Block Krylov methods for accelerating ensembles of variational data assimilations

François Mercier, Selime Gurol, Pierre Jolivet, Yann Michel, Thibaut Montmerle

Block Krylov methods for accelerating ensembles of variational data assimilations

François Mercier[1] | Selime Gürol[2] | Pierre Jolivet[3] | Yann Michel[1] | Thibaut Montmerle[1]

[1]CNRM, Météo-France/CNRS, Toulouse, France
[2]Cerfacs, Toulouse, France
[3]IRIT, CNRS, Toulouse, France

Correspondence
François Mercier, François Mercier, CNRM, Météo-France/CNRS, 42 Avenue Gustave Coriolis, 31057 Toulouse Cedex, France.
Email: francois.mercier@meteo.fr

We consider the problem of efficiently solving ensembles of variational data assimilations in the context of numerical weather prediction. Running several assimilations notably allows us to initialize ensemble prediction systems and to more accurately represent background-error statistics, but is computationally expensive, limiting ensemble size. We propose a new class of algorithms for speeding up the minimization of the ensemble of data assimilations. It consists of using block Krylov methods to simultaneously perform the minimization for all members of the ensemble, instead of performing each minimization separately.

We develop preconditioned block Krylov versions of the Full Orthogonal Method and of the Lanczos algorithm in both primal and dual space. The latter works in observation space that is usually of smaller dimension than the state space, thus giving an advantage in terms of memory requirements and computational cost. We describe and compare several parallelization strategies for speeding up the minimization and limiting the communications.

These methods have been tested on a quasi-geostrophic system, consisting of a simplified atmospheric circulation model equipped with an ensemble of 3D-Var schemes tuned to mimic some features of a limited-area numerical weather prediction system. Experimentation shows that the number of iterations needed to converge is drastically reduced by the block Krylov approaches. We indicate that, while working in primal space does not save significant computational time, working in the dual space may reduce the computational time by a factor of 2 to 5 (depending on ensemble size) compared to standard Krylov methods, making our approach attractive for operational use.

KEYWORDS

block Krylov techniques, dual approach, ensemble assimilation, Lanczos algorithm, parallelism, quasi-geostrophic model, variational assimilation

1 | INTRODUCTION

In numerical weather prediction (NWP), the initialization of the state requires determination of the best initial state of the atmosphere from all available information, in particular from a background state (a previous short-range forecast) and a set of observations. Most meteorological centres rely on variational data assimilation schemes that are efficient in the context of very large state space (10^6 to 10^9) and numerous observations (10^3 to 10^7). This is because variational schemes (just like ensemble methods) do not explicitly store the full covariance matrices (Dee, 1995). A minimization algorithm finds the solution iteratively; it only requires matrix–vector products. Variational data assimilation has allowed for the direct assimilation of radiances from numerous satellites and this has been a source of huge improvement in the quality of operational forecasts (Rabier, 2005).

Originally, variational assimilation schemes used climatological estimates of the background-error covariances (Bannister, 2008). However, those estimates lacked the flow-dependency and the cycling of the statistics as in the Kalman filter. More recently, the covariances have been

estimated from a current ensemble of forecasts, or by making a weighted (hybrid) combination of such ensemble-based covariances together with (modelled) climatological ones (Hamill and Snyder, 2000). This is fully compatible with the variational framework (Lorenc, 2003; Buehner, 2005). In some implementations, this ensemble comes from a separate ensemble data assimilation scheme that is not variational, but rather based on some form of the Ensemble Kalman Filter (EnKF; Evensen, 1994; Houtekamer and Mitchell, 2001) or on the Local Ensemble Transform Kalman Filter (Hunt *et al.*, 2007).

In the stochastic EnKF, every ensemble member includes observation perturbations which are drawn from the observation-error covariance matrix, and model perturbations which are constructed to be representative of model errors. This scheme can be mimicked by running an ensemble of variational data assimilations (EDAs) which includes similar observation and model perturbations (Fisher, 2003; Žagar *et al.*, 2005; Kucukkaraca and Fisher, 2006). This EDA can be used to initialize ensemble prediction systems as the EnKF, and is believed to more accurately represent the background-error statistics (Berre *et al.*, 2006; Bowler *et al.*, 2017). For limited-area modelling (LAM) at Météo-France, a regional EDA based on the AROME model (Seity *et al.*, 2011) is being developed.

Performing EDA with many members in the ensemble is costly, as many variational assimilation systems have to be run. So EDAs are generally performed at a coarser resolution than the deterministic system and with a limited ensemble size. This implies spurious sampling noise in the estimation of the covariances, which can be mitigated with spatial filtering techniques (Buehner and Charron, 2007; Berre and Desroziers, 2010; Ménétrier *et al.*, 2015). However, larger ensemble sizes or more computationally efficient EDAs are clearly desirable. This is particularly true for the operational AROME-France which is run at high resolution (currently 1.3 km in its deterministic version and 3.8 km in its EDA version) over western Europe, implying large state space (of dimension 10^8 for the EDA). Reducing the cost of the minimization of the EDAs would certainly be very beneficial for operations.

Different methods have been proposed in order to speed up the minimizations in the EDA. Desroziers and Berre (2012) use the similarity of the minimization problems that have to be solved. They suggest building a better starting point for a new perturbed minimization using Lanczos vectors estimated through one or several previous perturbed minimizations. Lorenc *et al.* (2017) propose the Mean-Pert method: the minimizations of the EDA are replaced by a minimization for the ensemble mean followed by minimizations for the perturbations from it that are accelerated by preconditioning and early stopping. Those two schemes do obtain gains in efficiency, but at the detriment of more sequential computations; a minimization with unperturbed data or for the ensemble mean has to be performed first, possibly degrading the total time to solution.

In this study we propose another complementary approach based on the use of block Krylov methods. Under an appropriate linearization hypothesis, the EDA results in an ensemble of linear systems sharing the same Hessian but with different (perturbed) right-hand sides. The standard implementation of EDA consists of applying a Krylov method such as the conjugate gradient algorithm (Hestenes and Stiefel, 1952) to independently solve these systems. However, block Krylov methods (O'Leary, 1980) are especially designed to solve this problem jointly, in one pass. They broaden the subspace in which an approximate solution is iteratively searched, and so can be able to speed up the convergence rate.

The purpose of this paper is to discuss, implement and test block Krylov approaches in the context of EDAs with large systems as encountered in NWP. More precisely, we will use in this paper a simplified but realistic numerical model, the quasi-geostrophic (QG) model of Fandry and Leslie (1984), associated with an ensemble of 3D-Var that mimics some properties of the AROME EDA run at Météo-France. The QG EDA system used here will be run at a resolution implying a control vector of dimension 4×10^5. The implementation will be made in the context of the Oriented Object Prediction System (OOPS; Fisher *et al.*, 2011; Bonavita *et al.*, 2017), a common C++ framework developed by ECMWF, Météo-France and their partners. This software allows an easy use of new assimilation schemes with different numerical models, including the QG model and AROME.

Gratton and Tshimanga (2009) have introduced range-space variants of Krylov methods that work in observation space. This allows us to speed up the minimization when the number of observations is small compared to the dimension of the problem. Indeed, the inner products are less expensive to compute, and the algorithm also uses less memory to store the vectors. We propose in this paper to adapt this dual formulation to block Krylov methods. A particular focus will be given to the development of efficient parallelization strategies in the framework of OOPS.

Section 2 briefly introduces the classical variational formulation and the algorithms commonly used to minimize the cost function, as well as the EDA problem. Section 3 presents the block Krylov methods used to solve the EDA problem, both in primal and dual spaces, and details the algorithms implemented in this work. Then section 4 focuses on the implementation of these algorithms in the OOPS framework and on the associated parallelization strategies. Section 5 presents the results obtained with the QG model, both in terms of convergence and in terms of time consumption. Finally section 6 recalls the main conclusions of this paper and introduces future ongoing developments.

2 | VARIATIONAL ASSIMILATION AND THE EDA: FORMULATION

2.1 | Deterministic variational assimilation

The following derivation directly applies to 3D state estimation by 3D-Var, 3DEnVar or hybrid schemes, which differ only with respect to the way the background-error covariance matrix is formed and which we do not detail. The 4D extension of those formulations, e.g. 4D-Var (Le Dimet and Talagrand, 1986; Courtier *et al.*, 1994) and 4DEnVar (Liu *et al.*, 2008; Buehner *et al.*, 2013; Desroziers *et al.*, 2014) are also of great interest. Following Lorenc *et al.* (2015), the same derivation can be applied with states that become four-dimensional. However, in the 4D-Var case the propagation by the nonlinear model has to be accounted for within the four-dimensional observation operator. We will return to this specific case in section 2.4. The derivations of sections 2.1, 2.2, and 2.3 are valid only for 3D-Var, 3DEnVar and 4DEnVar (including hybrid versions).

Following the notations of Ide *et al.* (1997), $\mathbf{x}$ denotes a model state, $\mathbf{x}^{\mathrm{b}}$ the background state, and $\mathbf{x}^{\mathrm{a}}$ the best estimate of the model state – the analysis. $\mathbf{y}^{\mathrm{o}}$ refers to the observation vector. In this paper, we denote by N the dimension of the problem, i.e. $\mathbf{x} \in \mathbb{R}^N$, and N_{obs} the number of observations, i.e. $\mathbf{y}^{\mathrm{o}} \in \mathbb{R}^{N_{\mathrm{obs}}}$. $\mathbf{B} \in \mathbb{R}^{N\times N}$ is the error covariance matrix of the background and $\mathbf{R} \in \mathbb{R}^{N_{\mathrm{obs}}\times N_{\mathrm{obs}}}$ is the error covariance matrix of the observations. $\mathcal{H}$ is the observation operator, allowing us to estimate an observation vector from a model state. Under these notations, the analysis $\mathbf{x}^{\mathrm{a}}$ is defined as the model state minimizing the cost function:

$$J(\mathbf{x}) = \frac{1}{2}\left(\mathbf{x}-\mathbf{x}^{\mathrm{b}}\right)^{\mathrm{T}}\mathbf{B}^{-1}\left(\mathbf{x}-\mathbf{x}^{\mathrm{b}}\right) + \frac{1}{2}\{\mathbf{y}^{\mathrm{o}} - \mathcal{H}(\mathbf{x})\}^{\mathrm{T}}\mathbf{R}^{-1}\{\mathbf{y}^{\mathrm{o}} - \mathcal{H}(\mathbf{x})\}. \quad (1)$$

This cost function is in general non-quadratic, due to the presence of the nonlinear observation operator $\mathcal{H}$. All schemes use an incremental approach, where the minimization is performed with respect to the deviations from the background state. Two different approaches can be taken to perform the (approximate) minimization of Equation (1). The first one is a quasi-Newton method (Nocedal and Wright, 2006), as in the operational version of AROME or in the Met Office 4D-Var (Lorenc *et al.*, 2000)which both use the M1QN3 minimizer (Gilbert and Lemaréchal, 2006). Another option, used for instance in the global 4D-Var of Météo-France and ECMWF, consists of applying an approximate Gauss–Newton method (Nocedal and Wright, 2006; Gratton *et al.*, 2007). This amounts to solving a sequence of quadratic problems (referred as the outer loop) where the operators are relinearized against the current estimate (the guess). Each quadratic problem is then solved with a Conjugate Gradient (CG) or Lanczos method (referred to as the inner loop).

For the sake of simplicity, and because we are only discussing the inner loop problem, we write the incremental linearized cost function as:

$$J(\delta\mathbf{x}) = \frac{1}{2}\delta\mathbf{x}^{\mathrm{T}}\mathbf{B}^{-1}\delta\mathbf{x} + \frac{1}{2}\left(\mathbf{d}-\mathbf{H}\delta\mathbf{x}\right)^{\mathrm{T}}\mathbf{R}^{-1}\left(\mathbf{d}-\mathbf{H}\delta\mathbf{x}\right), \quad (2)$$

in which $\delta\mathbf{x} \in \mathbb{R}^N$ is an increment related to $\mathbf{x}$ and $\mathbf{x}^{\mathrm{b}}$ by $\mathbf{x} = \mathbf{x}^{\mathrm{b}} + \delta\mathbf{x}$, and $\mathbf{H}$ is a linearized version of the observation operator. The innovation vector $\mathbf{d}$ measures the gap between the observations and the corresponding values predicted by the model:

$$\mathbf{d} = \mathbf{y}^{\mathrm{o}} - \mathcal{H}(\mathbf{x}^{\mathrm{b}}).$$

In the simpler case described by Equation 2, the analyzed increment $\delta\mathbf{x}^{\mathrm{a}}$ is determined by solving the linear system which allows the gradient to vanish:

$$\left(\mathbf{B}^{-1} + \mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}\right)\delta\mathbf{x} = \mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{d}. \quad (3)$$

Krylov subspace methods can be used for numerically solving this linear system. They include the Full Orthogonalization Method (FOM), the Lanczos method, or the Conjugate Gradient (CG) method. All these algorithms produce mathematically equivalent iterates in exact arithmetic if the linear system matrix in Equation 3 is symmetric and positive definite (cf. Chapter 6 of Saad, 2003).

In order to improve the convergence of Krylov subspace methods, a preconditioner is desirable. An efficient preconditioner accelerates the convergence sufficiently such that the gain in convergence speed overcomes the extra cost of constructing and applying the preconditioner (Chapter 10 of Saad, 2003). For variational data assimilation, a common choice to precondition Equation 3 is the background-error covariance matrix (Bannister, 2008). In order to apply $\mathbf{B}$ as a preconditioner, there are two equivalent variants of the algorithms. The first one applies a split preconditioner through the square root of $\mathbf{B}$ and its adjoint to a Krylov method using the canonical inner product. The second performs a right $\mathbf{B}$ preconditioning to a Krylov method using the $\mathbf{B}$ inner product[1] (Derber and Rosati, 1989; Gürol *et al.*, 2014) and does not require the factorization of $\mathbf{B}$.

Alternatively, an approximate solution of Equation 3 can be found by solving the dual problem (Courtier, 1997; Gratton *et al.*, 2013) which performs the minimization in the observation space. The dual formulation may be preferable when the number of observations is smaller than the dimension of the control variable $\delta\mathbf{x}$ since it reduces the computational cost and requires less memory. In the dual space, the Krylov subspace method is applied to the following linear system:

$$\left(\mathbf{R} + \mathbf{H}\mathbf{B}\mathbf{H}^{\mathrm{T}}\right)\lambda = \mathbf{d}, \quad (4)$$

with $\lambda \in \mathbb{R}^{N_{\mathrm{obs}}}$ a control variable in the observation space related to $\delta\mathbf{x}$ by

$$\delta\mathbf{x} = \mathbf{B}\mathbf{H}^{\mathrm{T}}\lambda.$$

[1] $\forall(\mathbf{u},\mathbf{v}) \in \mathbb{R}^N \times \mathbb{R}^N$, $(\mathbf{u},\mathbf{v})_{\mathbf{B}} = \mathbf{v}^{\mathrm{T}}\mathbf{B}\mathbf{u}$

The Physical Space Assimilation System (PSAS; Courtier, 1997; Cohn *et al.*, 1998) solves this linear system using split $\mathbf{R}^{-1}$ preconditioning and CG with canonical inner product. Unfortunately, PSAS has a non-monotonic behaviour in the reduction of the primal quadratic cost function Equation 2 (El Akkraoui *et al.*, 2008). Therefore, a better alternative is to use the Restricted Preconditioned CG (RPCG) algorithm from Gratton and Tshimanga (2009). Following Gürol *et al.* (2014), RPCG is solving the left $\mathbf{R}^{-1}$ preconditioned dual system:

$$\left(\mathbf{Id}_{\mathbf{N}_{\text{obs}}} + \mathbf{R}^{-1}\mathbf{H}\mathbf{B}\mathbf{H}^{\mathrm{T}}\right)\lambda = \mathbf{R}^{-1}\mathbf{d}, \tag{5}$$

using CG equipped with the (semi-definite) $\mathbf{HBH}^{\mathrm{T}}$ inner product, and where $\mathbf{Id}_{\mathbf{N}_{\text{obs}}}$ is the identity matrix of $\mathbb{R}^{N_{\text{obs}}\times N_{\text{obs}}}$. RPCG generates the same iterates as those generated when solving the primal linear system Equation 3. Equivalent restricted Lanczos or FOM algorithms can be used as well (Gratton *et al.*, 2011; 2012; Gürol *et al.*, 2014).

2.2 | Ensemble of variational data assimilations (EDA)

Unlike the EnKF or LETKF, variational schemes do not provide an estimate of the background- and analysis-error covariances (Fisher and Courtier, 1995). A consistent way of estimating the errors in the system is to perform an EDA with perturbed statistics, following a Monte Carlo approach. We will denote the ensemble size by m and use the subscript k to design the kth member. In the EDA, observation errors are simulated by adding random perturbations $\delta\mathbf{y}_k^{\mathrm{o}}$ drawn from $\mathbf{R}$, and some model error effects are simulated by inflation of forecast perturbations (Raynaud *et al.*, 2012) or by perturbing physical tendencies of the model. These perturbations are propagated through the data assimilation cycling, which leads to analysis and background perturbations (denoted by $\delta\mathbf{x}_k^{\mathrm{b}}$) whose covariances are consistent with those of associated errors (Belo-Pereira and Berre, 2006). The EDA, in most implementations, is thus the (parallel) minimization of m perturbed cost functions based on the deterministic cost function Equation 1 (Desroziers and Berre, 2012; Lorenc *et al.*, 2017):

$$\begin{aligned} J_k(\mathbf{x}_k) = &\frac{1}{2}\left(\mathbf{x}_k - \mathbf{x}^{\mathrm{b}} - \delta\mathbf{x}_k^{\mathrm{b}}\right)^{\mathrm{T}}\mathbf{B}_k^{-1}\left(\mathbf{x}_k - \mathbf{x}^{\mathrm{b}} - \delta\mathbf{x}_k^{\mathrm{b}}\right) \\ &+ \frac{1}{2}\left\{\mathbf{y}_k^{\mathrm{o}} + \delta\mathbf{y}_k^{\mathrm{o}} - \mathcal{H}_k\left(\mathbf{x}_k\right)\right\}^{\mathrm{T}} \\ &\times \mathbf{R}_k^{-1}\left\{\mathbf{y}_k^{\mathrm{o}} + \delta\mathbf{y}_k^{\mathrm{o}} - \mathcal{H}_k\left(\mathbf{x}_k\right)\right\}. \end{aligned} \tag{6}$$

Note that in Equation 6, we have so far retained nonlinear effects of variational data assimilation through the dependence of $\mathbf{B}, \mathbf{R}$ and $\mathcal{H}$ on the ensemble member k. The quality control applied to the perturbed observations can potentially lead to their rejection for some members. Then the observation vector (and thus the observation operator and the observation-error covariance matrix) can be different between the members. Variational quality control also changes the weights in $\mathbf{R}$, possibly in a different way between the members. Finally, the dependence of $\mathbf{B}$ with k may arise from the use of nonlinear balance equations or nonlinear humidity change of variable.

2.3 | Linearized EDA

The EDA as formulated by Equation 6 includes nonlinear effects. As discussed by Lorenc *et al.* (2017), the EDA describes the background-error statistics under a linear approximation that is common to the EnKF. There are several ways to formulate a linearized EDA that differ on the linearization state (or trajectory). Lorenc *et al.* (2017) propose to first solve for the ensemble mean analysis, retaining all nonlinear effects, then solve for the ensemble perturbations in a linear context. We will see that the block Krylov approach is directly applicable in such a linearized EDA context. Another approach would be to linearize against the ensemble mean for the background, which is simpler and more efficient but possibly detrimental as the EDA is not recentred. We first plan to implement this method with the AROME EDA, which is linearized except for the quality control (through the first-guess check) and the nonlinearity of some observation operators (in particular due to the dependence on the vertical coordinate).

We suppose that in the linearized EDA we have a common observation vector $\mathbf{y}^{\mathrm{o}}$ and statistics. To this end we suppose that there is no quality control applied to the observations after their perturbation (although one can remain before the perturbation). The observation operator $\mathbf{H}$ is linearized against a common state, the (background or analysis) ensemble mean. This linearization around a common state, as well as the absence of quality control on perturbed observations, are obviously simplifications as $\mathbf{H}$ is supposed to depend on the member index k. However in the 3D-Var application considered in this paper (QG model, section 5) there is no quality control in any case and $\mathcal{H}$ is independant of k, so that we will not further explore the implications of these assumptions. Nevertheless for real 3D-Var systems, we would verify the consistency of our solution by re-evaluating the nonlinear cost function value (using $\mathcal{H}_k$) at the end of the minimization process. Then, for each member k, the cost functions have the same expression as in Equation 2. The corresponding linearized subproblems in incremental assimilation are then

$$J_k(\delta\mathbf{x}_k) = \frac{1}{2}\delta\mathbf{x}_k^{\mathrm{T}}\mathbf{B}^{-1}\delta\mathbf{x}_k + \frac{1}{2}(\mathbf{d}_{\mathrm{k}} - \mathbf{H}\delta\mathbf{x}_k)^{\mathrm{T}}\mathbf{R}^{-1}(\mathbf{d}_k - \mathbf{H}\delta\mathbf{x}_k), \tag{7}$$

where $\delta\mathbf{x}_k = \mathbf{x}_k - \mathbf{x}^{\mathrm{b}} - \delta\mathbf{x}_k^{\mathrm{b}}$ are the deviations from the perturbed backgrounds for each member and $\mathbf{d}_k = \mathbf{y}^{\mathrm{o}} + \delta\mathbf{y}_k^{\mathrm{o}} - \mathcal{H}_k(\mathbf{x}^{\mathrm{b}} + \delta\mathbf{x}_k^{\mathrm{b}})$ are the innovations with perturbed observations, using $\mathcal{H}_k(\mathbf{x}^{\mathrm{b}} + \delta\mathbf{x}_k^{\mathrm{b}} + \delta\mathbf{x}_k) \approx \mathcal{H}_k(\mathbf{x}^{\mathrm{b}} + \delta\mathbf{x}_k^{\mathrm{b}}) + \mathbf{H}\delta\mathbf{x}_k)$. So, in the primal space, the minimization of these m linearized subproblems results in m linear systems with the same Hessian $\mathbf{B}^{-1} + \mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}$ (cf. Equation 3):

$$\left(\mathbf{B}^{-1} + \mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}\right)\delta\mathbf{x}_k = \mathbf{b}_k, \tag{8}$$

with right-hand sides $\mathbf{b}_k = \mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{d}_k$ for $k \in [\![1; m]\!]$.

Similarly, in the dual space, we have m linear systems with the same Hessian $\mathbf{R}+\mathbf{HBH}^{\mathrm{T}}$ (cf. Equation 4) for $k \in [\![1;m]\!]$:

$$\left(\mathbf{R}+\mathbf{HBH}^{\mathrm{T}}\right)\lambda_k = \mathbf{d}_k. \tag{9}$$

The standard EDA implementation, in primal or dual space, consists of running m independent minimizations in parallel. This does not take full advantage of the fact that, in the linearized context defined above, all the linear systems have the same Hessian (with perturbed right-hand sides). The purpose of this paper is to introduce block Krylov methods which exploit the mentioned structure of the problem to speed up the numerical minimization process.

2.4 | Comments for 4D-Var application

The propagation of the state by a nonlinear model in the 4D-Var formulation makes the previous derivation more complex. First, we note that in this case, the nonlinear operator $\mathcal{H}$ of Equation 1 incorporates the propagation by the nonlinear model.

Then, when using the Gauss–Newton approach to solve the incremental approach, the problem is relinearized around the guess through the outer loop process, which is not taken into account in Equation 2. There are additional nonlinear refinements to some 4D-Var schemes which we do not discuss here. They include the quality control of observations with non-Gaussian errors (Andersson and Jarvinen, 1999; Tavolato and Isaksen, 2015), the variational bias correction (Derber and Wu, 1998; Auligné *et al.*, 2007), the dependence of background-error statistics on the first guess through nonlinear balance equations (Fisher, 2003; Barker *et al.*, 2004) or the nonlinear humidity change of variable (Andersson *et al.*, 2005; Ingleby *et al.*, 2013).

Moreover, for performing an EDA written in its linearized incremental form as in Equation 7, we would have to linearize the nonlinear operators $\mathcal{H}_k$ of Equation 6 (which include the nonlinear model trajectories) against a reference trajectory given by the ensemble mean background or analysis, and to perform a single outer loop. However, the influence of nonlinear effects in the EDA is outside the scope of this paper and deserves further research.

3 | BLOCK KRYLOV METHODS FOR LINEARIZED EDA

3.1 | Block Krylov methods in primal space

In this section, we introduce block Krylov methods for solving simultaneously the linearized EDA in primal space as described in section 2.2. General introductions to block Krylov techniques can be found in O'Leary (1980) and Gutknecht (2006).

3.1.1 | Notation / right-B preconditioning

The numerical experiments carried out in this paper apply a block Krylov subspace method equipped with $\mathbf{B}$ inner product to the right-$\mathbf{B}$ preconditioned linear system. For an arbitrary member k in the ensemble, it consists of rewriting Equation 8 in the form

$$\left(\mathbf{Id_N}+\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{HB}\right)\mathbf{u}_k = \mathbf{b}_k,$$

with $\mathbf{Id_N}$ the identity matrix of $\mathbb{R}^{N\times N}$, still $\mathbf{b}_k = \mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{d}_k$, and with the new transformed control variable $\mathbf{u}_k \in \mathbb{R}^N$ linked to $\delta\mathbf{x}$ through

$$\delta\mathbf{x}_k = \mathbf{B}\mathbf{u}_k.$$

Then, we introduce the notation $\mathbf{A} = \mathbf{Id_N}+\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{HB}$ for the Hessian of this problem, so that it can be written

$$\forall k \in [\![1;m]\!], \quad \mathbf{A}\mathbf{u}_k = \mathbf{b}_k. \tag{10}$$

The block Krylov methods we describe here are iterative processes. So in this section, subscript p stands for the final iteration, while subscript i denotes the quantities corresponding to an arbitrary iteration. Moreover, as previously, subscript m is the total number of members while subscript k denotes an arbitrary member.

At final iteration p, an iterative process produces an approximate solution for $\mathbf{u}_k$ which will be written $\mathbf{u}_{k,p}$. We note that we suppose here that the starting point of the minimization is $\mathbf{u}_{k,0} = 0$. Then, the initial residuals for member k, denoted $\mathbf{r}_{k,0}$, is $\mathbf{r}_{k,0} = \mathbf{b}_k - \mathbf{A}\mathbf{u}_{k,0} = \mathbf{b}_k$.

Under these assumptions, Equation 10 can be written in a block form:

$$\mathbf{A}\mathbf{u} = \mathbf{b}, \tag{11}$$

with the block control variable $\mathbf{u} \in \mathbb{R}^{N\times m}$:

$$\mathbf{u} = \left[\mathbf{u_1} \ \cdots \ \mathbf{u}_m\right],$$

and the block right-hand side $\mathbf{b} \in \mathbb{R}^{N\times m}$:

$$\mathbf{b} = \left[\mathbf{b_1} \ \cdots \ \mathbf{b}_m\right].$$

The block initial residuals are $\mathbf{r}_0 = \left[\mathbf{r}_{1,0} \ \cdots \ \mathbf{r}_{m,0}\right] = \mathbf{b}$.

At the end of an iterative process with p iterations, the approximate solution is $\mathbf{u}_p \in \mathbb{R}^{N\times m}$, while kth member approximate solution is still $\mathbf{u}_{k,p}$.

3.1.2 | Presentation

A block Krylov method is an iterative projection method which consists, at iteration p, in looking for an approximate $\mathbf{u}_p$ of $\mathbf{u}$ with

$$\mathbf{u}_p \in \mathcal{B}_p\left(\mathbf{A}, \mathbf{r_0}\right),$$

in which $\mathcal{B}_p\left(\mathbf{A}, \mathbf{r_0}\right)$ is the block Krylov subspace defined by

$$\mathcal{B}_p\left(\mathbf{A}, \mathbf{r_0}\right) = \text{Blockspan}\left(\mathbf{r_0}, \mathbf{A}\mathbf{r_0}, \ldots, \mathbf{A}^{p-1}\mathbf{r_0}\right).$$

Here, "Blockspan" (Gutknecht, 2006, p. 9) is defined by

$$\mathcal{B}_p(\mathbf{A}, \mathbf{r}_0) = \left\{\mathbf{r}_0\boldsymbol{\gamma}_1 + \mathbf{A}\mathbf{r}_0\boldsymbol{\gamma}_2 + \cdots + \mathbf{A}^{p-1}\mathbf{r}_0\boldsymbol{\gamma}_p, \right.$$
$$\left. \text{with } \boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_p \in \mathbb{R}^{m\times m}\right\}. \tag{12}$$

We can write this as well for any single member $\mathbf{u}_{k,p}$:

$$\mathbf{u}_{k,p} \in \text{Span}\left\{\mathbf{A}^i\mathbf{r}_{j,0} \;\middle|\; (i,j) \in [\![0;p-1]\!] \times [\![1;m]\!]\right\}. \tag{13}$$

This highlights the fact that all members use the same research space, which is spanned from the entire ensemble.

$$\mathbf{u}_p \in \mathcal{B}_p(\mathbf{A}, \mathbf{r}_0) \text{ means } \mathbf{u}_p = \mathbf{V}_p \mathbf{s}, \tag{14}$$

where $\mathbf{V}_p$ is an orthonormal basis of $\mathcal{B}_p(\mathbf{A}, \mathbf{r}_0)$, i.e. $m \cdot p$ vectors and $\mathbf{V}_p \in \mathbb{R}^{N \times m \cdot p}$ if all the vectors implied in Equation 13 are independent, and $\mathbf{s} \in \mathbb{R}^{m \cdot p \times m}$ the coefficients of a linear combination. We call the process allowing us to estimate $\mathbf{V}_p$ the *Arnoldi process*, and $\mathbf{V}_p$ is called the Arnoldi basis.

To define uniquely the approximate $\mathbf{u}_p$, we impose a Galerkin condition. Namely, we impose that $\mathbf{r}_p$, the block of residuals at iteration p, is orthogonal to the pth block Krylov subspace. This can be used to define the block FOM. If $\mathbf{A}$ is symmetric, it leads to the block Lanczos method which is mathematically equivalent to the block CG and the block FOM. The condition of orthogonality of the residuals $\mathbf{r}_p \perp \mathcal{B}_p(\mathbf{A}, \mathbf{r}_0)$ can be written

$$\mathbf{V}_p^{\mathrm{T}} \mathbf{r}_p = 0.$$

Using the definition of the block of residuals $\mathbf{r}_p = \mathbf{b} - \mathbf{A}\mathbf{u}_p$ and the fact that $\mathbf{b} = \mathbf{r}_0$,

$$\mathbf{V}_p^{\mathrm{T}} \mathbf{r}_0 = \mathbf{V}_p^{\mathrm{T}} \mathbf{A} \mathbf{u}_p. \tag{15}$$

Combining Equations 14 and 15 leads to an expression for $\mathbf{u}_p$:

$$\mathbf{u}_p = \mathbf{V}_p \underline{\mathbf{T}}^{-1} \mathbf{V}_p^{\mathrm{T}} \mathbf{r}_0, \tag{16}$$

with $\underline{\mathbf{T}} = \mathbf{V}_p^{\mathrm{T}} \mathbf{A} \mathbf{V}_p \in \mathbb{R}^{m \cdot p \times m \cdot p}$. $\underline{\mathbf{T}}$ is constructed in the iterative process during the orthonormalization of the basis of the block Krylov subspace (section 3.1.3). Generally, in the NWP context, we have small p and m compared to the dimensions of the problem ($m \cdot p < 2 \times 10^3$ versus $N = 10^8$ and $N_{\mathrm{obs}} = 10^5$), so that the inversion of $\underline{\mathbf{T}}$ is not challenging.

We can note that the search subspace at iteration 1 for any member is the space spanned by $\mathbf{r}_0$, so that we can write

$$\mathbf{r}_0 = \mathbf{v}_1 \boldsymbol{\beta}_0,$$

where $\mathbf{v}_1 \in \mathbb{R}^{N \times m}$ are the m first base vectors and $\boldsymbol{\beta}_0 \in \mathbb{R}^{m \times m}$ is a projection matrix resulting from a QR decomposition of $\mathbf{r}_0$.

Then, Equation 16 can be written

$$\mathbf{u}_p = \mathbf{V}_p \underline{\mathbf{T}}^{-1} \begin{bmatrix} \mathbf{v}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{v}_p^{\mathrm{T}} \end{bmatrix} \mathbf{v}_1 \boldsymbol{\beta}_0.$$

Using the orthogonality of the base vectors,

$$\mathbf{u}_p = \mathbf{V}_p \underline{\mathbf{T}}^{-1} \underline{\mathbf{e}_1} \boldsymbol{\beta}_0, \tag{17}$$

with $\underline{\mathbf{e}_1} = \begin{bmatrix} \mathbf{Id}_m & \mathbf{0}_m & \cdots & \mathbf{0}_m \end{bmatrix}$, where $\mathbf{Id}_m$ is the identity matrix of $\mathbb{R}^{m \times m}$ and $\mathbf{0}_m$ the null matrix of the same dimension.

We remark that the search subspace of the block Krylov methods is the same for all the members. It combines information coming from the Hessian and from all the right-hand sides of the problem. Then its dimension is lower than or equal to $p \cdot m$, with equality if all the vectors implied in Equation 13 are independent. At each iteration, the dimension of the block Krylov subspace increases by (at most) m, the number of members.

If at some point of the iterative process we generate vectors which appear to be linked, the dimension of the search subspaces decreases; we call this an inexact breakdown. Such a breakdown may appear for example when an exact solution is found for one of the right-hand sides in the block (also known as happy breakdown in the context of non-block Krylov methods), or when there is a loss of orthogonality between residuals. Dealing with this phenomenon involves making an explicit reduction of the number of right-hand sides, for instance using deflation (Gutknecht, 2006). Deflation may be used not only in the case of exact linear dependence of the generated vectors; techniques have been developed to force deflation by defining a tolerance threshold below which vectors are considered as linked, so that we can reduce the size of the Arnoldi basis.

However, in this paper, we deal with vectors of large dimensions ($> 10^5$) which are randomly perturbed (both backgrounds and observations); right-hand sides are unlikely to be dependent. Also, we perform a limited number of iterations (typically less than 50), such that the minimization is stopped quite early, i.e. before full convergence to machine precision. This may limit the occurrence of finding dependencies between the vectors through the algorithm. In all experiments presented in this paper, we have not encountered such cases of inexact breakdowns. Therefore, deflation has not been found to be necessary and we will not look into this issue in this paper. This point may be considered again with a more realistic system, although preliminary experiments confirm that we do not need deflation.

3.1.3 | Block B-FOM

In our numerical experiments we use a block Krylov subspace method with the right-$\mathbf{B}$ preconditioner based on the Arnoldi process, the block B-FOM.

First, we notice that the right-$\mathbf{B}$ preconditioned Hessian, i.e. $\mathbf{A} = \mathbf{Id}_N + \mathbf{H}^{\mathrm{T}} \mathbf{R}^{-1} \mathbf{H} \mathbf{B}$, is not symmetric with respect to the canonical inner product. Although block B-FOM can be used for unsymmetric systems, it is known that the iterative solution of a symmetric system is more stable. For this reason, we use the $\mathbf{B}$-inner product with respect to which the preconditioned matrix is symmetric. The corresponding block B-FOM is given in Algorithm 1.

In Algorithm 1, $\mathbf{V}_p$ and $\mathbf{Z}_p$, both in $\mathbb{R}^{N \times m \cdot p}$ at iteration p, contain the orthonormal basis of the block Krylov subspace respectively in the physical space and in the $\mathbf{B}$-preconditioned space, i.e. $\mathbf{Z}_p = \mathbf{B}\mathbf{V}_p$. During an iteration (lines 5–17), we firstly define the new m search directions ($\mathbf{w_i} \in \mathbb{R}^{N \times m}$, line 6). Then, lines 7–11 orthogonalize these $\mathbf{w}_i$ against all the base vectors computed at previous iterations. The projection matrices $\boldsymbol{\alpha}_j$ are stored as block components of matrix $\underline{\mathbf{T}}$ (line 10). In line 14, matrix $\boldsymbol{\beta}_i$ represents the lower-diagonal block component of matrix $\underline{\mathbf{T}}$. Lines 18–19 allow us to calculate

Algorithm 1 Block B-FOM

1: $\mathbf{z}_0 = \mathbf{B}\mathbf{r}_0$
2: $\left[\mathbf{v}_1, \mathbf{z}_1, \boldsymbol{\beta}_0\right] = qr\left(\mathbf{r}_0, \mathbf{z}_0\right)$
3: $\mathbf{V}_1 = [\mathbf{v}_1]$
4: $\mathbf{Z}_1 = [\mathbf{z}_1]$
5: **for** $i = 1 : p-1$ **do**
6: $\quad \mathbf{w}_i = \mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}\mathbf{z}_i + \mathbf{v}_i$
7: $\quad$ **for** $j = 1 : i$ **do**
8: $\quad\quad \boldsymbol{\alpha}_j = \mathbf{z}_j^{\mathrm{T}}\mathbf{w}_i$
9: $\quad\quad \mathbf{w}_i := \mathbf{w}_i - \mathbf{v}_j\boldsymbol{\alpha}_j$
10: $\quad\quad \underline{\mathbf{T}}_{jm:(j+1)m, im:(i+1)m} = \boldsymbol{\alpha}_j$
11: $\quad$ **end for**
12: $\quad \mathbf{z}_i = \mathbf{B}\mathbf{w}_i$
13: $\quad \left[\mathbf{v}_{i+1}, \mathbf{z}_{i+1}, \boldsymbol{\beta}_i\right] = qr\left(\mathbf{w}_i, \mathbf{z}_i\right)$
14: $\quad \underline{\mathbf{T}}_{(i+1)m:(i+2)m, im:(i+1)m} = \boldsymbol{\beta}_i$
15: $\quad \mathbf{V}_{i+1} := [\mathbf{V}_i; \mathbf{v}_{i+1}]$
16: $\quad \mathbf{Z}_{i+1} := [\mathbf{Z}_i; \mathbf{z}_{i+1}]$
17: **end for**
18: solve $\underline{\mathbf{T}}_{1:p\cdot m, 1:p\cdot m}\mathbf{s}_p = \mathbf{e}_1\boldsymbol{\beta}_0$
19: $\boldsymbol{\delta}\mathbf{x}_p = \mathbf{Z}_p\mathbf{s}_p$

the approximate solution at iteration p, corresponding to Equation 17, using matrix $\underline{\mathbf{T}}_{1:m\cdot p, 1:m\cdot p} \in \mathbb{R}^{m\cdot p \times m\cdot p}$. (We notice that $\boldsymbol{\delta}\mathbf{x}_p = \mathbf{B}\mathbf{u}_p = \mathbf{B}\mathbf{V}_p\mathbf{s}_p = \mathbf{Z}_p\mathbf{s}_p$.)

We note that the quantities in the **B**-preconditioned space ($\mathbf{z}_i$ and $\mathbf{Z}_i$) are used as intermediates of calculations to facilitate the application of the **B** inner product and of the Hessian, so that any operator ($\mathbf{B}$, $\mathbf{H}$, $\mathbf{H}^{\mathrm{T}}$, $\mathbf{R}^{-1}$) is applied only once per iteration.

We note as well that there is no criterion allowing us to stop the minimization in the case of satisfying convergence in Algorithm 1. (We suppose that we perform a predefined number of iterations.) Obviously we could add a criterion calculated at the end of each iteration to monitor the convergence, for instance on the residuals or on the background part of the cost function. However because many criteria exist, we leave this discussion for section 5.3.1 on our results.

The QR decomposition (lines 2 and 13) has to be made with respect to the **B** inner product as well (oblique QR decomposition). It consists of writing $\mathbf{w}_i$ in the form $\mathbf{w}_i = \mathbf{v}_{i+1}\boldsymbol{\beta}_i$, with $\mathbf{v}_{i+1} \in \mathbb{R}^{N\times m}$, $\boldsymbol{\beta}_i \in \mathbb{R}^{m\times m}$, $\mathbf{v}_{i+1}^{\mathrm{T}}\mathbf{B}\mathbf{v}_{i+1} = \mathbf{Id}_m$, and $\boldsymbol{\beta}_i$ upper triangular. Several techniques exist to do so. The first method considered in this work is the Modified Gram–Schmidt QR algorithm (MGSQR). It is based on the fact that the columns of $\mathbf{v}_{i+1}$ form an orthonormal base with respect to the **B** inner product of the space spanned by the columns of $\mathbf{w}_i$. The modified orthogonalization process of the Gram–Schmidt method allows to estimate these base vectors, and the coefficients of the matrix $\boldsymbol{\beta}_i$ are estimated as the projection coefficients of the process; the Appendix gives details. Another algorithm is the Cholesky QR method (CholQR; Stathopoulos and Wu, 2002). It is based on the fact that if $\mathbf{w}_i = \mathbf{v}_{i+1}\boldsymbol{\beta}_i$, we can write $\mathbf{w}_i^{\mathrm{T}}\mathbf{B}\mathbf{w}_i = \boldsymbol{\beta}_i^{\mathrm{T}}\mathbf{v}_{i+1}^{\mathrm{T}}\mathbf{B}\mathbf{v}_{i+1}\boldsymbol{\beta}_i = \boldsymbol{\beta}_i^{\mathrm{T}}\boldsymbol{\beta}_i$, thanks to the orthogonality of $\mathbf{v}_{i+1}$. Thus, $\boldsymbol{\beta}_i$ appears to be the Cholesky factor of $\mathbf{w}_i^{\mathrm{T}}\mathbf{B}\mathbf{w}_i$. Finally, solving the triangular system $\mathbf{v}_{i+1}\boldsymbol{\beta}_i = \mathbf{w}_i$ gives $\mathbf{v}_{i+1}$.

The last method, Pre-Cholesky QR (PRE-CHOLQR), algorithm 2 in Lowery and Langou (2014), is a variant of the previous one in which we perform in a first step a standard QR decomposition with respect to the canonical inner product of $\mathbf{w}_i$. Then, CholQR is used to perform an oblique QR decomposition of the Q factor resulting from the first step. All these algorithms are described and evaluated in terms of stability in Lowery and Langou (2014): overall, these experiments on small systems and various conditioning both for **B** and $\mathbf{w}_i$ show that PRE-CHOLQR is more stable than the other two, notably in terms of loss of orthogonality, but it is more costly as well. We have never found any serious orthogonality issue in our numerical experiments, nor have we found significant numerical differences between the three algorithms. So we finally decided not to use the more costly PRE-CHOLQR algorithm, and to work with MGSQR instead, detailed in the Appendix.

3.1.4 | Block B-Lanczos algorithm

In the case in which the Hessian **A** is symmetric, the matrix $\underline{\mathbf{T}}$ of the block B-FOM is symmetric (Saad, 2003, p. 185).

Using the notations of Algorithm 1, this leads to $\boldsymbol{\alpha}_j = 0$ for $j \notin \{i-1, i\}$ and $\boldsymbol{\alpha}_{i-1} = \boldsymbol{\beta}_i^{\mathrm{T}}$. Finally, this leads to a block tridiagonal matrix $\underline{\mathbf{T}}$ with blocks of size $m \cdot m$.

This results in a considerable simplification of the block B-FOM (the *for* loop at line 7 is replaced by the two cases $j = i-1; j = i$), and defines the block B-Lanczos algorithm.

However, it appears that, in finite precision because of the round-off errors, the orthogonality may not be ensured (Gürol *et al.*, 2014). It is therefore often necessary to add reorthogonalization in the block B-Lanczos procedure to ensure convergence (lines 8–9 of Algorithm 1 $\forall j$). Nevertheless, for systems in which p and m are small compared to the dimension N of the problem, the inversion of $\underline{\mathbf{T}}$ in Algorithm 1 is not costly compared to the inner products. In this case, it is better to use the block B-FOM algorithm which does not restrict $\underline{\mathbf{T}}$ to a block tridiagonal matrix.

3.2 | Case $m = 1$

The purpose of this paper is to evaluate the efficiency of block Krylov methods for minimizing the cost function of an EDA. To do so, we will compare the results of such a m-member block minimization with results given by m independent non-block minimizations (m minimizations with a single member). As briefly described in section 2.1, Krylov methods, such as the FOM, can be used to perform a single member minimization. Actually, the block B-FOM in Algorithm 1 applied with $m = 1$ results in the B-FOM. In the numerical sections of this paper, we will compare the results given by a m-member block B-FOM with results given by m independent applications of the B-FOM.

Algorithm 2 B-FOM

1: $\mathbf{z}_0 = \mathbf{B}\mathbf{r}_0$
2: $\beta_0 = \sqrt{(\mathbf{r}_0, \mathbf{z}_0)}$
3: $\mathbf{v}_1 = \mathbf{r}_0/\beta_0$
4: $\mathbf{z}_1 = \mathbf{z}_0/\beta_0$
5: $\mathbf{V}_1 = [\mathbf{v}_1]$
6: $\mathbf{Z}_1 = [\mathbf{z}_1]$
7: **for** $i = 1 : p-1$ **do**
8: $\quad \mathbf{w}_i = \mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}\mathbf{z}_i + \mathbf{v}_i$
9: $\quad$ **for** $j = 1 : i$ **do**
10: $\quad\quad \mathbf{T}_{j,i} = \left(\mathbf{w}_i, \mathbf{z}_j\right)$
11: $\quad\quad \mathbf{w}_i := \mathbf{w}_i - \mathbf{T}_{j,i}\mathbf{v}_j$
12: $\quad$ **end for**
13: $\quad \mathbf{z}_i = \mathbf{B}\mathbf{w}_i$
14: $\quad \mathbf{T}_{i+1,i} = \sqrt{(\mathbf{v}_i, \mathbf{z}_i)}$
15: $\quad \mathbf{v}_{i+1} = \mathbf{w}_i/\mathbf{T}_{i+1,i}$
16: $\quad \mathbf{z}_{i+1} = \mathbf{z}_i/\mathbf{T}_{i+1,i}$
17: $\quad \mathbf{V}_{i+1} := [\mathbf{V}_i; \mathbf{v}_{i+1}]$
18: $\quad \mathbf{Z}_{i+1} := [\mathbf{Z}_i; \mathbf{z}_{i+1}]$
19: **end for**
20: solve $\mathbf{T}_{1:p,1:p}\mathbf{s}_p = \beta_0\mathbf{e}_1$
21: $\boldsymbol{\delta}\mathbf{x}_p = \mathbf{V}_p\mathbf{s}_p$

For readability purposes, we rewrite in Algorithm 2 the B-FOM and recall that it can be compared to Algorithm 1 when it is applied m times, potentially concurrently, with a single member.

In Algorithm 2, we now have $\mathbf{w}_i, \mathbf{v}_i$, and $\mathbf{z}_i$ in $\mathbb{R}^N$ and the bases $\mathbf{V}_p$ and $\mathbf{Z}_p$ in $\mathbb{R}^{N\cdot p}$ at iteration p. Matrices $\boldsymbol{\alpha}_j$ and $\boldsymbol{\beta}_i$ of Algorithm 1 become scalars, directly stored in $\mathbf{T}$. The QR decompositions result in simple normalizations.

3.3 | First remarks on efficiency

It is known that the rate of convergence of Krylov and block Krylov methods is linked to the condition number of the Hessian and to the clustering of its eigenvalues. The condition number of a symmetric matrix is the ratio of the largest and the smallest eigenvalues. The greater the condition number, the slower the convergence. In NWP models, the unpreconditioned Hessian $\mathbf{B}^{-1} + \mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}$ is generally very poorly conditioned (cf. table 1 in Desroziers and Berre, 2012). Using $\mathbf{B}$ as a preconditioner reduces the condition number, with the lowest eigenvalue equal to 1, though the condition number will be equal to the largest eigenvalue. This could allow the minimization to be speeded up. Additionally, it is shown on a simplified model in Haben *et al.* (2011) that the condition number of $\mathbf{A}$ is linearly linked to the number of observations available and to $\sigma_{\mathrm{b}}^2/\sigma_{\mathrm{o}}^2$, with σ_{b}^2 the variance of the background errors and σ_{o}^2 the variance of the observations errors.

Concerning the differences between block B-FOM (Algorithm 1) and m times B-FOM (Algorithm 2), it can be observed that block B-FOM looks, for any member, for approximates of δx in a search subspace of size $m \cdot p$ (number of members $\times$ number of iterations) whereas B-FOM looks for approximate solutions in search subspaces of size p. So by using a larger search subspace, we expect a faster convergence with block methods. Moreover, both algorithms imply one application of each operator ($\mathbf{B}$, $\mathbf{H}$, $\mathbf{H}^{\mathrm{T}}$, and $\mathbf{R}^{-1}$) for each member and for each iteration (m applications by iteration). However, there are many more inner products and vector summations in block B-FOM than in m B-FOM. For instance, in block B-FOM, in the j loop at line 8 of Algorithm 1, we need m^2 inner products to calculate $\boldsymbol{\alpha}_j$, compared to m inner products for m B-FOM (line 10 of Algorithm 2 applied m times). Because we expect fewer iterations to converge with block B-FOM, we can already guess that the block approach will be much more efficient if the main part of the computational cost lies in the application of the operators (typically $\mathbf{B}$ or $\mathbf{H}$) and less efficient if inner products and vector summations are very costly. Although this question will be addressed in detail in section 5, we can already note that this is a very good reason for using dual space algorithms. In the dual space, vectors of lower dimensions ($N_{\mathrm{obs}} \ll N$) are used so that the inner products and vector summations will be less costly while the number of operators to be used will be the same. In the next section, block Krylov methods in dual space are introduced.

3.4 | Block Krylov methods in dual space

Similarly to Equation 11 in primal space, we can write the ensemble problem in dual space defined by Equation 9 under the block form:

$$\widehat{\mathbf{A}}\lambda = \widehat{\mathbf{b}},$$

with $\widehat{\mathbf{A}}$ the Hessian in dual space with left $\mathbf{R}^{-1}$ preconditioning $\widehat{\mathbf{A}} = \mathbf{Id}_{\mathbf{N}_{\mathrm{obs}}} + \mathbf{R}^{-1}\mathbf{H}\mathbf{B}\mathbf{H}^{\mathrm{T}}$, $\lambda = \lambda_1 \cdots \lambda_m$, and $\widehat{\mathbf{b}} = \widehat{\mathbf{b}}_1 \cdots \widehat{\mathbf{b}}_m$, $\widehat{\mathbf{b}}_k = \mathbf{R}^{-1}\mathbf{d}_k, k \in [\![1; m]\!]$.

Then, if we use a Krylov method equipped with the $\mathbf{HBH}^{\mathrm{T}}$-inner product to solve this linear system, it will yield the same iterations as in the primal space.

The Full Orthogonalization Method applied under these conditions gives the RB-FOM algorithm detailed in Algorithm 3. R stands for "Restricted" (Gratton *et al.*, 2011; Gürol *et al.*, 2014.

4 | IMPLEMENTATION

4.1 | Oriented object prediction system

OOPS is a generic driving layer for NWP systems initiated by ECMWF and developed both at ECMWF and Météo-France(Fisher *et al.*, 2011; Arbogast *et al.*, 2017; Bonavita *et al.*, 2017). The purpose of OOPS is to develop a flexible code, written in the object-oriented language C++, to deal with the complex codes of NWP and oceanography. It was notably developed to make easier the development

Algorithm 3 Block RB-FOM

1: $\widehat{\mathbf{z}}_0 = \mathbf{HBH}^{\mathrm{T}}\widehat{\mathbf{r}}_0$
2: $\left[\widehat{\mathbf{v}}_1, \widehat{\mathbf{z}}_1, \boldsymbol{\beta}_0\right] = qr\left(\widehat{\mathbf{r}}_0, \widehat{\mathbf{z}}_0\right)$
3: $\widehat{\mathbf{V}}_1 = [\widehat{\mathbf{v}}_1]$
4: $\widehat{\mathbf{Z}}_1 = [\widehat{\mathbf{z}}_1]$
5: **for** $i = 1 : p-1$ **do**
6: $\quad \widehat{\mathbf{w}}_i = \mathbf{R}^{-1}\widehat{\mathbf{z}}_i + \widehat{\mathbf{v}}_i$
7: $\quad$ **for** $j = 1 : i$ **do**
8: $\quad\quad \boldsymbol{\alpha}_j = \widehat{\mathbf{z}}_j^{\mathrm{T}}\widehat{\mathbf{w}}_i$
9: $\quad\quad \widehat{\mathbf{w}}_i := \widehat{\mathbf{w}}_i - \widehat{\mathbf{v}}_j\boldsymbol{\alpha}_j$
10: $\quad\quad \underline{\mathbf{T}}_{jm:(j+1)m,im:(i+1)m} = \boldsymbol{\alpha}_j$
11: $\quad$ **end for**
12: $\quad \widehat{\mathbf{z}}_i = \mathbf{HBH}^{\mathrm{T}}\widehat{\mathbf{w}}_i$
13: $\quad \left[\widehat{\mathbf{v}}_{i+1}, \widehat{\mathbf{z}}_{i+1}, \boldsymbol{\beta}_i\right] = qr\left(\widehat{\mathbf{w}}_i, \widehat{\mathbf{z}}_i\right)$
14: $\quad \underline{\mathbf{T}}_{(i+1)m:(i+2)m,im:(i+1)m} = \boldsymbol{\beta}_i$
15: $\quad \widehat{\mathbf{V}}_{i+1} := [\widehat{\mathbf{V}}_i; \widehat{\mathbf{v}}_{i+1}]$
16: $\quad \widehat{\mathbf{Z}}_{i+1} := [\widehat{\mathbf{Z}}_i; \widehat{\mathbf{z}}_{i+1}]$
17: **end for**
18: solve $\underline{\mathbf{T}}_{1:p\cdot m,1:p\cdot m}\widehat{\mathbf{s}}_p = \mathbf{e}_1\boldsymbol{\beta}_0$
19: $\lambda_p = \widehat{\mathbf{V}}\widehat{\mathbf{s}}_p$

and assessment of new data assimilation systems. OOPS uses generic abstract objects to represent the limited number of quantities used in data assimilation, such as operators ($\mathbf{B}$, $\mathbf{H}$, $\mathbf{H}^{\mathrm{T}}$, and $\mathbf{R}^{-1}$), model states ($\mathbf{x}$), increments in primal or dual spaces ($\delta\mathbf{x}$, $\mathbf{u}$, and λ), observation vectors ($\mathbf{y}^{\mathrm{o}}$) or cost functions (J). This part of OOPS is independent of the underlying model generally written in FORTRAN. This model and the interface layer of OOPS are modified in order to be coupled. Even if it is not yet used operationally, several models have been adapted to OOPS, including theEuropean ocean model NEMO and the ECMWF NWP model IFS (Bonavita *et al.*, 2017), and both NWP models of Météo France, ARPEGE (Arbogast *et al.*, 2017) and AROME, in their deterministic and EDA versions.

In this work, we have developed our block Krylov algorithms in the abstract layer of OOPS and have interfaced these algorithms with a FORTRAN model: the quasi-geostrophic (QG) numerical model (Fisher *et al.*, 2011), cf. section 5, a simplified model of atmospheric circulation in midlatitudes. Distributed memory parallelism through the Message Passing Interface (MPI) is possible in the OOPS C++ layer. However, there can also be another MPI parallelization layer in the underlying FORTRAN code of the models (e.g. in AROME), even if it will not be the case for the QG model.

In this section, we introduce the three versions (mathematically equivalent) of the block Krylov algorithms that we have implemented in OOPS. The first one in section 4.2 does not include any parallelism in OOPS, while the two others in sections 4.3 and 4.4 do. For introducing these codes, we use as an illustration the block B-FOM described in Algorithm 1 (in primal space). However, the conclusions which will be drawn are also applicable for dual space block RB-FOM.

4.2 | Sequential version

The first version of block B-FOM implemented in OOPS will be referred to as the "SEQ" version. This is a sequential implementation in the sense that there is no parallelization included in the OOPS layer. It can nevertheless include a MPI parallelization but only over the geographical domain (if such a parallelization is available). Figure 1a illustrates this situation for $m = 2$ and for four MPI processes on the Western Europe domain of AROME-France. Each process deals with small geographical areas but for all members in the ensemble. This implies for each process to perform more applications of operators and inner products/vector summations but with smaller vectors.

Then in the SEQ version, all the computations implied by Algorithm 1 are defined as loops on members. For instance, we detail the algorithm used to calculate line 6 of block B-FOM: application of the Hessian at iteration i to compute the new search directions.

for $k = 1 : m$ **do**
$\quad (\mathbf{w}_i)_k = \mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}\,(\mathbf{z}_i)_k + (\mathbf{v}_i)_k$
end for

And for line 8 of Algorithm 1, at iteration i:

for $k = 1 : m$ **do**
$\quad$ **for** $l = 1 : m$ **do**
$\quad\quad \left(\boldsymbol{\alpha}_j\right)_{l,k} = \left((\mathbf{w}_i)_k, \left(\mathbf{z}_j\right)_l\right)$
$\quad$ **end for**
end for

The next paragraph explains to what extent this SEQ approach could be competitive with a NWP model including an underlying geographical parallelization.

Suppose that we run EDA experiments with m members using N^{MPI} MPI processes. Our interest is to compare the speed of convergence between concurrent independent B-FOM and block B-FOM. First, in the non-block case, each minimization for each member will be run on N^{MPI}/m processes. So the geographical domain of the NWP model will be divided into N^{MPI}/m parts. Figure 1b illustrates this situation with $m = 2$ and $N^{MPI} = 4$. Processes 1 and 2 perform the minimization for the first member, while processes 3 and 4 perform the minimization for the second member. On the other hand, the SEQ block B-FOM algorithm has no OOPS parallelization, so that when it will be run on N^{MPI} processes, the geographical domain will be divided into N^{MPI} parts, giving a finer decomposition as is illustrated on Figure 1a with $m = 2$, $N^{\mathrm{MPI}} = 4$: the geographical domain is divided into four parts, and each process deals with both members.

Table 1 summarizes this analysis and compares the number of operator applications, inner products, and vector summations for non-block and SEQ block versions (columns 2

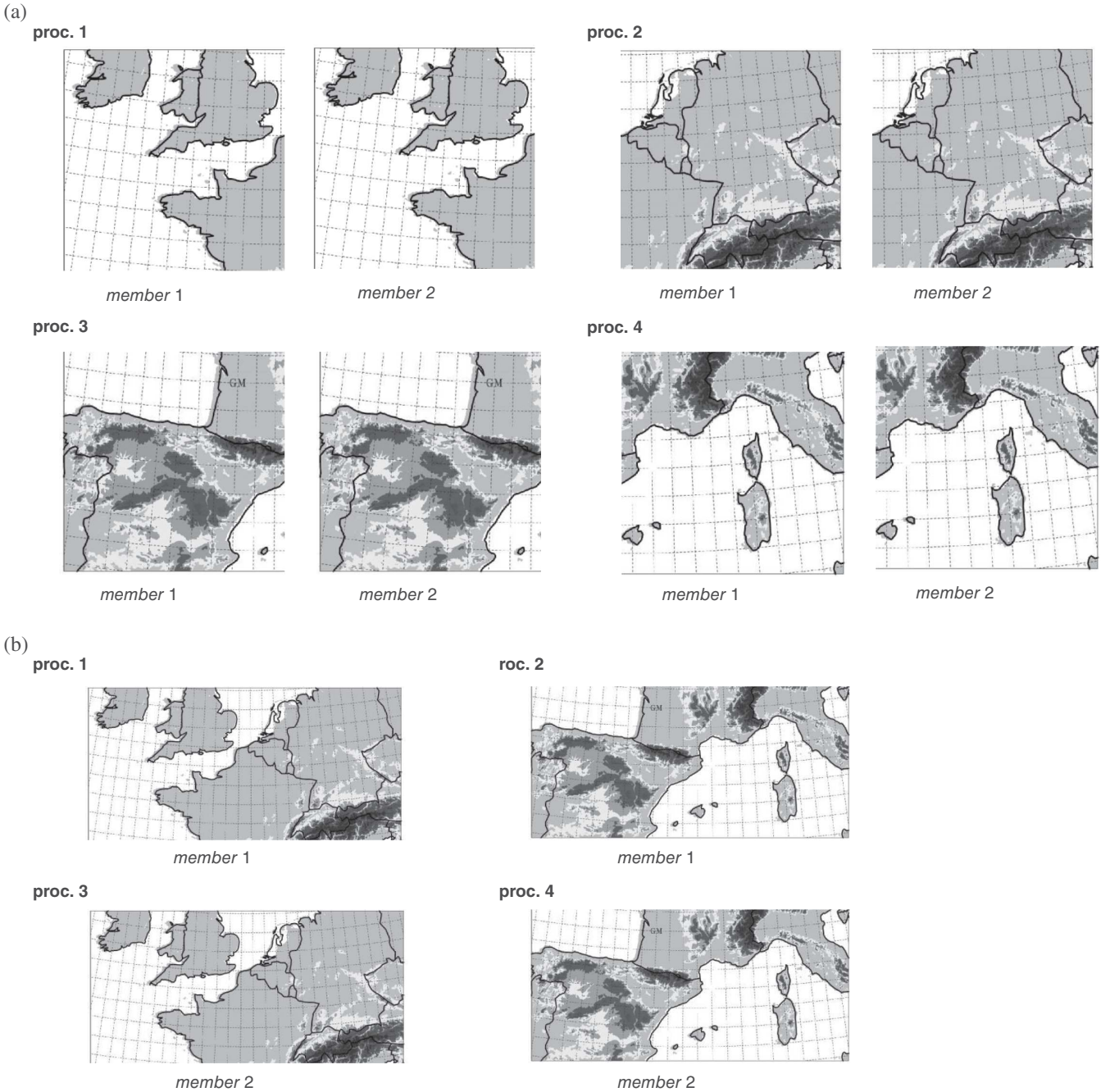

FIGURE 1 Illustration of the two workload distributions implemented in this work on the AROME-France domain for four MPI processes and two members in the ensemble. (a) Workload geographical distribution (no distribution by member), as used in the SEQ version of block B-FOM. (b) Workload distribution by member, combined with an underlying geographical distribution, as used in the MPI and MPIstored versions of block B-FOM, and in B-FOM

and 3) for one iteration of the minimizer with $N^{\mathrm{MPI}} = m$. The SEQ method leads to m times more operator applications, and to roughly m^2 more inner products and vector summations. But the geographical decomposition is m times finer than in the non-block version, so that we can expect the same computational time for the operator applications and m more computational time for the inner products and vector summations. This is true if we neglect the communication time between geographical domains, which is not obvious. Knowing that we expect fewer iterations to converge with the block version, this approach could be competitive, especially if the operators are costly compared to the inner products.

If there is no parallelization in the underlying FORTRAN model, this SEQ version has no interest. All the MPI processes would run the same entire problem, and it should be faster to perform m non-block B-FOM concurrently. This is the case for the QG model described in section 5, so no results using the SEQ version will be studied in this paper. It has nevertheless been described here both as a simpler framework before further MPI developments and for future use in NWP models including geographical parallelization.

TABLE 1 Characteristics of the different algorithms compared in this paper

	Non-block	SEQ	MPI	MPIstored
Number of MPI processes	m	m	m	m
Proportion of the geographical domain for each process	1	$1/m$	1	1
B, **H** applications (/iteration/process)	1	m	1	1
Inner products, vector summations (/iteration/process)	$j+1$	$jm^2 + m(m+1)/2$	$jm + k$	$jm + k$
Increments exchanged in OOPS	0	0	$(4+2j)(m-1)^2$	$3(m-1)^2$

j =current iteration of the minimization process, m =number of members in the EDA, and k =current member

4.3 | MPI version

Another version of block B-FOM, referred to as the "MPI" version, implies a workload distribution by member. This distribution can be combined with an underlying geographical decomposition. Figure 1b illustrates this situation, still with $m = 2$ and four MPI processes. Now each process deals only with one member, but on larger geographical areas. This reduces the number of operator applications and inner products/vector summations but they have to be performed with larger vectors.

Under these conditions, line 6 of block B-FOM becomes very simple, without any loop. For example, for the process tied to member k:

$$(\mathbf{w}_i)_k = \mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1}\mathbf{H}\,(\mathbf{z}_i)_k + (\mathbf{v_i})_k$$

This implies of course communication between processes at the OOPS level, for instance to compute the projections. Line 8 of Algorithm 1 becomes:

for $l = 1 : m$ **do**
  $l' = (m - k + l) \bmod m$
  if $(l' \neq k)$ **then**
    send $\left(\mathbf{z}_j\right)_k$ to proc l'
    get $\left(\mathbf{z}_j\right)_{l'}$ from proc l'
    $\left(\boldsymbol{\alpha}_j\right)_{l',k} = \left((\mathbf{z}_j)_{l'}, (\mathbf{w}_i)_k\right)$
  else
    $\left(\boldsymbol{\alpha}_j\right)_{k,k} = \left((\mathbf{z}_j)_k, (\mathbf{w}_i)_k\right)$
  end if
end for

Here, for member k at iteration i, we want to compute the kth column of $\boldsymbol{\alpha}_j$, namely the projection of $(\mathbf{w}_i)_k$ on all the $\left(\mathbf{z}_j\right)_l$. The permutation implied by the definition of l' is the one used by Arbogast *et al.* (2017) to speed up the exchanges in a similar context. So this version implies many communications and the OOPS decomposition by member implies a rougher geographical decomposition. While the geographical domain was divided into N^{MPI} parts with the SEQ version, it is divided in N^{MPI}/m parts in this MPI version, as in the non-block version (Figure 1b). Table 1 presents the number of operator applications, inner products, and vector summations implied by this method, as well as the total number of communications per iteration. Compared to the non-block version, even if there are more inner products and vector summations, the number of operator applications is the same, so that we can expect benefits if fewer iterations are needed to reach convergence and if it is costlier to apply the operators than computing inner products, which is expected at least in dual space. Compared to the SEQ version, this MPI version could be competitive only when dealing with extremely fine geographical decompositions.

For the number of increments exchanged in OOPS, we note that it is proportional to m^2 and j, the index of the current iteration. Indeed, we need to exchange all the previous base vectors to all processes at each iteration for the reorthogonalization part of block B-FOM (lines 7–11 of Algorithm 1). An idea for avoiding these communications would be to store the full Krylov bases corresponding to all members on each MPI process. This is the purpose of the next section.

4.4 | MPI stored version

The third version of block B-FOM, referred to as the "MPI stored" version, is a variant of the MPI version which reduces the number of communications at the expense of increased memory. In this version, the full block Krylov base is stored redundantly on all processes. Computationally, it is nevertheless very similar but, at the end of each iteration, all processes send their new base vectors to all the others.

Consequently, we do not need any communication in the reorthogonalization part of Algorithm 1 (lines 7–11), given that the full $\mathbf{w}_i$ and $\mathbf{z}_j$ are stored on each process. Line 8 of Algorithm 1 becomes:

for $l = 1 : m$ **do**
  $\left(\boldsymbol{\alpha}_j\right)_{l,k} = \left((\mathbf{z}_j)_l, (\mathbf{w}_i)_k\right)$
end for

We can see in Table 1 that we avoid the dependence in j of the number of increments exchanged in OOPS by iteration.

The disadvantage of this approach is that we store all the Krylov bases m times. This implies storing m times objects of dimension $2 \cdot j \cdot m \cdot N$ (two base vectors of dimension N per member and per iteration, though divided in reality in N^{MPI}/m pools by the geographical decomposition). This can be intractable if m, j, and/or the dimension of the problem are

large. So we can guess that this method will be mainly usable in the dual space approach in which smaller dimensions are in play. In the case that is considered in this paper, working in dual space with $N_{\text{obs}} = 2 \times 10^4$, 40 iterations and 25 members, allows us to handle vectors of dimension $(2 \cdot j \cdot m \cdot N)$ upper bounded by 4×10^7. Given that these experiments will be run on m supercomputer nodes, each with $64GB$ of memory, memory will not be an issue. For future applications with the AROME France NWP system, we expect to run ensembles of 75 members with around 10^5 assimilated observations. With 20 iterations, this implies storing vectors of dimension around 3×10^8, which is still a tractable value given that 2–5 nodes by member are generally available to perform such experiments.

5 | RESULTS ON THE QG MODEL

5.1 | The two-layers QG model

The QG model (Fandry and Leslie, 1984) allows to mimic QG flow in a cyclic channel with two layers. It is widely used as a simple way to realistically represent a large part of large-scale dynamics in the atmosphere in midlatitudes, notably to understand and to model cyclogenesis and storms (e.g. Plu and Arbogast, 2005). A version of the QG model has been implemented in OOPS at ECMWF in order to test different assimilation strategies in a simple framework (Fisher *et al.*, 2011). We choose as well to test these new block Krylov developments on this model because it is available in OOPS and the developments made for the QG model should be easily portable to the OOPS version of AROME. Moreover, it is easy to tune for having realistic characteristics (see below). It also allows us to test our code in a fully controlled and "perfect" framework: we have a well-posed problem with, for instance, the same operator **H** for all members. In this section, we briefly give the equations of the QG model. Readers can refer to Snyder *et al.* (2003) or Fisher *et al.* (2011) for more details about this framework and its use in data assimilation. The values of the parameters used in this study as well as the discretization strategy are detailed later in section 5.2.

The QG model is based on the conservation of the potential vorticity. In the equations below, subscripts 1 and 2 denote respectively the top and bottom layers of the model. q is the potential vorticity, defined as:

$$\begin{cases} q_1 = \Delta\Psi_1 - F_1(\Psi_1 - \Psi_2) + \beta y, \\ q_2 = \Delta\Psi_2 - F_2(\Psi_2 - \Psi_1) + \beta y + R_{\text{s}}, \end{cases} \tag{18}$$

with Δ the Laplacian, Ψ the two-dimensional stream function, β the northward derivative of the Coriolis parameter, R_{rms} a coefficient representing the ground effects in the bottom layer (orography, heating), and F_1 and F_2 constant coefficients allowing the non-dimensionalization (Fisher *et al.*, 2011 gives their detailed expressions), depending notably on the depths of the two layers, D_1 and D_2.

The Lagrangian conservation of q can be written:

$$\begin{cases} \dfrac{\mathrm{D}q_1}{\mathrm{D}t} = \dfrac{\delta q_1}{\delta t} + u_1 \dfrac{\delta q_1}{\delta x} + v_1 \dfrac{\delta q_1}{\delta y} = 0, \\ \dfrac{\mathrm{D}q_2}{\mathrm{D}t} = \dfrac{\delta q_2}{\delta t} + u_2 \dfrac{\delta q_2}{\delta x} + v_2 \dfrac{\delta q_2}{\delta y} = 0, \end{cases} \tag{19}$$

with u and v the zonal and meridional 2D wind fields, which can be derived from the stream function:

$$u_i = \frac{\delta\Psi_i}{\delta x}, \qquad v_i = \frac{\delta\Psi_i}{\delta y}. \tag{20}$$

The numerical resolution of this model is done with a pseudo-spectral method which involves the following sequence at each time step:

- Propagate q_1 and q_2 to the next time step using their conservation Equation 19, using a semi-Lagrangian scheme;
- Invert Equation 18 to derive the stream function;
- Apply Equation 20 to get the horizontal wind fields in both layers.

This model is relevant for the large-scale dynamics of the atmosphere in terms of error growth, as it represents baroclinic instability (Fisher *et al.*, 2011).

5.2 | Experimental framework

5.2.1 | Parametrization of the QG model

The domain on which these experiments are performed covers an horizontal area of $12{,}000 \times 6{,}300$ km^2, discretized on a grid with 640×320 points. The spatial resolution is around 20 km in both horizontal directions. Then, the dimension of the problem will be $N = 2 \times 640 \times 320 \approx 4 \times 10^5$.

The true state and the true observations are generated using this model with depths of $D_1 = 6$ and $D_2 = 4\,km$ for the vertical layers. There are $N_{\text{obs}} = 1.2 \times 10^4$ observations of the stream function randomly distributed over the domain. Both the background and the observations are perturbed offline by appropriate codes in order to generate an ensemble. These perturbations are made according to the chosen statistics of errors. For the observation, **R** is supposed diagonal, with a constant standard deviation of $\sigma_{\text{o}} = 0.4$. Concerning the background, the horizontal correlations are supposed to be Gaussian, isotropic, and homogeneous, with a length-scale of 1000 km, as in Fisher *et al.* (2011). The standard deviation is set to 1.6. The vertical correlation coefficient between the two layers is set to 0.2. We also add a model error by using in the assimilation step $D_1 = 5.5$ km and $D_2 = 4.5$ km for the depths of the layers.

The dimensions of this system have been chosen in order to be tractable but realistic to allowing us to be confident in the portability of the results to the AROME model. First, we note that, as has been mentioned in section 4, there is no underlying geographical workload distribution in the QG OOPS code. Consequently it is not possible to extent the QG model dimensions to values comparable to the AROME ones. That is why we limit ourselves to $N = 4{\times}10^5$, yielding a large but tractable

system. Then, we remark that in dual space, the performance of the system will be driven both by N via the operators acting in primal space, e.g. $\mathbf{B}$, and N_{obs}. So we choose a value for N_{obs} close to the one acting in the operational EDA of AROME (1.2×10^4 versus 10^4–10^5, depending on the weather conditions). It will nevertheless be inferior by an order of magnitude to the N_{obs} used in the operational deterministic system or expected in a few years even for the EDA.

We have seen in section 3.1 that the speed of convergence of the block Krylov methods is mainly driven by the condition number of the problem and by the spectrum of the Hessian in general. The condition number is closely related to the covariances of the background and observation errors, as well as to the number of observations available. So it is quite important to use realistic values for all these parameters. In AROME, the number of assimilated observationsdepends highly on the weather situation. Notably, radars provide usable information when it is raining over France, so that heavy rain tends to significantly increase the condition number.

In AROME experiments carried on both for rainy and dry days, we have found that the condition number ranged from 2×10^2 up to 6×10^3, corresponding to 1.4×10^4 to 4.9×10^4 assimilated observations. In the QG experiment described here, we have a condition number around 9×10^3. We can reasonably believe that this matches the worst possible case of AROME. However, the relative dimensions of both problems produce eigenvalues decreasing much faster in our QG experiment than in any case of AROME. Eventually, what we can expect in the block Krylov methods applied to this QG problem is a convergence similar to the one expected with AROME in the first iterations of the minimization processes, but ultimately a faster convergence after a few iterations. So we have created a reasonably realistic model given the dimension constraints, but we still need to be careful before extrapolating the results to real systems.

5.2.2 | Data assimilation experiments carried out

We finally describe the EDA experiments carried out with this QG model.

First, it has to be noted that, in the QG experiments, we have found no numerical differences between all the algorithms described previously, namely the SEQ, MPI and MPIstored versions, in primal and dual space. Second, we recall that the QG model has no underlying geographical workload distribution, so the SEQ version has no computational interest. Thus we will compare the time performances of four algorithms: the MPI version of the block B-FOM (B-MPI) and of the dual block RB-FOM (RB-MPI), and the corresponding primal (B-MPIstored) and dual (RB-MPIstored) versions of the MPIstored algorithm. Regarding the number of members m considered, this will be set to 1, 5, 10, 20, or 40. All the experiments will use m MPI processes. This strategy allows to estimate the time needed to perform EDA minimization: for example, a one-member experiment run on one MPI process would need the same time to complete as 40 independent one-member experiments launched concurrently on 40 MPI processes.

5.3 | Numerical results

In this section, the results for the QG EDA experiments are presented, first in terms of number of iterations, and then with respect to wall-clock time.

5.3.1 | Convergence

We firstly study the number of iterations needed to reach convergence depending on the number of members in the ensemble. Figure 2 shows the convergence in terms of iterations according to three different criteria. We note again that the numerical results are the same for the four algorithms used here, so that there is no distinction between them. Figure 2a shows the $\mathbf{B}$-norm of the residuals for the first member, i.e. at iteration p, $\mathbf{r}_{p_1} = \left\|\mathbf{A}\left(\delta\mathbf{x}_p\right)_1 - \mathbf{b}_1\right\|_{\mathbf{B}}$. We can see that the convergence speed, which is quite slow for the one-member case, increases on a regular basis when the number of members increases. Figure 2b shows the background part of the cost function J_{b} which converges to a value of approximately 2.1×10^2 at the end of the minimization process. This final value is reached sooner and sooner when the number of members increases. We note that $J_{\text{b}} = \|\delta\mathbf{x}\|_{\mathbf{B}}^2/2$. It shows that this criterion can be estimated through the minimization process, with the notations of Algorithm 1, as: $\left(\mathbf{V}\mathbf{s}_p\right)^{\text{T}}\mathbf{Z}\mathbf{s}_p/2$. Figure 2c shows the distance to its final value of the linearized cost function along the minimization processes: $J_{\text{lin}} - J_{\text{lin}}^*$, with J_{lin}^* being the value of J_{lin} at the end of the minimization. The same conclusion can be drawn: convergence is reached faster as the number of members increases.

Now we choose a criterion for stopping the minimization. Several criteria can be used. The ECMWF system uses a criterion based on the relative variation of J_{b} from an iteration to the next one: the minimization is stopped when it drops below 5%, but with three outer loops to ensure convergence to the nonlinear problem (Trémolet, 2007). In this paper, we choose a criterion based on the one currently acting operationally in deterministic AROME. In this system, the minimization is stopped after 40 iterations. So we set the stopping criterion at 40 iterations for the non-block one-member minimization, and we note the corresponding value of the $\mathbf{B}$-norm of residuals (dotted horizontal grey line in Figure 2a). Then, block methods with various number of members are considered as converged when they reach the same value for the $\mathbf{B}$-norm of residuals. This is obviously an *a posteriori* criterion, but it allows us to easily compare the non-block and block methods. Column 2 of Table 2 gives the corresponding number of iterations needed to converge for different numbers of members. We notice that the number of iterations is divided by more than four compared to the non-block method when using more than 20 members. We also note in Figure 2b that applying a

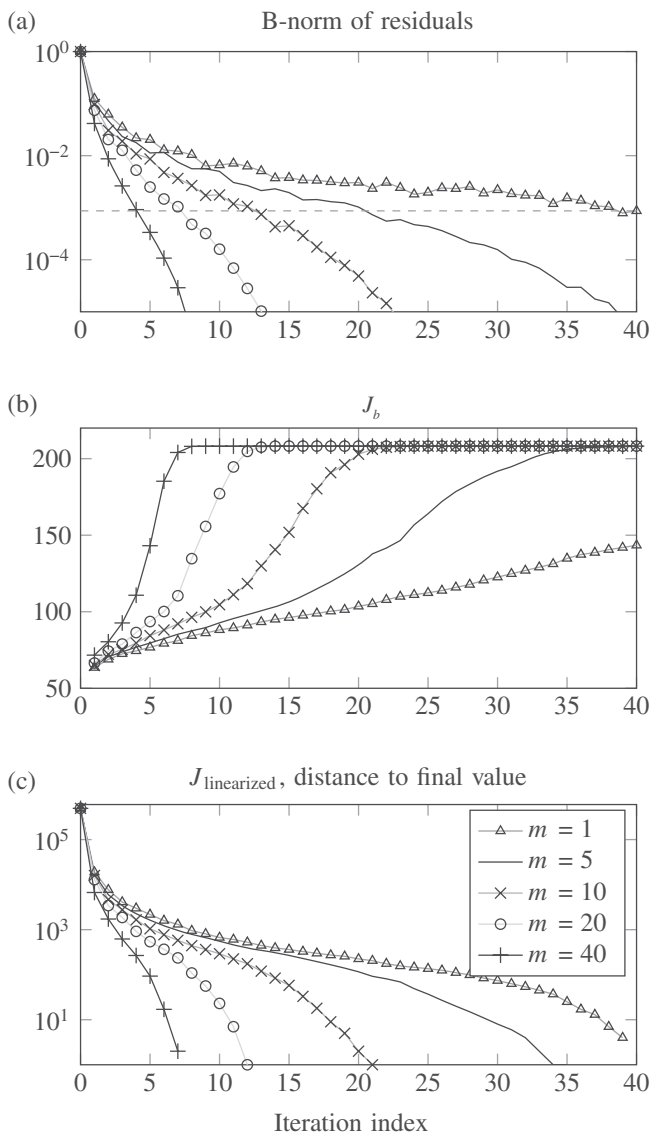

FIGURE 2 For different numbers of members in the ensemble (m from 1 to 40), convergence of the minimization for the first member in terms of iterations for different criteria: (a) **B**-norm of the residuals, (b) background part of the cost function, J_b, and (c) distance to the final value of the linearized cost function, J_{lin}

comparable criterion but based on J_b instead of $(\mathbf{r}_p)_1$ would have led to similar results.

5.3.2 | Wall-clock time

Increasing the number of members thus allows us to decrease the number of iterations needed to reach convergence, at the cost of more expensive iterations. In this section, the final cost of the minimization is studied in order to quantify the potential efficiency of block methods for speeding up the minimization process overall. We firstly explore the time needed to perform two individual iterations of the minimization, according to the algorithm used and the number of members in the ensemble. This is the purpose of Figure 3.

For the B-MPI algorithm in primal space, Figure 3a, we see that the computational time dramatically increases when the number of members increases. It takes less than 0.5 s to perform iteration 2 with one member, but around 3 s with forty members. With forty members, iteration 15 takes 12.5 s. The B-MPI algorithm is not scalable and cannot compete against non-block methods. On the one hand, the time spent in the operators is quite constant with respect to the number of members and to the iteration index. This was expected since we perform in any case one application of each operator by member and by iteration. On the other hand, the time spent in inner products, vector summations, and communications soar with the number of members and the iteration index. The block minimization is thus intractable.

TABLE 2 For different number of members, m, the number of iterations needed to reach convergence (details in the text) and corresponding computation times (s) needed to perform the minimization process (columns 3–6) for the four algorithms tested here. For each algorithm, the best time performance is in bold

	No. of	Primal		Dual	
m	iterations	B-MPI	B-MPIst	RB-MPI	RB-MPIst
1	40	**18.3**	18.1	16.60	16.90
5	22	22.8	13.6	9.89	9.60
10	14	21.6	11.0	6.45	6.32
20	9	20.7	**10.4**	4.36	4.27
40	6	25.6	12.5	**3.23**	**3.10**

The goal of the B-MPIstored algorithm (B-MPIst in Figure 3a) is to decrease the number of communications. While the time spent in the operators remains obviously the same, the time spent in other tasks significantly decreases. For instance, it goes from 12.5 s to 3 s for the 15th iteration of the minimization with 40 members. This proves that the communications are quite costly and that it is important to reduce their number. Nevertheless, the time needed to perform the 15th iteration is still eight times greater than the one needed to perform the same iteration with the non-block one-member version. Even if we perform far fewer iterations with forty members, it is not sure that this B-MPIstored algorithm will be able to improve the overall results. In order to decrease the cost of the inner products and vector summations, and to further decrease the cost of the communications, we have to work in the dual space.

The results for the dual space algorithm without base storage (RB-MPI) are presented in Figure 3b. Now, no matter the number of members, most of time is spent in the operators. Then, the time needed to perform iteration 15 with 40 members is no more than twice the time needed to perform the same iteration with one member. This is due to the fact that dual vectors have smaller dimension than primal ones: $N_{\text{obs}} = 1.2 \times 10^4$ versus $N = 4 \times 10^5$. This decreases dramatically the cost of the inner products, vector summations, and communications, since vectors of smaller dimensions are used. The cost of the operators remains the same.

The RB-MPIstored algorithm (RB-MPIst in Figure 3b) further reduces the number of communications compared to

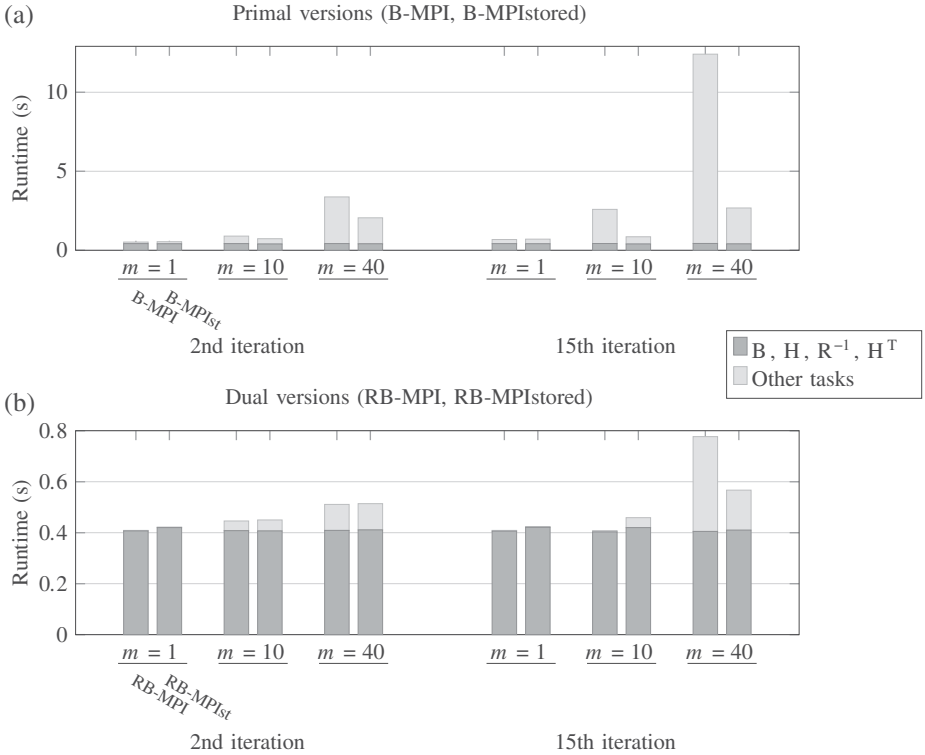

FIGURE 3 Computational time for the second iteration (six left bars of each plot) and the 15th iteration (six right bars) of the minimization, according to the number of members (m) in the ensemble, for four different algorithms: (a) primal versions B-MPI and B-MPIstored, and (b) dual versions RB-MPI and RB-MPIstored. Dark shading denotes time spent in the operators, and light shading in other tasks (axpys, inner products, communications) [AUTHOR: The abbreviation 'axpys' has not been used elsewhere.]

RB-MPI. With this approach, the time needed to compute an iteration is almost independent of the number of members and of the iteration index. For instance, 0.4 s are needed to compute iteration 15 with one member, and less than 0.6 s with 40 members. Since increasing the number of members allows us to decrease the number of iterations, we guess that this method will allow us to decrease the computational time needed to perform the full minimization.

The time spent in tasks other than operator applications for the best described algorithm, namely the RB-MPIstored, cf. grey part of Figure 3b, is decomposed in Figure 4 for iteration 15. These other tasks are divided in three groups: orthogonalization (Algorithm 3, lines 7–11 with $j = i - 1, i$) and reorthogonalization (Algorithm 3, lines 7–11 with $j < i - 1$) phases of RB-FOM (inner products and vector summations), and QR decomposition, which includes all the MPI communications in the MPIstored version of RB-FOM. We can see in Figure 4 that the reorthogonalization and the QR decomposition roughly account for the same part of the total extra-operator time consumption, whatever the number of members. This time consumption is proportional to the number of members for the reorthogonalization, which was expected (inner products and vector summations; Table 1).

Now, we can look at the total time needed to perform the minimization process, given that this process ends when we reach the criterion defined in section 5.3.1, which corresponds to the number of iterations listed in Table 2.

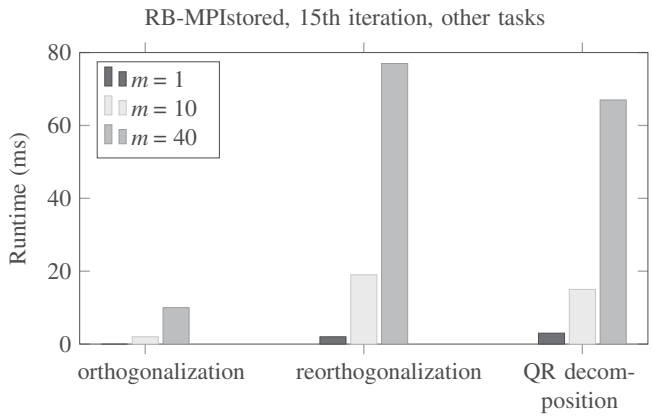

FIGURE 4 For the RB-MPIstored algorithm at iteration 15, time spent in tasks which are not operator applications for different numbers of members (corresponding to the light shading part of the three right bars on Figure 3d). These tasks are orthogonalization and reorthogonalization parts of RB-FOM (lines 7–11 of Algorithm 3) and QR decomposition and MPI communications (lines 13–16 of Algorithm 3)

These results for the four algorithms are plotted in Figure 5, and summarized in columns 3–6 of Table 2. In Figure 5a corresponds to the primal algorithms while Figure 5b shows the dual ones. On each plot, bars show the total time needed to perform the minimization for different number of members. We still distinguish the time spent in the operators (dark) and in the other tasks (grey). This figure confirms the previous results. With the B-MPI algorithm, the best results are obtained with the non-block version ($m = 1$). The inner

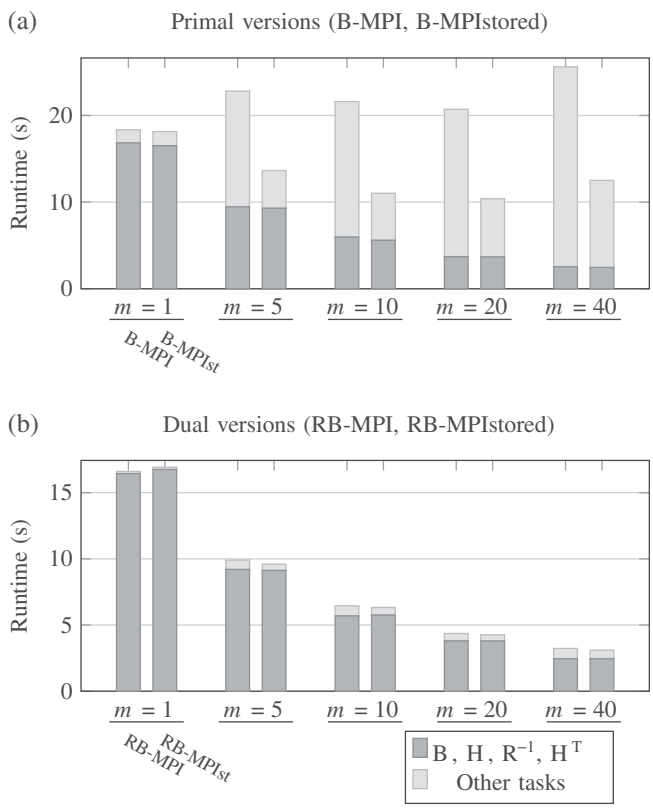

FIGURE 5 As Figure 3, but showing total computational time for performing the minimization process, according to the number of members in the ensemble.

products and vector summations are too costly to have any advantage in using block methods, even if they require far fewer iterations. In block methods, the number of inner products and vector summations scales as m, and the number of communications as $(m-1)^2$. This explains the increase with m in the time spent in other tasks in Figure 5a despite the decrease of the number of iterations that is performed. The reduction of the control vector dimension and of the communication number is thus crucial. Then if we store the bases with B-MPIstored (B-MPIst in Figure 5a) we obtain intermediate results. The performance of the algorithm is best with twenty members, but the gains are not really convincing. All methods imply a total computational time ranging from 10 to 20 s. The use of block methods starts to be attractive when using the dual space, as in algorithm RB-MPI (Figure 5b). We clearly see that increasing the number of members speeds up the minimization. We have even better results with RB-MPIstored (RB-MPIst in Figure 5b). Because most time is spent in operators, the time needed by the iteration is almost independent of the number of members. Then, the total computational time is around 3 s for 40 members compared to the 17 s of the non-block one-member version (Table 2). We also note in this table that the non-block version has performance almost independent of the algorithm used: 17 to 18 s in any case, almost completely spent in the operators.

6 | CONCLUSION

In this paper, block Krylov methods have been considered for solving ensembles of variational data assimilations. In a linearized context, the EDA consists of an ensemble of linear systems sharing the same Hessian but with different right-hand sides. In this study, this implies linearization of the observation operator around the background ensemble mean and performing no quality control after the observation perturbation. Then Block Krylov methods are a natural solution to this problem. We propose to use the block Full Orthogonal Method, and derive variants both in primal space and in dual space with suitable preconditioning and convergence properties. We note that we limit ourselves to an assimilation system without outer loops.

The implementation of these algorithms has been made under OOPS, in order to be easily applicable to different numerical weather and ocean prediction systems in the future. Several parallelization strategies have been developed to make the minimization tractable in terms of memory and efficient in terms of time consumption, relying on the infrastructure developed by Arbogast *et al.* (2017). Finally, all these implementations have been tested on a QG numerical system with an ensemble of 3D-Var, a simplified but nevertheless realistic model. The dimension of its domain (4×10^5) was much smaller than is commonly used in NWP systems, but it has been run with a number of observations (1.2×10^4), consistent with the configuration of the EDA of the limited-area model of Météo-France, AROME.

On the one hand, these experiments on the QG system have shown that the block Krylov methods allow us to notably reduce the number of iterations needed to reach convergence (from 40 with one member to 14 with 10 members and 6 with 40 members). But on the other hand, block Krylov methods applied in primal space do not appear really convincing in terms of time consumption: the wall-clock times are greater or similar (depending on the parallelization strategy applied) to the ones obtained with independent non-block classical Krylov approaches. Nevertheless, performing block Krylov methods in dual space gives much more interesting results and considerably speeds up the minimization, so that the time needed to perform a single iteration is almost independent of the number of members in the ensemble. This is because we gain in the inner products and in the cost of communications. This approach allows us to gain in wall-clock time almost what is gained in iterations, reducing the time consumption from 17 *s* with one member down to 6.3 *s* with 10 members and 3.1 *s* with 40 members.

Moreover, because the dimension of the observation vectors is similar to the one in AROME, we can expect close results in dual space with the real system, even if these results need to be confirmed because of the existing differences in the Hessian eigenvalue spectra. We may need to further investigate the consequences of using a linear approximation in the EDA, notably the implication of the observation

operator linearization around the background ensemble mean. Also, we note that the FOM algorithm may handle inexact matrix–vector products (Gratton *et al.*, 2011). Therefore, we expect our block FOM algorithm also to perform with limited (single) precision, which may further reduce the computational cost. So exploring these block Krylov methods in a real NWP context is now the purpose of our ongoing work and already gives encouraging results which will be reported in the future.

ACKNOWLEDGEMENTS

This research has been supported by the AVENUE project funded by the RTRA-STAE foundation, Toulouse, France. The authors would also like to thank Etienne Arbogast for his help with OOPS.

ORCID

François Mercier https://orcid.org/0000-0001-7950-8314
Thibaut Montmerle https://orcid.org/0000-0003-3999-1633

REFERENCES

Andersson, E. and Jarvinen, H. (1999) Variational quality control. *Quarterly Journal of the Royal Meteorological Society*, 125, 697–722.

Andersson, E., Bauer, P., Beljaars, A., Chevallier, F., Hólm, E.V., Janisková, M., Källberg, P., Kelly, G., Lopez, P., McNally, A.P., Moreau, E., Simmons, A.J., Thépaut, J.-N. and Tompkins, A.M. (2005) Assimilation and modeling of the atmospheric hydrological cycle in the ECMWF forecasting system. *Bulletin of the American Meteorological Society*, 86(3), 387–402.

Arbogast, É., Desroziers, G. and Berre, L. (2017) A parallel implementation of a 4DEnVar ensemble. *Quarterly Journal of the Royal Meteorological Society*, 143, 2073–2083.

Auligné, T., McNally, A.P. and Dee, D.P. (2007) Adaptive bias correction for satellite data in a numerical weather prediction system. *Quarterly Journal of the Royal Meteorological Society*, 133, 631–642.

Bannister, R.N. (2008) A review of forecast-error covariance statistics in atmospheric variational data assimilation. II: modelling the forecast-error covariance statistics. *Quarterly Journal of the Royal Meteorological Society*, 134, 1971–1996.

Barker, D., Huang, W., Guo, Y.-R., Bourgeois, A. and Xiao, Q. (2004) A three-dimensional variational data assimilation system for MM5: implementation and initial results. *Monthly Weather Review*, 132, 897–914.

Belo-Pereira, M. and Berre, L. (2006) The use of an ensemble approach to study the background-error covariances in a global NWP. *Monthly Weather Review*, 134, 2466–2489.

Berre, L. and Desroziers, G. (2010) Filtering of background-error variances and correlations by local spatial averaging: a review. *Monthly Weather Review*, 138(10), 3693–3720.

Berre, L., Ecaterina, S. and Belo-Pereira, M. (2006) The representation of the analysis effect in three error simulation techniques. *Tellus A*, 58, 196–209.

Bonavita, M., Trémolet, Y., Hólm, E.V., Lang, S., Chrust, M., Janisková, M., Lopez, P., Laloyaux, P., de Rosnay, P., Fisher, M., Hamrud, M. and English, S. (2017) A strategy for data assimilation. Technical Memorandum 800. ECMWF, Reading, UK.

Bowler, N.E., Clayton, A.M., Jardak, M., Jermey, P.M., Lorenc, A.C., Wlasak, M.A., Barker, D.M., Inverarity, G.W. and Swinbank, R. (2017) The effect of improved ensemble covariances on hybrid variational data assimilation. *Quarterly Journal of the Royal Meteorological Society*, 143, 785–797.

Buehner, M. (2005) Ensemble-derived stationary and flow-dependent background-error covariances: evaluation in a quasi-operational NWP setting. *Quarterly Journal of the Royal Meteorological Society*, 131, 1013–1043.

Buehner, M. and Charron, M. (2007) Spectral and spatial localization of background-error correlations for data assimilation. *Quarterly Journal of the Royal Meteorological Society*, 133, 615–630.

Buehner, M., Morneau, J. and Charette, C. (2013) Four-dimensional ensemble-variational data assimilation for global deterministic weather prediction. *Nonlinear Processes in Geophysics*, 20(5), 669–682.

Cohn, S.E., da Silva, A., Guo, J., Sienkiewicz, M. and Lamich, D. (1998) Assessing the effects of data selection with the DAO physical-space statistical analysis system. *Monthly Weather Review*, 126(11), 2913–2926.

Courtier, P. (1997) Dual formulation of four-dimensional variational assimilation. *Quarterly Journal of the Royal Meteorological Society*, 123, 2449–2461.

Courtier, P., Thépaud, J. and Hollingsworth, A. (1994) A strategy of operational implementation of 4D-Var using an incremental approach. *Quarterly Journal of the Royal Meteorological Society*, 120, 1367–1388.

Dee, D.P. (1995) On-line estimation of error covariance parameters for atmospheric data assimilation. *Monthly Weather Review*, 123, 1128–1145.

Derber, J.C. and Rosati, A. (1989) A global oceanic data assimilation system. *Journal of Physical Oceanography*, 19, 1333–1347.

Derber, J.C. and Wu, W.-S. (1998) The use of TOVS cloud-cleared radiances in the NCEP SSI analysis system. *Monthly Weather Review*, 126(8), 2287–2299.

Desroziers, G. and Berre, L. (2012) Accelerating and parallelizing minimizations in ensemble and deterministic variational assimilations. *Quarterly Journal of the Royal Meteorological Society*, 138, 1599–1610.

Desroziers, G., Camino, J.-T. and Berre, L. (2014) 4DEnVar: link with 4D state formulation of variational assimilation and different possible implementations. *Quarterly Journal of the Royal Meteorological Society*, 140, 2097–2110.

El Akkraoui, A., Gauthier, P., Pellerin, S. and Buis, S. (2008) Intercomparison of the primal and dual formulations of variational data assimilation. *Quarterly Journal of the Royal Meteorological Society*, 134, 1015–1025.

Evensen, G. (1994) Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research: Oceans*, 99(C5), 10143–10162.

Fandry, C.B. and Leslie, L.M. (1984) A two-layer quasi-geostrophic model of summer trough formation in the Australian subtropical easterlies. *Journal of the Atmospheric Sciences*, 41(5), 807–818.

Fisher, M. (2003) Background-error covariance modelling. In *Seminar on Recent Development in Data Assimilation for Atmosphere and Ocean,* 8–12 September 2003, ECMWF, Reading, UK.

Fisher, M. and Courtier, P. (1995) Estimating the covariance matrices of analysis and forecast error in variational data assimilation. Technical Memorandum 200. ECMWF, Reading, UK.

Fisher, M., Trémolet, Y., Auvinen, H., Tan, D.G.H. and Poli, P. (2011) Weak-constraint and long-window 4D-Var. Technical Memorandum 655. ECMWF, Reading, UK.

Gilbert, J.-C. and Lemaréchal, C. (2006) *The Module M1QN3–Version 3.1*. Rocquencourt: INRIA.

Gratton, S. and Tshimanga, J. (2009) An observation-space formulation of variational assimilation using a restricted preconditioned conjugate gradient algorithm. *Quarterly Journal of the Royal Meteorological Society*, 135, 1573–1585.

Gratton, S., Lawless, A.S. and Nichols, N.K. (2007) Approximate Gauss–Newton methods for nonlinear least squares problems. *SIAM Journal on Optimization*, 18(1), 106–132.

Gratton, S., Toint, P.L. and Tshimanga Ilunga, J. (2011) Range-space variants and inexact matrix-vector products in Krylov solvers for linear systems arising from inverse problems. *SIAM Journal on Matrix Analysis and Applications*, 32(3), 969–986.

Gratton, S., Gürol, S., Toint, P.L., Tshimanga, J. and Weaver, A.T. (2012) Krylov methods in the observation space for data assimilation. In *Workshop: Mathematical and Algorithmic Aspects of Atmosphere-Ocean Data Assimilation*, Report No. 58, Oberwolfach, Germany

Gratton, S., Gürol, S. and Toint, P.L. (2013) Preconditioning and globalizing conjugate gradients in dual space for quadratically penalized nonlinear-least squares problems. *Computational Optimization and Applications*, 54(1), 1–25.

Gürol, S., Weaver, A.T., Moore, A.M., Piacentini, A., Arango, H.G. and Gratton, S. (2014) B-preconditioned minimization algorithms for variational data assimilation with the dual formulation. *Quarterly Journal of the Royal Meteorological Society*, 140, 539–556.

Gutknecht, M.H. (2006) Block Krylov space methods for linear systems with multiple right-hand sides: an introduction. http://www.sam.math.ethz.ch/~mhg/pub/delhipap.pdf; accessed 14 September 2018

Haben, S.A., Lawless, A.S. and Nichols, N.K. (2011) Conditioning of incremental variational data assimilation, with application to the Met Office system. *Tellus A*, 63(4), 782–792.

Hamill, T.M. and Snyder, C. (2000) A hybrid ensemble Kalman filter–3D variational analysis scheme. *Monthly Weather Review*, 128(8), 2905–2919.

Hestenes, M.R. and Stiefel, E. (1952) Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49(6), 409–436.

Houtekamer, P.L. and Mitchell, H.L. (2001) A sequential ensemble Kalman filter for atmospheric data assimilation. *Monthly Weather Review*, 129(1), 123–137.

Hunt, B.R., Kostelich, E.J. and Szunyogh, I. (2007) Efficient data assimilation for spatiotemporal chaos: a local ensemble transform Kalman filter. *Physica D: Nonlinear Phenomena*, 230(1–2), 112–126.

Ide, K., Courtier, P., Ghil, M. and Lorenc, A.C. (1997) Unified notation for data assimilation: operational, sequential and variational. *Journal of the Meteorological Society of Japan Series II*, 75(1B), 181–189.

Ingleby, N.B., Lorenc, A.C., Ngan, K., Rawlins, F. and Jackson, D.R. (2013) Improved variational analyses using a nonlinear humidity control variable. *Quarterly Journal of the Royal Meteorological Society*, 139, 1875–1887.

Kucukkaraca, E. and Fisher, M. (2006) Use of analysis ensembles in estimating flow-dependent background error variances. Technical Memorandum 492. ECMWF, Reading, UK.

Le Dimet, F. and Talagrand, O. (1986) Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects. *Tellus*, 38A, 97–110.

Liu, C., Xiao, Q. and Wang, B. (2008) An ensemble-based four-dimensional variational data assimilation scheme. Part I: technical formulation and preliminary test. *Monthly Weather Review*, 136(9), 3363–3373.

Lorenc, A.C. (2003) The potential of the ensemble Kalman filter for NWP: a comparison with 4D-Var. *Quarterly Journal of the Royal Meteorological Society*, 129, 3183–3203.

Lorenc, A.C., Ballard, S.P., Bell, R.S., Ingleby, N.B., Andrews, P.L.F., Barker, D.M., Bray, J.R., Clayton, A.M., Dalby, T., Li, D., Payne, T.J. and Saunders, F.W. (2000) The Met Office global three-dimensional variational data assimilation scheme. *Quarterly Journal of the Royal Meteorological Society*, 126, 2991–3012.

Lorenc, A.C., Bowler, N.E., Clayton, A.M., Pring, S.R. and Fairbairn, D. (2015) Comparison of hybrid-4DEnVar and hybrid-4DVar data assimilation methods for global NWP. *Monthly Weather Review*, 143(1), 212–229.

Lorenc, A.C., Jardak, M., Payne, T., Bowler, N.E. and Wlasak, M.A. (2017) Computing an ensemble of variational data assimilations using its mean and perturbations. *Quarterly Journal of the Royal Meteorological Society*, 143, 798–805.

Lowery, B.R. and Langou, J. (2014) Stability analysis of QR factorization in an oblique inner product. arXiv preprint arXiv:1401.5171

Ménétrier, B., Montmerle, T., Michel, Y. and Berre, L. (2015) Linear filtering of sample covariances for ensemble-based data assimilation. Part I: optimality criteria and application to variance filtering and covariance localization. *Monthly Weather Review*, 143(5), 1622–1643.

Nocedal, J. and Wright, S.J. (2006) *Numerical Optimization*, 2nd edition. New York, NY: Springer.

O'Leary, D.P. (1980) The block conjugate gradient algorithm and related methods. *Linear Algebra and its Applications*, 29, 293–322.

Plu, M. and Arbogast, P. (2005) A cyclogenesis evolving into two distinct scenarios and its implications for short-term ensemble forecasting. *Monthly Weather Review*, 133(7), 2016–2029.

Rabier, F. (2005) Overview of global data assimilation developments in numerical weather-prediction centres. *Quarterly Journal of the Royal Meteorological Society*, 131, 3215–3233.

Raynaud, L., Berre, L. and Desroziers, G. (2012) Accounting for model error in the Météo-France ensemble data assimilation system. *Quarterly Journal of the Royal Meteorological Society*, 138, 249–262.

Saad, Y. (2003) *Iterative Methods for Sparse Linear Systems*. Philadephia, PA: SIAM.

Seity, Y., Brousseau, P., Malardel, S., Hello, G., Bénard, P., Bouttier, F., Lac, C. and Masson, V. (2011) The AROME-France convective-scale operational model. *Monthly Weather Review*, 139(3), 976–991.

Snyder, C., Hamill, T.M. and Trier, S.B. (2003) Linear evolution of error covariances in a quasigeostrophic model. *Monthly Weather Review*, 131(1), 189–205.

Stathopoulos, A. and Wu, K. (2002) A block orthogonalization procedure with constant synchronization requirements. *SIAM Journal on Scientific Computing*, 23(6), 2165–2182.

Tavolato, C. and Isaksen, L. (2015) On the use of a Huber norm for observation quality control in the ECMWF 4D-Var. *Quarterly Journal of the Royal Meteorological Society*, 141, 1514–1527.

Trémolet, Y. (2007) Incremental 4D-Var convergence study. *Tellus A*, 59(5), 706–718.

Žagar, N., Andersson, E. and Fisher, M. (2005) Balanced tropical data assimilation based on a study of equatorial waves in ECMWF short-range forecast errors. *Quarterly Journal of the Royal Meteorological Society*, 131, 987–1011.

How to cite this article: Mercier F, Gürol S, Jolivet P, Michel Y, Montmerle T. Block Krylov methods for accelerating ensembles of variational data assimilations. *Q J R Meteorol Soc.* 2018;144:2463–2480. https://doi.org/10.1002/qj.3329

APPENDIX

MODIFIED GRAM–SCHMIDT QR ALGORITHM

This appendix provides the details of the Modified Gram–Schmidt QR algorithm (MGSQR) with respect to the $\mathbf{B}$ inner product used for performing the QR decomposition implied by block B-FOM Algorithm 1 and block RB-FOM Algorithm 3. MGSQR is given in Algorithm 4. Its inputs are two matrices $\mathbf{w}$ and $\mathbf{z}$, both in $\mathbb{R}^{N\times m}$, with $\mathbf{z} = \mathbf{B}\mathbf{w}$. The outputs are matrices $\mathbf{v}$ and $\mathbf{z}$, again in $\mathbb{R}^{N\times m}$, and $\boldsymbol{\beta}$ in $\mathbb{R}^{m\times m}$, which have to verify $\mathbf{z} = \mathbf{B}\mathbf{v}$, $\mathbf{w} = \mathbf{v}\boldsymbol{\beta}$ (QR factorization), $\boldsymbol{\beta}$ upper triangular and $\mathbf{v}^{\mathrm{T}}\mathbf{z} = \mathbf{Id}_m$ (orthonormality with respect to the $\mathbf{B}$ inner product). In Algorithm 4, w_p denotes the pth column of matrix $\mathbf{w}$ (same for all matrices of the same size).

Algorithm 4 MGSQR

$\beta_{0,0} = \sqrt{(w_0, z_0)}$
$v_0 = w_0/\beta_{0,0}$
$z_0 := z_0/\beta_{0,0}$
for $k = 1 : m-1$ **do**
  $v_k = w_k$
  for $p = 0 : k-1$ **do**
    $\beta_{p,k} = \sqrt{(v_k, z_p)}$
    $z_k := z_k - \beta_{p,k} z_p$
    $v_k := v_k - \beta_{p,k} v_p$
  end for
  $\beta_{k,k} = \sqrt{(v_k, z_k)}$
  $v_k := v_k/\beta_{k,k}$
  $z_k := z_k/\beta_{k,k}$
end for