.1 The Present Work

A research collaboration between us and CMCC (Centro Euro Mediterraneo per i Cambiamenti Climatici) give us the opportunity to use the DA software called OceanVar [8,11]. OceanVar is based on a three dimensional variational scheme (3D-Var) and it is used in Italy to combine observational data (Sea level anomaly, sea-surface temperatures, etc.) with backgrounds produced by computational models of ocean currents for the Mediterranean Sea (namely, the NEMO framework, [37]), within the Mediterranean Forecasting System (MFS). The final aim of this research is to integrate the existing software code of OceanVar with NEMO software into a high performance computing environment able to take advantage of the emerging architectures, while fulfilling the requirements of its configuration coupled with the existing parallel forecast model, within the MFS.

Starting from this demand, we present a scalable approach for variational DA: we introduce a decomposition of the global domain into sub domains. On these sub domains we define local 3D-Var functionals and we prove that the minimum of the global 3D-Var functional can be obtained by collecting the minimum of each local functional. The (global) problem is decomposed into (local) sub problems in such a way. The resulted algorithm consists of several copies of a slightly modified 3D-Var algorithm, each one requiring approximately the

same amount of computations on each sub domain and an exchange of boundary conditions between adjacent sub domains. The data is flowing across the surfaces, the so-called *surface-to-volume* effect is produced [4,14].

We perform a feasibility analysis of the algorithm, analyzing its execution time and scalability. To this aim we introduce the scale-up factor as performance metric of the algorithm.

In conclusion, the benefits of this approach are the following:

1. efficiency: the subproblems are computationally easier (smaller) than the original;
2. parallelism: the subproblems can be solved concurrently;
3. design of innovative numerical algorithms (ensamble-based variational data assimilation EnDA).

Our approach allows to tackle the ill conditioning of DA inverse and ill posed problem without reducing the number of available data furthermore, as discussed in [20].

The paper is organized as follows. In Sect. 2 we review main results related to our work. In Sect. 3 we introduce the domain decomposition and the functional defined on this decomposition by using restriction and extension operators. In Sect. 4 we introduce the three dimensional variational data assimilation problem. It will be employed in our testing problem discussed in Sect. 5. Conclusions are drawn in Sect. 6.

2 Related Works

During the last 20 years, parallel algorithms for data assimilations reached a widespread interests at many federal research institutes as well as at many universities [NCAR (National Center for Atmospheric Research), NCEP (National Centers for Environmental Prediction), DWD (Deutscher Wetterdienst), UK Met Office, JMA (Japan Meteorological Agency), CMC (Canadian Association of Management Consultants) and the CMCC (Centro Euro-Mediterraneo per i Cambiamenti Climatici)].

Good scalability of the data assimilation system is necessary to make these applications feasible. Sequential data assimilation methods based on ensemble forecasts (an example is the ensemble based Kalman Filters) provide scalability, because the forecast of each ensemble member can be performed independently. The ensemble approach has to be combined with the parallelization of the numerical model and the data assimilation algorithm yet. In order to simplify the implementation of scalable data assimilation systems based on existing numerical models, the Parallel Data Assimilation framework (PDAF) has been developed [40]. PDAF provides several parallel DA algorithms. Ensemble filters like the Local Ensemble Transform Kalman Filter (LETKF) and the Error Subspace Transform Kalman Filter (ESTKF) are included.

The present work is an effort to the development of variational data assimilation scalable algorithms.

3 Preliminaries

In this section, we explain the mathematical setting of the decomposition of the space and the functions. We also state some notations, we will need to use it later.

Definition 1 (*Domain decomposition*) Let $\Omega \subset \Re^N$ be decomposed into a sequence of overlapping sub domains $\Omega_i \subset \Re^{r_i}$, $r_i \leq N$, $i = 1, \ldots, p$ such that

$$\Omega = \bigcup_{i=1}^{p} \Omega_i \tag{1}$$

where

$$\Omega_i \cap \Omega_j = \Omega_{ij} \neq \emptyset.$$

□

Let $t \in [0, T]$. Let f be a function belonging to the Hilbert space $\mathcal{K}([0, T] \times \Omega)$, that is:

$$f(t, x) : [0, T] \times \Omega \mapsto \Re$$

Associated with decomposition (1) we define the Restriction Operator:

Definition 2 (*Restriction operator*) Let us define:

$$RO_i : \mathcal{K}([0, T] \times \Omega) \mapsto \mathcal{K}([0, T] \times \Omega_i)$$

such that:

$$RO_i(f(t, x)) \equiv f(t, x), \quad (t, x) \in [0, T] \times \Omega_i .$$

Moreover, for simplicity of notations, we let:

$$f_i^{RO}(t, x) \equiv RO_i[f(t, x)].$$

□

In this respect we define the adjoint Extension Operator also. Given a set of p functions $g_i, i = 1, \ldots, p$, each belonging to the Hilbert space $\mathcal{K}([0, T] \times \Omega_i)$:

Definition 3 (*Extension operator*) Let us define:

$$EO_i : \mathcal{K}([0, T] \times \Omega_i) \mapsto \mathcal{K}([0, T] \times \Omega)$$

such that:

$$EO_i(g_i(t, x)) = \begin{cases} g_i(t, x) & x \in \Omega_i \\ 0 & elsewhere \end{cases} .$$

Moreover, for simplicity of notations, we let:

$$g_i^{EO}(t, x) \equiv EO_i[g_i(t, x)].$$

□

We observe that, for any function $f \in \mathcal{K}([0, T] \times \Omega)$, associated to the decomposition (1), it holds that

$$f(t, x) = \sum_{i=1,p} EO_i \left[f_i^{RO}(t, x) \right]. \tag{2}$$

Given p functions $g_i(t, x) \in \mathcal{K}([0, T] \times \Omega_i)$, the summation

$$\sum_{i=1,p} g_i^{EO}(t, x) \tag{3}$$

defines a function $g \in \mathcal{K}([0,T]\times\Omega)$ such that:

$$RO_j[g(t,x)] = RO_j\left[\sum_{i=1,p} g_i^{EO}(t,x)\right] = g_j(t,x).$$

We will use the same notation to denote the restriction/extension operators acting on points and vectors. If $\Omega = \{x_1, x_2, \ldots, \ldots, x_{NP}\} \subseteq \Re^{NP\times N}$ and $\mathbf{f} = (f(x_1), f(x_2), \ldots, f(x_{NP})) \in \Re^{NP}$, where $f : \Re^N \mapsto \Re$, let us assume that Ω can be decomposed into a sequence of $p \geq 1$ overlapping sub domains Ω_i such that

$$\Omega = \bigcup_{i=1}^{p} \Omega_i$$

where $\Omega_i \subset \Re^{r_i\times N}$ and $r_i \leq NP$. Hence

$$RO_i(\mathbf{f}) \equiv \mathbf{f}^{\mathbf{RO_i}} \equiv (f(x_i))_{x_i\in\Omega_i}, \quad \mathbf{f}^{\mathbf{RO_i}} \in \Re^{r_i}.$$

In this respect we define the adjoint Extension Operator also. If $\mathbf{g} = (g(z_i))_{z_i\in\Omega_i}$, it is

$$EO_i(\mathbf{g}) = \begin{cases} g(z_k) & z_k \in \Omega_i \\ 0 & elsewhere \end{cases}$$

and $EO_i(\mathbf{z}) \equiv \mathbf{g}^{EO_\mathbf{i}} \in \Re^{NP}$.
Let $\mathbf{w} \in \Re^{NP}$, and let $\mathbf{C}(\mathbf{w})$ denote the matrix such that:

$$\mathbf{C}(\mathbf{w}) = \mathbf{w}\mathbf{w}^T.$$

Associated to the domain decomposition (1), we define:

Definition 4 (*Covariance matrix decomposition*) Let $\mathbf{C}(\mathbf{w}) \in \Re^{NP\times NP}$ be the covariance matrix of a random vector $\mathbf{w} = (w_1, w_2, \ldots, w_{NP}) \in \Re^{NP}$, that is coefficient $c_{i,j}$ of $\mathbf{C}$ is $c_{i,j} = \sigma_{ij} \equiv Cov(w_i, w_j)$. Let $s < NP$, we define the restriction operator RO_s onto $\mathbf{C}(\mathbf{w})$ as follows:

$$RO_s : \mathbf{C}(\mathbf{w}) \in \Re^{NP\times NP} \mapsto RO_s[\mathbf{C}(\mathbf{w})] \overbrace{=}^{def} \mathbf{C}(\mathbf{w}^{RO_\mathbf{s}}) \in \Re^{s\times s}$$

i.e., it is the covariance matrix defined on the restriction $\mathbf{w}^{RO_\mathbf{s}}$ of $\mathbf{w}$. □

Hereafter, we refer to $\mathbf{C}(\mathbf{w}^{RO_\mathbf{s}})$ using the notation $\mathbf{C_s}$.

4 The DA Mathematical Model

In this section we define the DA inverse problem and the three dimensional variational (3D-Var) DA problem. Finally, we prove the main result: we obtain the minimum of the global functional (defined on the entire domain) as a piecewise function by collecting the minimum of each local functional (defined on a sub domain).

We underline that the assumptions we will do (for example the assumption concerning the covariance matrices) come out from applications of the DA on the real problems. Regarding the numerical approach, we refer to the 3D-Var algorithm as implemented in OceanVar [11] instead. Let $t \in [0,T]$, $x \in \Omega \subset \Re^N$ and let $u(t,x)$ be the state evolution of a predictive

system from time $t - \Delta t$ to time t, governed by the mathematical model $\mathcal{M}_t[u(t,x)]$. So, it is

$$\mathcal{M}_{t-\Delta t,t}: \quad u(t-\Delta t, x) \mapsto u(t,x). \tag{4}$$

At each time, the system state at time t depends on the initial condition.

Let:

$$v(t,y) = \mathcal{H}(u(t,y)), \quad y \in \Omega, \tag{5}$$

denote the observations mapping, where $\mathcal{H}$ is a given nonlinear operator that includes transformations and grid interpolations.

The aim of the DA problem is to find an optimal tradeoff between the current estimate of the system state (background) and the available observations at that time. More precisely, according to the practical applications of model-based assimilation of observations, we will use the following definition of DA.

Let $\{t_k\}_{k=0,1,\ldots,}$ be a discretization of the interval $[0,T]$, where $t_k = t_0 + k\Delta t$, and let $D_{NP}(\Omega) = \{(x_j)\}_{j=1,\ldots,NP} \in \Re^{NP\times N}$, be a discretization of $\Omega \subset \Re^N$, where $x_j \in \Omega$.

For each $k = 0, 1, \ldots$, we consider

- $\mathbf{u}_k^b = \{u_k^j\}_{j=1,\ldots,NP}^b \equiv \{u(t_k, x_j)^b\}_{j=1,\ldots,NP} \in \Re^{NP}$: (background) numerical solution of the model $\mathcal{M}_t[u(t,x)]$ on $\{t_k\} \times D_{NP}(\Omega)$;
- $\mathbf{v}_k = \{v(t_k, y_j)\}_{j=1,\ldots,nobs}$: the vector values of the observations on $y_j \in \Omega$ at time t_k;
- $\mathcal{H}(x) \simeq \mathcal{H}(y) + \mathbf{H}(x-y)$: a linearization of $\mathcal{H}$, where $\mathbf{H} \in \Re^{NP\times nobs}$ is the matrix obtained by the first order approximation of the Jacobian of $\mathcal{H}$ and $nobs \ll NP$;
- $\mathbf{R}$ and $\mathbf{B}$ the covariance matrices of the errors on the observations and on the background, respectively. These matrices are symmetric and positive definite.

Definition 5 (*The DA inverse problem*) The DA inverse problem is to compute the vector $\mathbf{u}_k^{DA} = \{u_k^j\}_{j=1,\ldots,NP}^{DA} \in \Re^{NP}$ such that:

$$\mathbf{v}_k = \mathbf{H}[\mathbf{u}_k^{DA}]$$

Since $\mathbf{H}$ is typically rank deficient and highly ill conditioned, DA is an ill posed inverse problem [31]. The Tikhonov-regularized formulation leads to an unconstrained least square problem [44]. The weighted background term acts as a regularization term, and it ensures the existence of a solution with reduced sensitivity [26,27]. In the following we let time t_k be fixed, i.e. we consider the so-called 3D-Var DA problem, then for simplicity of notations, we refer to $\mathbf{u}_k^b$ and $\mathbf{u}_k^{DA}$ omitting index k.

Definition 6 (*The 3D-Var DA problem*) 3D Variational DA problem is to compute the vector $\mathbf{u}^{DA}$ such that

$$\mathbf{u}^{DA} = argmin_{\mathbf{u}\in\Re^{NP}} J(\mathbf{u}) = argmin_u \left\{ \|\mathbf{H}\mathbf{u} - \mathbf{v}\|_{\mathbf{R}}^2 + \lambda \|\mathbf{u} - \mathbf{u}^b\|_{\mathbf{B}}^2 \right\} \tag{6}$$

where λ is the regularization parameter, while $\|\cdot\|_{\mathbf{B}}$ and $\|\cdot\|_{\mathbf{R}}$ denote the weighted euclidean norm on $\Re^{NP}$. □

So, the 3D-Var operator is:

$$J(\mathbf{u}) = (\mathbf{H}\mathbf{u} - \mathbf{v})^T \mathbf{R} (\mathbf{H}\mathbf{u} - \mathbf{v}) + \lambda (\mathbf{u} - \mathbf{u}^b)^T \mathbf{B} (\mathbf{u} - \mathbf{u}^b). \tag{7}$$

It depends on $\mathbf{u}$, $\mathbf{R}$, $\mathbf{B}$, and $D_{NP}(\Omega)$, then in order to emphasize these relationship, we write $J(\mathbf{u}, \mathbf{R}, \mathbf{B}, D_{NP}(\Omega))$.

Remark The 3D-Var operator depends on the regularization parameter, too. As is evident, when the regularization parameter λ tends to zero the regularized problem tends to the DA (ill posed) inverse problem, while larger values are expected to produce more stable solutions but with less fidelity to the observations and background state. Furthermore, we see that reducing the regularization parameter has the effect of increasing the uncertainty in the background, which means the data assimilation scheme trusts the observation term more. Conversely, increasing the regularization parameter has the effect of decreasing the uncertainty in the background which means the data assimilation scheme trusts the background term more. Therefore, the regularization parameter plays an important trade-off role.

The choice of a good regularization parameter is one of the most important issues in solving inverse problems and there exists a significant amount of research in the literature on the development of appropriate strategies for selecting regularization parameters. Parameter choice methods can be classified according to the input they require. There are two basic types

- a-priori methods, requiring information about the noise level on data. The discrepancy principle, developed and analyzed by Morozov [35] is the oldest one of them.
- data-driven methods, require no extra information. The Generalized cross-validation (GCV), due to Wahba [16], is one of the most popular methods.

Of course, latter methods are more expensive than the former ones, which are mainly used in operational data assimilation. Recently [5,12,45], some experiments focusing on the regularization parameter estimation based on data-driven methods (L-Curve and GCV) have been performed with the aim of analyzing the improvement of the computed result by variational data assimilation. In general, operational DA software assume $\lambda = 1$. By choosing $\lambda = 1$ can be considered as giving the same relative weight to the observations in comparison to the background state. Here, following the OceanVar and NEMOVAR software, we let $\lambda = 1$.

Associate to the decomposition given in Sect. 2, we give the following:

Definition 7 (*Functional restriction*) We generalize the definition of the restriction operator RO_i acting on J, as follows:

$$RO_i : J(\mathbf{u}, \mathbf{R}, \mathbf{B}, D_{NP}(\Omega)) \mapsto J(\mathbf{u}^{RO_i}, RO_i[\mathbf{R}], RO_i[\mathbf{B}], \Omega_i)$$

□

For simplicity of notations, and to underline that the restriction operator is associated to the domain decomposition given in Sect. 2 we let:

$$J(\mathbf{u}^{RO_i}, RO_i[\mathbf{R}], RO_i[\mathbf{B}], \Omega_i) \equiv J_{\Omega_i}$$

Definition 8 (*Functional extension*) We generalize the definition of the extension operator EO_i acting on J as

$$EO_i : J_{\Omega_i} \mapsto J_{\Omega_i}^{EO_i},$$

where

$$EO_i[J_{\Omega_i}] = \begin{cases} J_{\Omega_i} & x \in \Omega_i \\ 0 & elsewhere \end{cases}$$

□

We observe that

$$J(\mathbf{u}, \mathbf{R}, \mathbf{B}, D_{NP}(\Omega)) = \sum_{i=1,p} J_{\Omega_i}^{EO_i}. \tag{8}$$

We are now able to define the *local* 3D-Var regularization functionals. A local 3D-Var regularization functional describes the local DA problem on a sub domain Ω_i of the domain decomposition. It is obtained applying the restriction operator to the 3D-Var regularization functional J and by adding a *local* constraint to such restriction then. This is in order to enforce the continuity of each solution of the local DA problem onto the overlap region between adjacent domains Ω_i and Ω_j.

Definition 9 (*Local 3D-Var regularization functional*) Let us introduce the operator J_i defined as follows:

$$J_i(\mathbf{u}^{\mathbf{RO_i}}) = \|\mathbf{H_i}\mathbf{u}^{\mathbf{RO_i}} - \mathbf{v}^{\mathbf{RO_i}}\|^2_{\mathbf{R_i}} + \|\mathbf{u}^{\mathbf{RO_i}} - (\mathbf{u^b})^{\mathbf{RO_i}}\|^2_{\mathbf{B_i}} + \mu\|\mathbf{u}^{\mathbf{RO_i}}/\Omega_{ij} - \mathbf{u}^{\mathbf{RO_j}}/\Omega_{ij}\|^2_{\mathbf{B_{ij}}} \tag{9}$$

where $\mathbf{v}^{\mathbf{RO_i}}$, $\mathbf{u}^{\mathbf{RO_i}}$, $\mathbf{R_i}$, $\mathbf{B_i}$, $\mathbf{H}_i$ and $(\mathbf{u^b})^{\mathbf{RO_i}}$ are restrictions on Ω_i of vectors and matrices in (7), and $\mathbf{u}^{\mathbf{RO_i}}/\Omega_{ij}$, $\mathbf{u}^{\mathbf{RO_j}}/\Omega_{ij}$, $\mathbf{B_{ij}}$ are restriction on $\Omega_{ij} = \Omega_i \cap \Omega_j$ of the quantities defined in (7). Parameter μ is the regularization parameter. □

Parameter μ is equal to 1: it ensures the continuity of each solution of the local DA problem onto the overlap region between adjacent domains while keeping it sufficiently close to observations and background state, with the same relative weight.
We observe that each J_i is quadratic (and convex) then its minimum is unique. Let

$$\mathbf{u}_i^{DA} = argmin_{\mathbf{u}} J_i(\mathbf{u}). \tag{10}$$

We are able to prove the following result:

Theorem 1 *If*

$$\Omega = \bigcup_{i=1,p} \Omega_i$$

is a decomposition of Ω then, under the assumptions of Sect. 2, let

$$\widetilde{\mathbf{u}}^{DA} \overbrace{=}^{DEF} \sum_{i=1,p} (\mathbf{u}_i^{EO_i})^{DA}, \tag{11}$$

then it follows that:

$$\widetilde{\mathbf{u}}^{DA} = \mathbf{u}^{DA}.$$

Proof The functional J as well as all the functionals J_i, are quadratic (hence, convex), so their unique minimum, $\mathbf{u}^{DA}$ and $\mathbf{u}_i^{DA}$, respectively, are obtained as zero of their gradients, i.e.:

$$\nabla J[\mathbf{u}^{DA}] = 0 \quad , \quad \nabla J_i[\mathbf{u}_i^{DA}] = 0. \tag{12}$$

From the Definition 9 of J_i it is

$$\nabla J_i[\mathbf{u}_i^{DA}] = \nabla J_{\Omega_i}[\mathbf{u}_i^{DA}]. \tag{13}$$

By applying the extension operator EO_i (see Definition 3) to $\mathbf{u}_i^{DA}$, instead of $\mathbf{u}_i^{DA}$, we may consider the vector $(\mathbf{u}_i^{EO_i})^{DA}$. From the Definition of the extension operator EO_i it is:

$$\mathbf{u}_i^{DA} = \sum_j (\mathbf{u}_i^{EO_i})^{DA}, \quad on \quad \Omega_i. \tag{14}$$

Applying the extension operator EO_i to J_{Ω_i}, and from the (12), the (13), and the (14), it follows that

$$0 = \nabla J_{\Omega_i}(\mathbf{u}_i^{DA}) = \nabla J_{\Omega_i}^{EO_i}\left(\sum_j (\mathbf{u}_j^{EO_i})^{DA}\right). \tag{15}$$

From (8), by summing each equation in (15) for $i = 1, \ldots, p$ on all sub domains Ω_i and from (11) it follows that:

$$\sum_i \nabla J_{\Omega_i}^{EO_i}\left(\sum_j (\mathbf{u}_j^{EO_i})^{DA}\right) = 0 \Leftrightarrow \sum_i \nabla J_{\Omega_i}^{EO_i}(\tilde{\mathbf{u}}^{DA}) = 0. \tag{16}$$

Thanks to the linearity of the gradients of J_{Ω_i}, it is

$$\sum_i \nabla J_{\Omega_i}^{EO_i}(\tilde{\mathbf{u}}^{DA}) = \nabla \sum_i J_{\Omega_i}^{EO_i}(\tilde{\mathbf{u}}^{DA}) = \tag{17}$$

where the last term in (17) is

$$= \nabla J(\tilde{\mathbf{u}}^{DA}). \tag{18}$$

Hence, from the (16), the (17) and the (18) it follows

$$\sum_i \nabla J_{\Omega_i}(\tilde{\mathbf{u}}^{DA}) = 0 \Leftrightarrow \nabla J(\tilde{\mathbf{u}}^{DA}) = 0.$$

Finally,

$$\nabla J(\tilde{\mathbf{u}}^{DA}) = 0 \Rightarrow \tilde{\mathbf{u}}^{DA} \equiv \mathbf{u}^{DA},$$

where the last equality holds because the minimum of J is unique. □

This result ensures that $\mathbf{u}^{DA}$ (the minimum of J) can be obtained by patching together all the vectors $\mathbf{u}_i^{DA}$ (minimum of the operators J_{Ω_i}), i.e. by using the domain decomposition, the global minimum of the operator J can be obtained by patching together the minimum of the *local* functionals J_{Ω_i}. This result has important implications from the computational viewpoint as it will be explained in the next section.

4.1 The Preconditioned 3D-Var Functional

The ill conditioning of the DA inverse problem (i.e. the sensitivity of the analysis to small perturbations in the data), depends on the conditioning of the Hessian of J. Small errors in the Hessian lead to large errors in its inverse, so the computed solution to the DA problem may be very inaccurate. In designing of the DA schemes, it is important to ensure that the conditioning of the Hessian is as small as possible, or it is essential to use preconditioning techniques to improve the conditioning.

As implemented in OceanVar software, we consider the incremental formulation of the preconditioned local 3D-Var functionals (see [6] and [7]). More precisely, to compute the preconditioner matrix $\mathbf{V_i}$ we apply a spectral decomposition to $\mathbf{B_i}$. So, we get

$$\mathbf{B_i} = \mathbf{U_i D_i U_i^T} = \mathbf{U_i D_i}^{\frac{1}{2}} \mathbf{D_i}^{\frac{1}{2}} \mathbf{U_i}^T = (\mathbf{U_i D_i}^{\frac{1}{2}})(\mathbf{U_i D_i}^{\frac{1}{2}})^T$$

and by posing $\mathbf{V_i} = \mathbf{U_i D_i}^{\frac{1}{2}}$ we get $\mathbf{V_i}$ such that

$$\mathbf{B_i} = \mathbf{V_i V_i}^T.$$

Let

$$\mathbf{d_i} = [\mathbf{v_i} - \mathbf{H_i}(\mathbf{u})^{RO_i}]$$

be the *misfit*, and let $\delta \mathbf{u}^{DA_i} = (\mathbf{u}^{\mathbf{DA_i}})^{\mathbf{RO_i}} - (\mathbf{u}^{\mathbf{b}})^{\mathbf{RO_i}}$ be the increment. By setting $\mathbf{w}_i = \mathbf{V}_i^T \delta \mathbf{u}^{DA_i}$, the cost function becomes

$$\begin{aligned} \mathbf{J_i}(\mathbf{w_i}) = & \frac{1}{2} \mathbf{w_i}^T \mathbf{w_i} + \frac{1}{2} (\mathbf{H_i V_i w_i} - \mathbf{d_i})^T \mathbf{R_i}^{-1} (\mathbf{H_i V_i w_i} - \mathbf{d_i}) \\ & + \frac{1}{2} \left(\mathbf{V_{ij} w_i}^+ - \mathbf{V_{ij} w_i}^- \right)^T \left(\mathbf{V_{ij} w_i}^+ - \mathbf{V_{ij} w_i}^- \right) \end{aligned} \tag{19}$$

where

$$\mathbf{w_i}^+ = \mathbf{w_i} \quad on\ \Omega_{ij} \qquad \mathbf{w_i}^- = \mathbf{w_j} \quad on\ \Omega_{ij}.$$

This scheme is the so-called incremental formulation of the preconditioned 3D-Var: the current estimate of the solution is updated with its computed increment.[1] Each linearised cost function is minimised in an inner-loop using iterative gradient methods. The minimiser is then used in an outer-loop step to update the current best estimate of the analysis. Generally only few iterations of the outer loops are performed. Hereafter, we focus on the inner-loop minimization which is the most expensive kernel in terms of time consuming.

As implemented in OceanVar software, each linearized preconditioned local functional is minimized using the L-BFGS (Limited-Broyden–Fletcher–Goldfarb Shanno) method[2] (see [41]), so, to compute the minimum, we need to compute the gradient of (19),

$$\nabla \mathbf{J_i}(\mathbf{w_i}) = \mathbf{w_i} + \mathbf{V_i}^T \mathbf{H_i}^T \mathbf{R_i}^{-1} (\mathbf{H_i V_i w_i} - \mathbf{d_i}). \tag{20}$$

The convergence rate of L-BFGS depends on the conditioning of the numerical problem, i.e. it depends on the condition number of the preconditioned Hessian of J_i [20]:

$$\mathbf{A}_{J_i} = \left(\mathbf{I} + \mathbf{B_i}^{1/2} \mathbf{H_i}^T \mathbf{R_i}^{-1} \mathbf{H_i B_i}^{1/2} \right)$$

[1] It was shown [28] that incremental 3D-Var is equivalent to a GaussNewton method [10] (i.e. an approximation to a Newton iteration, in which the second-order terms of the Hessian are neglected) applied to the full nonlinear regularization functional.

[2] The idea of using the L-BFGS method in variational data assimilation is not new because of its modest storage requirements and its high performance on large-scale unconstrained convex minimization [39,46]. L-BFGS method is a Quasi–Newton method that can be viewed as extension of conjugate-gradient methods in which the addition of some modest storage serves to accelerate the convergence rate. The L-BFGS update formula generates the matrices approximating the Hessian using information from the last m Quasi-Newton iterations, where m is determined by the user (generally $3 \leq m \leq 30$). After having used the m vector storage locations for m quasi-Newton updates, the approximation of the Hessian matrix is updated by dropping the oldest information and replacing it by the newest information. Hence, time complexity of the L-BFGS increases linearly with the size of the m vectors [30].

In [17] the authors have established that the condition number is significantly reduced by such preconditioning, assuming that:

1. the correlation structures of $\mathbf{B}$ are homogeneous and isotopic, the coefficients of $\mathbf{B}$ depend only on the distance between points, i.e. it is:

$$\mathbf{B} = \sigma_b^2 \mathbf{C}$$

where

$$c_{ij} = \rho^{|i-j|^2}, \quad \rho = exp\left(\frac{-\Delta x^2}{2L^2}\right), \quad |i-j| < N/2$$

N is the size of the domain, $\mathbf{C}$ denotes the Gaussian correlation structure of the background errors while σ_b^2 is the background error variance. As a consequence:

$$\mu(\mathbf{B}) = \mu(\mathbf{C})$$

2. the error covariance matrix $\mathbf{R}$ is such that:

$$\mathbf{R} = \sigma_o^2 \mathbf{I}_{nobs}$$

where $nobs$ is the number of the observations and σ_o^2 is the observational error variance.

In our testing problems, under the same assumptions on $\mathbf{B}_i$ and $\mathbf{R}_i$ we experimentally found that the condition number of the preconditioned (local) Hessian matrix is about of order unity, and on each sub domain, the minimization of the local 3D-Var functionals requires about the same number of iterations (<10).

We observe that quantifying observation error correlations is not a straightforward problem and this is a research issue [20,43]. Secondly, even when good estimates of the errors can be made, there were conditioning problems with the minimization that had to be overcome [17–19,24,25]. Observation errors are usually assumed uncorrelated because diagonal matrices are simple and numerical efficient. However, it is often better to include an approximate correlation structure in the observation error covariance matrix than to incorrectly assume error independence, for example, by choosing a suitable matrix approximation (diagonal, block-diagonal, …). Here, following the OceanVar software, we choose the same correlation structure of the covariance matrices.

4.2 The Local 3D-Var Algorithm

Let $\mathcal{A}_{M3D}(r_i)$ be the notation for the algorithm used to compute the minimum of the local 3D-Var functional on $\Omega_i \subset \Re^{r_i \times N}$. Hereafter we describe the our $\mathcal{A}_{M3D}(r_i)$ algorithm: a modified 3D-Var algorithm on Ω_i.

Algorithm 1 *$\mathcal{A}_{M3D}(r_i)$: Modified 3D-Var algorithm on Ω_i, $i = 1, \ldots, p$*

1: **Input:** $\mathbf{v}_i$ and $(\mathbf{u^b})^{RO_i}$
2: **Define** $\mathbf{H_i}$
3: **Compute** $\mathbf{d_i} \leftarrow \mathbf{v}_i - \mathbf{H_i}(\mathbf{u^b})^{RO_i}$
4: **Define** $\mathbf{R_i}$ and $\mathbf{B_i}$
5: **Compute** the matrix $\mathbf{V_i}$ from $\mathbf{B_i}$
6: **Define** the initial value of $\mathbf{u}^{DA_i}$
7: **Compute** $\mathbf{w}_i \leftarrow \mathbf{V_i}^T \mathbf{u}^{DA_i}$
8: **repeat**

9: **Send and Receive** the boundary conditions from the adjacent domains
10: **Compute** $\mathbf{J_i} \leftarrow \mathbf{J_i}(\mathbf{w_i})$
11: **Compute** $\mathbf{gradJ_i} \leftarrow \nabla \mathbf{J_i}(\mathbf{w_i})$
12: **Compute** new values for $\mathbf{w}_i$ by the L-BFGS steps
13: **until** (Convergence on $\mathbf{w_i}$ is obtained)
14: **Compute** $\mathbf{u}_i^{DA} \leftarrow (\mathbf{u^b})^{RO_i} + \mathbf{V_i w_i}$

end

The convergence criterion used in the repeat-until loop, at each iteration **iter**, is

$$\frac{\mathbf{J_i(iter+1)} - \mathbf{J_i(iter)})}{\max\{|\mathbf{J_i(iter)}|, |\mathbf{J_i(iter+1)}|, 1\}} \leq TOL$$

where $TOL = 10^{-6}$ together with a maximum iteration number (about 10).

We observe that $\mathcal{A}_{M3D}(r_i)$ requires an exchange of the boundary conditions (local data communications) between adjacent sub domains only.

5 The Scalable Algorithm ant its Performance Analysis

At the first we analyze the time complexity of the $\mathcal{A}_{M3D}(r_i)$ algorithm. The number of floating point operations required at each step of the $\mathcal{A}_{M3D}(r_i)$ algorithm are:

- Step 3 and 14: $O(r_i^2 + r_i)$
- Step 5: $O(r_i^3)$
- Step 7: $O(r_i^2)$
- Step 10 and 11: $O(r_i^2 + r_i)$
- Step 12: $O(kmr_i)$ (k is the number of iterations of L-BFGS, m is the number of Quasi-Newton updates of L-BFGS, r_i is the size of each sub domain Ω_i; $k \ll r_i$ and $m \ll r_i$).

So, the time complexity for performing these floating point operations is $T(\mathcal{A}_{M3D}(r_i)) = O(p(r_i))$, where $p(r_i) \in \Pi_3$, that is a polynomial of degree 3.

At step 9, $\mathcal{A}_{M3D}(r_i)$ requires $O(r_i)$ data exchange between adjacent sub domains, so additional time is required for this step. This overhead can be estimated by using the so-called *surface-to-volume ratio*.

Definition 10 Let S/V be the surface-to-volume ratio. It is a measure of the amount of data exchange (proportional to surface area of domain) per unit operation (proportional to volume of domain).

The goal is to minimize this ratio. In case of algorithm $\mathcal{A}_{M3D}(r_i)$, we get the surface-to-volume ratio equals to

$$\frac{S}{V} = \frac{O(r_i)}{O(r_i^3)} = O\left(\frac{1}{r_i^2}\right)$$

and, as the size of each sub domain increases, amount of data exchange per unit computation decreases.

We now introduce the algorithm associated to the decomposition given in Sect. 2.

Definition 11 (*Scalable algorithm*) If $p \in \aleph$, $p \geq 1$, the algorithm associated to the decomposition given in Sect. 2 is

$$\mathcal{A}_{S3D}(NP, p) \overbrace{=}^{def} \underbrace{\{\mathcal{A}_{M3D}(r_1), \mathcal{A}_{M3D}(r_2), \ldots, \mathcal{A}_{M3D}(r_p)\}}_{p \ \ times}$$

where $\mathcal{A}_{M3D}(r_i)$ acts on Ω_i, subset of $D_{NP}(\Omega) \subset \Re^{NP}$.
We observe that if $p = 1$, then $i = 1, r_i = NP$ and the modified 3D-Var algorithm reduces to 3D-Var algorithm,

$$\mathcal{A}_{M3D}(r_i) \equiv \mathcal{A}_{S3D}(NP)$$

Definition 12 (*Scale up factor*) Let $p_1, p_2 \in \aleph$ and $p_1 < p_2$. Let $T(\mathcal{A}_{S3D}(NP, p_i))$, $i = 1, 2$ be the time complexity of $\mathcal{A}_{S3D}(NP, p_i)$, $i = 1, 2$. We name (relative) scale-up factor of $\mathcal{A}_{S3D}(NP, p_2)$, in going from p_1 to p_2, the following ratio:

$$S^{S3D}_{p_2, p_1}(NP) = \frac{T(\mathcal{A}_{S3D}(NP, p_1))}{(p_2/p_1)T(\mathcal{A}_{S3D}(NP, p_2))}.$$

The following result allows us to analyze the behaviour of the scale-up factor:

Proposition 1 *Let us consider an "uniform" domain decomposition, that is for $i = 1, \ldots, p$ it is $r_i = \frac{NP}{p} (\equiv r$, where r is a constant independent on i). Let us assume that $1 \leq r < \infty$.*

Let us assume that $p_1 = 1$ and let $p_2 = p$. Moreover, if $T(\mathcal{A}_{M3D}(NP))$ has a polynomial growth of degree $d > 1$, i.e.

$$T(\mathcal{A}_{M3D}(NP)) = a_d NP^d + a_{d-1} NP^{d-1} + \cdots + a_0, \quad a_d \neq 0$$

then scale up factor is

$$S^{S3D}_{p,1}(r) = \frac{T(\mathcal{A}_{M3D}(pr))}{pT(\mathcal{A}_{M3D}(r))} = \alpha(r, p)\, p^{d-1} \tag{21}$$

where

$$\alpha(r, p) = \frac{a_d + \frac{a_{d-1}}{pr} + \cdots + \frac{a_0}{(pr)^d}}{a_d + \frac{a_{d-1}}{r} + \cdots + \frac{a_0}{r^d}}$$

and, if $r = 1$

$$\alpha(1, p) = \beta \in]0, 1]$$

where, if $a_i = 0 \quad \forall i \in [0, d-1]$, then $\beta = 1$.

Finally,

$$\lim_{r \to \infty} \alpha(r, p) = 1.$$

Proof It holds

$$\begin{aligned} S^{S3D}_{p,1}(NP) &= \frac{a_d NP^d + a_{d-1} NP^{d-1} + \cdots + a_0}{p\left(a_d r^d + a_{d-1} r^{d-1} + \cdots + a_0\right)} \cdot r^d \cdot \frac{1}{r^d} = \frac{a_d \cdot p^d + a_{d-1} \frac{p^d}{NP} + \cdots + \frac{a_0 \cdot p^d}{NP^d}}{p \cdot \left(a_d + a_{d-1} \frac{p}{NP} + \cdots + \frac{a_0 p^d}{NP^d}\right)} \\ &= \frac{p^d \left(a_d + a_{d-1} \frac{1}{NP} + \cdots + \frac{a_0}{NP^d}\right)}{p \cdot \left(a_d + a_{d-1} \frac{p}{NP} + \cdots + \frac{a_0 p^d}{NP^d}\right)} = \frac{a_d + \frac{a_{d-1}}{pr} + \cdots + \frac{a_0}{(pr)^d}}{a_d + \frac{a_{d-1}}{r} + \cdots + \frac{a_0}{r^d}} \cdot p^{d-1} \end{aligned} \tag{22}$$

and (21) comes out. Note that $\alpha(r,\, p) > 0$. If $r = 1$,

$$\alpha(1,\, p) = \frac{a_d + a_{d-1}\frac{1}{p} + \cdots + \frac{a_0}{p^d}}{\sum_{k=0}^{k\le d} a_k} \le 1. \tag{23}$$

If $a_i = 0 \quad \forall i \in [0, d-1]$ then $\alpha(1,\, p) = \frac{a_d}{a_d} = 1$.

Finally, it is

$$\lim_{r\to\infty} \alpha(r,\, p) = \lim_{r\to\infty} \frac{a_d + \frac{a_{d-1}}{pr} + \cdots + \frac{a_0}{(pr)^d}}{a_d + \frac{a_{d-1}}{r} + \cdots + \frac{a_0}{r^d}} = 1.$$

We observe that, from (23), it is

$$\sum_{k=0}^{k\le d} a_k \ge a_d + a_{d-1}\frac{1}{NP} + \cdots + \frac{a_0}{NP^d}$$

and the offset increases as NP grows. In conclusion, if NP increases β grows too. □

In conclusion,

1. if NP is fixed and $p \sim NP$ (such as the so-called *strong scaling*), it follows that $r \sim 1$. We observe that in this case, $S_{p,1}^{S3D}(NP) \sim \beta\, NP^{d-1}$, where $\beta \le 1$; while the surface-to-volume ratio $S/V \sim 1$.
   Hence, the $\mathcal{A}_{S3D}(NP)$ algorithm has a poor (strong) scalability.
2. if $NP \to \infty$ and r is kept fixed, then p increases too (such as the so-called *weak scaling*). Observe that in this case, $S_{p,1}^{S3D}(NP) \sim \alpha(r,\, p)\, p^{d-1}$, where $\alpha(r,\, p) \sim 1$. The surface-to-volume ratio S/V the more decreases as r is large. Hence, the $\mathcal{A}_{S3D}(NP)$ algorithm has a good (weak) scalability.

6 The Scalable Algorithm on a Test Problem

6.1 Experimental Set-Up

For testing the $\mathcal{A}_{3D}(NP)$ algorithm we needed of

- $D_{NP}(\Omega) = \{((x_i,\, y_j) = (i,\, j)_{i=0,\ldots,n_x+1;\, j=0,\ldots,n_y+1}\}$, and $t_k \in [0, \infty[$;
- $\mathbf{u}_k^b \in \Re^{NP}$, the numerical solution of $\mathcal{M}_t[u(t, x)]$, given in [34] and computed in [34] on $D_{NP}(\Omega)$ and on t_k;
- $\mathbf{H}$: a piecewise linear interpolation operator whose coefficients are computed using the points of model domain nearest the observation values;
- Matrix $\mathbf{B}$:

$$\mathbf{B} = \sigma_b^2\mathbf{C}, \quad \sigma_b^2 = 0.5, \quad L = 1$$

- Matrix $\mathbf{R}$:

$$\mathbf{R} = \sigma_0^2\mathbf{I}, \quad \sigma_o^2 = 0.5$$

- $\mathbf{v}_k$: the observation values are obtained choosing (randomly) among the values of $\mathbf{u}_k^b$ and perturbing (randomly) these values; finally we set $nobs = \frac{NP}{3}$.

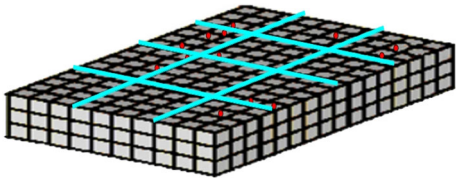

Fig. 1 Decomposition of $D_{NP}(\Omega)$: *points* are the observed data and lines are the overlapping regions

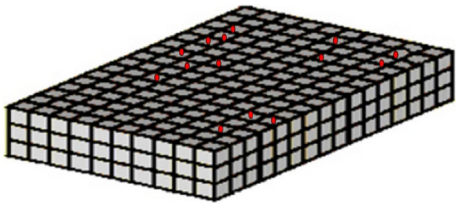

Fig. 2 Domain $D_{NP}(\Omega)$: *black points* are the observed data

6.2 The Mapping of the $\mathcal{A}_{S3D}(NP, p)$ Algorithm on the Parallel Computing Architecture

Let us consider the $\mathcal{A}_{S3D}(NP, p)$ algorithm running on a parallel architecture made of 8 distributed nodes (blades). Each node is a 8-core element (the cores share the same local memory). Here we use the notation *nproc* (*nproc* $\geq$ 8) referring to the processing elements.

Let $nproc = s \times q$. We assume a 2D uniform decomposition of $D_{NP}(\Omega)$ along the (x, y)-axis. The x-axis is divided by s and the y-axis by q (Figs. 1, 2). An implementation of $\mathcal{A}_{S3D}(NP, p)$ algorithm was obtained mapping each sub domain onto a processing element of the reference parallel architecture, i.e. we use the following correspondence

$$p \leftrightarrow nproc.$$

We assume that $NP = (n_x + 1) \times (n_y + 1) \times (n_z + 1)$ where $n_x = n_y = n$ and $n_z = 1$, then the size of each sub domain Ω_i is

$$r = \frac{NP}{nproc} = nloc_x \times nloc_y \times nloc_z$$

where:

$$nloc_x = \frac{n}{s} + 2o_x \ , \ nloc_y = \frac{n}{q} + 2o_y \ , \ nloc_z = 1. \tag{24}$$

These dimensions include the overlapping $(2o_x \times 2o_y)$.
By using n_{xpp} and n_{ypp} to denote the position of the $(1, 1)$ grid-point of each sub domain in the global domain, each element of the local array x^{loc} corresponds to the element of the global array x^{glob}, as following:

$$x^{glob}(i + n_{xpp} - 1; j + n_{ypp} - 1; k) = x^{loc}(i; j; k) \tag{25}$$

where $1 < i < nloc_x$, $1 < j < nloc_y$, and $1 < k < nloc_z$.

6.3 Experimental Results

The Table 1 shows the values of $\|\mathbf{u}^{DA} - \widetilde{\mathbf{u}}^{DA}\|_\infty$, where $\mathbf{u}^{DA}$ denotes the minimum of the 3D-Var (global) functional J on $D_{NP}(\Omega)$ while $\widetilde{\mathbf{u}}^{DA}$ is the sum of the minimum of the (local)

Table 1 Values of $\|\mathbf{u}^{DA} - \widetilde{\mathbf{u}}^{DA}\|_\infty$, for different values of *nproc* and of *n*

n	*nproc*	$\|\mathbf{u}^{DA} - \widetilde{\mathbf{u}}_{DA}\|_\infty$
64	$nproc = 1$	0.0000e−00
	$nproc = 2$	6.9418e−04
	$nproc = 4$	5.2801e−04
	$nproc = 8$	7.0604e−04
128	$nproc = 1$	0.0000e−00
	$nproc = 2$	7.7383e−04
	$nproc = 4$	1.3096e−03
	$nproc = 8$	1.7156e−03

3D-Var functional J_i on sub domain Ω_i. $\mathbf{u}^{DA}$ is computed by running $\mathcal{A}_{S3D}(NP, nproc)$ for $nproc = 1$. The results of Table 1 prove the reliability of the algorithm $\mathcal{A}_{S3D}(NP, nproc)$.

Last result specifies the scale-up factor of the $\mathcal{A}_{S3D}(NP, p)$ algorithm in our case study:

Corollary 2 *The scale-up factor of the* $\mathcal{A}_{S3D}(NP, p)$ *algorithm is*

$$S_{p,1}^{S3D}(NP) = \alpha(r, p)\, p^2. \tag{26}$$

Proof The time complexity of the $\mathcal{A}_{3D}(NP)$ algorithm is $T(NP) = O(p(NP))$ flops, on a problem of size NP, where $p(NP) \in \Pi_3$. Hence, the results follows from Proposition 1. □

Let t_{flop} be the time required by one floating point operation. The execution time of the $\mathcal{A}_{M3D}(NP)$ algorithm needed for performing $T(NP)$ floating point operations, is

$$T_{flop}(NP) = T(NP) \times t_{flop}.$$

Multiplying and dividing the (26) by t_{flop} we get

$$\alpha(r, p)p^2 = \frac{T_{flop}(NP)}{pT_{flop}(NP/p)}. \tag{27}$$

Let $T^{nproc}(NP)$ be the execution time of the $\mathcal{A}_{S3D}(NP, p)$ algorithm, measured in seconds, running on our parallel computing architecture. It is

$$T^{nproc}(NP) \underbrace{=}_{def} T_{flop}^{nproc}(NP) + T_{com}^{nproc}(NP)$$

where

- $T_{flop}^{nproc}(NP) \equiv T_{flop}(NP)$ is the computing time required for the execution of $T(NP)$ floating point operations;
- $T_{com}^{nproc}(NP)$ is the inter-node communication time of $T(NP)$ data.

In Table 2 we report $T^{nproc}(NP)$ and the measured values of S_{nproc,p_1}^{S3D}. We also compare the measured values of S_{nproc,p_1}^{S3D} with the values of S_{nproc,p_1}^{S3D}, given in the (26), where $p_1 = 8, 16$, $p_2 = nproc$ and $p = \frac{p_2}{p_1}$. Experimental results confirm the scalability analysis of the algorithm as discussed in Sect. 5.

We observe that measured values of scale-up factor are obtained as

$$\text{measured } S_{nproc,p_1}^{S3D} = \frac{T_{flop}^{nproc}(NP)}{pT_{flop}^{nproc}(NP/p) + pT_{com}^{nproc}(NP/p)},$$

Table 2 Execution time and *scale-up* factor for different values of $NP = 3n^2$ and *nproc*

n	*nproc*	$T^{nproc}(NP)$	Measured $S^{S3D}_{nproc,8}$	$S^{S3D}_{nproc,8}$
64	8	2.0545e+02	1.0	1
	16	3.1658e+01	3.25	4
	32	5.0012e+00	10.27	16
	64	1.0979e+00	23.39	64
n	*nproc*	$T^{nproc}(NP)$	Measured $S^{S3D}_{nproc,16}$	$S^{S3D}_{nproc,16}$
128	8	–	–	–
	16	3.9091e+03	1.0	1
	32	4.9976e+02	3.91	4
	64	6.8960e+01	14.17	16

hence, it follows that

$$\text{measured } S^{S3D}_{nproc,p_1} = \frac{\frac{T^{nproc}_{flop}(NP)}{pT^{nproc}_{flop}(NP/p)}}{1+\frac{T^{nproc}_{comm}(NP/p)}{T^{nproc}_{flop}(NP/p)}} < \frac{T_{flop}(NP)}{pT_{flop}(NP/p)} \equiv S^{S3D}_{nproc,p_1},$$

where the last equivalence follows from the (27).

7 Conclusions and Future Works

We presented an innovative algorithm for solving Variational Data Assimilation problem. The algorithm we considered starts from a decomposition of the physical domain; it uses a partitioning of the solution and a modified regularization functional describing the DA problem on the decomposition. We provided a mathematical formulation of the model and the algorithm. We also proved the mathematical validity of this formulation and we performed a feasibility analysis in terms of computational cost and of algorithmic scalability. In order to evaluate the performance of the parallel algorithm we introduced the scale-up factor which measure the performance gain in terms of time complexity reduction. We tested the model and the algorithm on a consistent test case (the shallow water equations). We are currently working on the deployment of this algorithm in a concrete scenario. Mainly, we are working on the variational DA systems used with the NEMO ocean model: OceanVar [11] and NEMOVAR [33]), including the extension to 4D-Var of this approach.

References

1. Antonelli, L., Carracciuolo, L., Ceccarelli, M., D'Amore, L., Murli, A.: Total Variation Regularization for Edge Preserving 3D SPECT Imaging in High Performance Computing Environments. Lecture Notes in Computer Science (LNCS), vol. 2330, pp. 171–180, Springer-Verlag, Berlin, Heidelberg (2002)
2. Auroux, D., Blum, J.: Back and forth nudging algorithm for data assimilation problems. C. R. Acad. Sci. Ser. I **340**(340), 873–878 (2005)
3. Blum, J., Le Dimet, F.X., Navon, I.M.: Data Assimilation for Geophysical Fluids, Volume XIV of Handbook of Numerical Analysis, chapter 9. Elsevier, Amsterdam (2005)
4. Carracciuolo, L., D'Amore, L., Murli, A.: Towards a parallel component for imaging in PETSc programming environment: a case study in 3-D echocardiography. Parallel Comput. **32**, 67–83 (2006)

5. Delahaies, S., Roulstone, L., Nichols, N.K.: Regularization of a Carbon-Cycle Model-Data Fusion Problem. Preprint University of, Reading, MPS-2013-10 (2013)
6. D'Amore, L., Arcucci, R., Marcellino, L., Murli, A.: A parallel three-dimensional variational data assimilation scheme. In: Numerical Analysis and Applied Mathematics, AIP C.P. vol. 1389, pp. 1829–1831 (2011)
7. D'Amore, L., Arcucci, R., Marcellino, L., Murli, A.: HPC computation issues of the incremental 3D variational data assimilation scheme in OceanVar software. J. Numer. Anal. Ind. Appl. Math. **7**(3–4), 91–105 (2012)
8. D'Amore, L., Arcucci, R., Carracciuolo, L., Murli, A.: OceanVAR software for use with NEMO: documentation and test guide. In: CMCC Research Papers Issue RP0173 (2012)
9. D'Elia, M., Perego, M., Veneziani, A.: A variational data assimilation procedure for the incompressible Navier–Stokes equations in hemodynamics. J. Sci. Comput. 1–20 (2011)
10. Dennis, J.E. Jr., Schnabel, R.B.: Numerical Methods for Unconstrained Optimization and Nonlinear Equations. Prentice Hall Series in Computational Mathematics. Prentice Hall Inc., Englewood Cliffs, NJ (1983)
11. Dobricic, S., Pinardi, N.: An oceanographic threedimensional variational data assimilation scheme. Ocean Modell. **22**, 89–105 (2008)
12. Freitag, M., Budd, C.J., Nichols, N.K.: Tikhonov regularization for large inverse problems. In: 17th ILAS Conference. Braunschweig, Germany (2011)
13. Fox, G.C., Williams, R.D., Messina, P.C.: Parallel. Computing Works!. Morgan Kaufmann Publishers Inc., Los Altos, CA (1994)
14. Foster, I.: Designing and Building Parallel Programs. Addison-Wesley, Reading, MA (1995)
15. Ghil, M., Malanotte-Rizzoli, P.: Data Assimilation in Meteorology and Oceanography, Advances in Geophysics, vol. 33. Academic Press, New York (1991)
16. Golub, G.H., Heath, M., Wahba, G.: Generalized cross-validation as a method for choosing a good ridge parameter. Technometrics **21**(2), 215–223 (1979)
17. Haben, S.A., Lawless, A.S., Nichols, N.K.: Conditioning of the 3DVAR Data Assimilation Problem, Mathematics Report 3/2009. Department of Mathematics, University of Reading (2009)
18. Haben, S.A., Lawless, A.S., Nichols, N.K.: Conditioning of Incremental Variational Data Assimilation, with Application to the Met Office System. Preprint University of Reading, MPS-2010-22 (2010)
19. Haben, S.A., Lawless, A.S., Nichols, N.K.: Conditioning and preconditioning of the variational data assimilation. Comput. Fluids **46**, 252–256 (2011)
20. Haben, S.A., Lawless, A.S., Nichols, N.K.: Conditioning of incremental variational data assimilation, with application to the Met Office system. Tellus A **63**, 782–792 (2011)
21. Kalman, R.E.: A new approach to linear filtering and prediction problems. Trans. ASME J. Basic Eng. **82**(Series D), 35–45 (1960)
22. Kalnay, E.: Atmospheric Modeling, Data Assimilation and Predictability. Cambridge University Press, Cambridge, MA (2003)
23. Keyes, D.E.: How scalable is domain decomposition in practice? In: Proceedings of the International Conference Domain Decomposition Methods, Greenwich, pp. 286–297 (1998)
24. Koivunen, A.C., Kostinski, A.B.: The feasibility of data whitening to improve performance of weather radar. J. Appl. Meteorol. **38**, 741–749 (1999)
25. Kostinski, A.B., Koivunen, A.C.: On the condition number of Gaussian sample-covariance matrices. IEEE Trans. Geosci. Remote Sens. **38**, 329–332 (2000)
26. Johnson, C., Hoskins, B.J., Nichols, N.K.: A singular vector perspective of 4-DVar: filtering and interpolation. Q. J. R. Meteorol. Soc. **131**, 120 (2005a)
27. Johnson, C., Nichols, N.K., Hoskins, B.J.: Very large inverse problems in atmosphere and ocean modelling. Int. J. Numeric. Methods Fluids **47**, 759771 (2005b)
28. Lawless, A.S., Gratton, S., Nichols, N.K.: Approximate iterative methods for variational data assimilation. Int. J. Numer. Meth. Fluids **47**, 1129–1135 (2005)
29. Le Dimet, F.X., Talagrand, O.: Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects. Tellus **38A**, 97–110 (1986)
30. Liu, D.C., Nocedal, J.: On the limited memory BFGS method for large scale optimization. Math. Programm. **45**, 503–528 (1989)
31. Marx, B. A., Potthast, R.W.E.: On Instabilities in Data Assimilation Algorithms. University of Reading, Department of Mathematics and Statistics, Preprint MPS-2012-06 (2012)
32. Miyoshi, T.: Computational Challenges in Big Data Assimilation with Extreme-Scale Simulations, Talk at BDEC Workshop. Charleston, SC (2013)

33. Mogensen, K., Alonso Balmaseda, M., Weaver, A.: The NEMOVAR ocean data assimilation system as implemented in the ECMWF ocean analysis for System 4. Research Department, CERFACS, Toulouse (2012)
34. Moler, C.: Experiments with MATLAB (2011)
35. Morozov, V.A.: Methods for Solving Incorrectly Posed Problems. Wien. Springer, New York (1984)
36. Murli, A., D'Amore, L., Carracciuolo, L., Ceccarelli, M., Antonelli, L.: High performance edge-preserving regularization in 3D SPECT imaging. Parallel Comput. **34**(2), 115–132 (2008)
37. NEMO, http://www.nemo.ocean.eu
38. Navon, I.M.: Data assimilation for numerical weather prediction: a review. In: Park, S.K., Xu, L. (eds.) Data Assimilation for Atmospheric, Oceanic and Hydrologic Applications. Springer, Berlin (2009)
39. Navon, I.M., Legler, D.M.: Conjugate gradient methods for large-scale minimization in metereology. Mon. Weather Rev. **115**, 1479–1502 (1987)
40. Nerger, L., Hiller, W. and Schrter, J. The Parallel Data Assimilation Framework PDAF—a flexible software framework for ensemble data assimilation, EGU General Assembly, April 23–27, 2012, Vienna, Austria (Geophysical Research Abstracts, vol. 14, EGU2012-1885)
41. Nocedal, J., Byrd, R.H., Lu, P., Zhu, C.: L-BFGS-B: fortran subroutines for large-scale bound-constrained optimization. ACM Trans. Math. Softw. **23**(4), 550–560 (1997)
42. Potthast, R., Graben, P.: Inverse problems in neural field theory. SIAM J. Appl. Dyn. Syst. **8**(4), 1405–1433 (2009)
43. Stewart, L.M., Dance, S., Nichols, N.K.: Data assimilation with correlated observation errors: experiments with a 1-D shallow water model. Tellus A. **65**, 19546 (2013)
44. Tikhonov, A.N.: Regularization of incorrectly posed problems. Sov. Math. Dokl. **4**, 1624–1627 (1963)
45. Vanoye, A., Mendoza, A.: Application of direct regularization techniques and bounded-variable least squares for inverse modeling of an urban emissions inventory. Athmosferic Pollut. Res. **5** (2014)
46. Zou, X., Navon, I.M., Berger, M., Phua, K.H., Schlick, T. LeDimet, F.X.: Numerical experience with limited-memory quasi-Newton and truncated Newton methods. SIAM J. Optim. **3**, 582–608 (1993)