Dear Anonymous Referee #1,

On behalf of all co-authors, first of all I would like to thank you for giving us very useful comments and suggestions. We really appreciate your detailed comments and suggestions, which certainly improve the quality of the paper. In the revision, we have carefully considered all the suggested changes and included them to the best of our ability.

In the following we will give our response to every comment. To make the changes easier to identify, we have numbered them.

Best regards,
Guangliang Fu
on behalf all co-authors

(The revised manuscript is in the latter part of this pdf.)

Reply to Specific comments:

1. The variable notation $N_{ash}$ should be changed to $n_{ash}$ as the small $n$ presents number of grid points (line 4.30).

Response: I agree. All $N_{ash}$ have been changed to $n_{ash}$ in the revision.

2. Equation (15) indicates whether we would have speedup by the MS method (line 4.5). Could it be extended to explicitly specify the expected amount of speed up (with some assumptions if necessary)?

Response: As suggested, it is extended in line(s) P9L29–P9L33 (P:page, L:lines):

The relation $n_{ash} < \frac{N - 1}{N} n$ indicates whether we would have speedup by the MS method, actually it can be extended to Eq. (15),

$$S_{ms} = \frac{O(nN^2)}{O(nN + n_{ash}N^2)} = O\left(\frac{n}{n_{ash}}\right),$$

which explicitly specifies the expected amount of speedup ($S_{ms}$) of $A^a = A^f X$ by the MS algorithm. In this case study, $N$ is taken at 100 and $\frac{n_{ash}}{n} \approx \frac{1}{3}$, so $S_{ms}$ is approximately 3.0.

3. Some explanations are necessary about the involved operations in the others part for the clarification and justification of the less (by 0.03h) computational time (Table-1).

Response: I agree. The explanations are added in line(s) P10L22–P10L26:

Note that it can also be observed that the computational time for the “other” parts in Table 1 (such as operations for setting environmental variables, starting and finalizing data assimilation algorithms, as mentioned in Section 3.1) is slightly reduced by the MS method (i.e., 0.03 h in this case). This is because in the conventional EnKF, the ensemble mean state $\bar{A}^f$ is calculated in the “other” parts as an output to finalize the data assimilation process.
algorithms, while in MS-EnKF, the calculations of $\tilde{A}^f$ are needed and directly involved in the “Analysis” part. ”

4. The computational overhead involved to transform the full matrix to a small one and vice versa should be explicitly discussed. The effect in the computation time should be also discussed.

Response: As suggested, the computational overhead involved to transform the full matrix to a small one and vice versa are explicitly discussed in line(s) P8L17–P8L19:

“ Construct masked ensemble state matrix $\tilde{A}^f$: Using the mask array $z_{n_{ash} \times 1}$ obtained from step (ii), $\tilde{A}^f_{n_{ash} \times N}$ can be constructed column by column according to Eq. (13), and the computational cost (overhead) for this step is $O(n_{ash} N)$.

\[
\tilde{A}^f(1 : n_{ash}, 1 : N) = A^f(z(1 : n_{ash}), 1 : N),
\]

(13)

and in line(s) P8L24–P8L27:

“ Construct analyzed ensemble state matrix $A^a$: With the computed $\tilde{A}^a$ from step (iv) and the mask array $z$ from step (ii), the final analyzed ensemble state matrix $A^a_{n \times N}$ can be constructed based on Eq. (14). The computational cost (overhead) for this step is $O(nN)$.

\[
A^a(z(1 : n_{ash}), 1 : N) = \tilde{A}^a(1 : n_{ash}, 1 : N),
\]

(14)

The effect in the computation time is discussed in line(s) P9L15–P9L20:

“ Now we analyze MS’s computational cost, which can be summed as $O(nN) + O(n) + O(n_{ash}N) + O(n_{ash}N^2) + O(nN)$, i.e., $O(nN + n_{ash} N^2)$. Thus, the computational overhead involved to transform the full matrix to a small one (i.e., $O(n_{ash}N)$ for procedure (iii)) has little effect in the total computation cost of MS (i.e., $O(nN + n_{ash} N^2)$). However, the computational overhead of transforming the small matrix to the full one (i.e., $O(nN)$ for procedure (v)) does contribute a part, which cannot be ignored, to the total MS’s computational cost.”

5. Line 3.20: It is mentioned that 72% of the total runtime is spent on the analysis step. This information can be further extended (preferably with mathematical formulation) to predict the maximum computational speedup (e.g. 3.57 for the case in this paper). c.f. Amdahl’s law (Amdahl, Gene M. (1967). “Validity of the Single Processor Approach to Achieving Large-Scale Computing Capabilities”. AFIPS Conference Proceedings (30): 483-485.)

6. Related to the above point, a further discussion about the discrepancy in the actual speedup (e.g. 2.24 for the case in this paper) from the predicted speedup could be included.

Response: I agree. An extended discussion is added in line(s) P10L1–P10L9:

“ According to Amdahl’s law (Amdahl, 1967), the total computational speedup ($S_{total}$
by MS can be predicted by Eq. (16),
\[ S_{\text{total}} = \frac{1}{(1 - p_{\text{ms}}) + \frac{p_{\text{ms}}}{S_{\text{ms}}}} \] (16)

where \( p_{\text{ms}} \) is the proportion of the computational cost of \( A^a = A^f X \) in the overall data assimilation computations. It has been evaluated that the computational cost of \( A^a = A^f X \) dominates the analysis step (see Fig. 2b), thus the proportion of the computational cost of \( A^a = A^f X \) approximates the proportion of the analysis step in the total data assimilation computations (i.e., \( p_{\text{ms}} \approx 72\% \) in this case, as described in Section 3.1). Therefore, based on Eq. (16), the maximum ("ideal") computational speedup can be predicted to be \( \frac{1}{1 - p_{\text{ms}}} \) (i.e., \( \approx 3.57 \) for this case study) when \( S_{\text{ms}} \) approximates infinity. However, this is not the actual speedup because \( S_{\text{ms}} \) is in fact specified by Eq. (15). (Based on discussions above, \( S_{\text{total}} \) can be therefore estimated by Eq. (15) at \( \approx 2.0 \) in this case.)

7. It is understood from equation (15) the total speedup is quite low, is there any statistical data regarding the value of \( n_{\text{ash}}/n \) to know the probability of the applicability of the proposed method and justify the weight of the work?

Response: Thanks for the suggestion. The related discussion is added in line(s) P12L1–P12L7:

" As stated in Eq. (15), the speedup of the MS method is approximately the inverse of \( n_{\text{ash}}/n \). So far there is no statistical data on the value of \( n_{\text{ash}}/n \). Consider the problem of volcanic ash transport, there is only one emission point (at the volcano), all the ashes in atmospheres are transported by the directional wind drive from the same source point. Thus volcanic ash cloud is actually transported in a shape of a plume, which in general doesn’t cover the full but only a small part of the 3D domain. At the start phase of a volcanic ash eruption, \( n_{\text{ash}}/n \) is much smaller than 1.0 (started from 0). During transport over a long time (one and a half months for this case study), \( n_{\text{ash}}/n \) increases to approximately \( \frac{1}{3} \). Therefore, the speedup of MS on volcanic ash data assimilation will be significant. ”

8. The paper investigates actually a single method (MS) to accelerate volcanic ash data assimilation. Other methods are only referred (e.g. parallelization of the analysis step) or contrasted (e.g. localization) and are not really investigated in this paper. So, the plural form (e.g. strategies) should be omitted.

Response: I agree. All “strategies” in the plural form have been changed to “a strategy” in the revision.

The revised manuscript starts from next page.
A mask-state algorithm to accelerate volcanic ash data assimilation: a case study with the LOTOS-EUROS model (version 1.10)

Guangliang Fu¹, Hai Xiang Lin¹, Arnold Heemink¹, Arjo Segers², Nils van Velzen¹³, Tongchao Lu⁴, Shiming Xu⁵, and Sha Lu¹

¹Delft University of Technology, Delft Institute of Applied Mathematics, Mekelweg 4, 2628 CD Delft, The Netherlands.
²TNO, Department of Climate, Air and Sustainability, P.O. Box 80015, 3508 TA Utrecht, The Netherlands.
³VORtech, P.O. Box 260, 2600 AG Delft, The Netherlands.
⁴School of Mathematics, Shandong University, Jinan, Shandong 250100, China.
⁵Ministry of Education Key Laboratory for Earth System Modeling, Center for Earth System Science (CESS), Tsinghua University, Beijing, China

Correspondence to: Guangliang Fu (G.Fu@tudelft.nl)

Abstract. In this study, we investigate a strategy to accelerate the data assimilation algorithm. Based on evaluations of the computational time, the analysis step of the assimilation turns out to be the most expensive part. After a study on the characteristics of the ensemble ash state, we propose a mask-state algorithm which records the sparsity information of the full ensemble state matrix and transforms the full matrix into a relatively small one. This will reduce the computational cost in the analysis step. Experimental results show the mask-state algorithm significantly speeds up the analysis step. Subsequently, the total amount of computing time for volcanic ash data assimilation is reduced to an acceptable level. The mask-state algorithm is generic and thus can be embedded in any ensemble-based data assimilation framework. Moreover, ensemble-based data assimilation with the mask-state algorithm is promising and flexible, because it implements exactly the standard data assimilation without any approximation and it realizes the satisfying performance without any change of the full model.

1 Introduction

Volcanic ash erupted into atmospheres can lead to severe influences on aviation society (Gudmundsson et al., 2012). Turbine engines of airplanes are extremely threatened by the ash’s ingestion (Casadevall, 1994). Thus, accurate real-time aviation advices are highly required during an explosive volcanic ash eruption (Eliasson et al., 2011). Recently, ensemble-based data assimilation (Evensen, 2003) has been evaluated very useful to improve volcanic ash forecasts and regional aviation advices (Fu et al., 2016). It corrects volcanic ash concentrations by continuously assimilating observations. In (Fu et al., 2016), real aircraft in situ measurements were assimilated using the ensemble Kalman filter (EnKF), which is the most known and popular ensemble-based assimilation method. Based on the validation with independent data, ensemble-based data assimilation was concluded powerful for improving the forecast accuracy.

However, to make the methodology efficient also in an operational (real-time) sense, the computational efforts must be acceptable. For volcanic ash assimilation problems, so far, no studies on the computational aspects have been reported in the literature. Actually, when large amounts of volcanic ash erupted into atmospheres, the computational speed of volcanic ash...
forecasts is just as important as the forecast accuracy (Zehner, 2010). For example, due to the lack of a fast and accurate forecast system, the sudden eruption of the Eyjafjallajökull volcano in Iceland from 14 April to 23 May 2010, had caused an unprecedented closure of the European and North Atlantic airspace resulting in a huge global economic loss of 5 billion US dollars (Oxford-Economics, 2010). Since then, research on fast and accurate volcanic ash forecasts have gained much attention, because it is needed to provide timely and accurate aviation advices for frequently operated commercial airplanes. It was shown the accuracy of volcanic ash transport can be significantly improved by the assimilation system in (Fu et al., 2016). Therefore, it is urgent to also consider the computational aspect, i.e., improving the computational speed of the volcanic ash assimilation system as fast as possible. This is the main focus of this study.

Due to the computational complexity of ensemble-based algorithms and the large scale of dynamical applications, applying these methods usually introduces a large computational cost. This has been reported from literature on different applications. For example, for operational weather forecasting with ensemble-based data assimilation, Houtekamer et al. (2014) reported computational challenges at Canadian Meteorological Center with an operational EnKF featuring 192 ensemble members, using a large $600 \times 300$ global horizontal grid and 74 vertical levels. That an initialization requirement of over $7 \times 10^{10}$ values to specify each ensemble, results in large computational efforts on the initialization and forecast steps in weather forecasting.

For oil reservoir history-matching (Tavakoli et al., 2013), the reservoir simulation model usually has a large number of state variables, thus the forecasts of an ensemble of simulation models are often time-consuming. Besides, when time-lapse seismic or dense reservoir data is available, the analysis step of assimilating these large observations becomes very time-consuming (Khairullah et al., 2013). Large computational requirements of ensemble-based data assimilation have also been reported in ocean circulation models (Keppenne, 2000; Keppenne and Rienecker, 2002), tropospheric chemistry assimilation (Miyazaki et al., 2015), and many other applications.

To accelerate an ensemble-based data assimilation system, the ensemble forecast step can be first parallelized because the propagation of different ensemble members is independent. Thus if a computer with a sufficiently large number of parallel processors is available, all the ensemble members can be simultaneously integrated. In the analysis stage, to calculate the Kalman gain and the ensemble error covariance matrix, all ensemble states must be combined together. In weather forecasting and oceanography sciences, Keppenne (2000); Keppenne and Rienecker (2002); Houtekamer and Mitchell (2001) have reported using parallelization approaches to accelerate the expensive analysis stage. In reservoir history matching, a three-level parallelization has been proposed by Tavakoli et al. (2013); Khairullah et al. (2013) in recent years, to significantly reduce computational efforts of both forecast and analysis steps due to massive dense observations and large simulation models. The first parallelization level is to separately perform the ensemble simulations on different processors during the forecast step.

This approach is usually quite efficient when a large ensemble size is used. However, the scale or model size of one reservoir simulation is constrained by the memory of a single processor. Thus, the second parallelization level is to perform one ensemble member simulation using a parallel reservoir model. These two levels do not deal with the analysis step, which collects all ensemble members to do computations usually on a single processor. Therefore, a third level of parallelization was implemented by Tavakoli et al. (2013); Khairullah et al. (2013) through parallelizing matrix-vector multiplications in the analysis steps. Furthermore, some other approaches on accelerating ensemble-based assimilation systems, have also been reported, such
as GPU-based acceleration (Quinn and Abarbanel, 2011) in Numerical Weather Prediction (NWP), domain decomposition in atmospheric chemistry assimilation (Segers, 2002; Miyazaki et al., 2015). The observations used in an assimilation system can be also optimized with some preprocessing procedures, as reported by Houtekamer et al. (2014).

Although for other applications, there were many efforts in dealing with large computational requirements in an ensemble-based data assimilation system, most of them cannot be directly used to accelerate volcanic ash data assimilation. This is because the acceleration algorithms are strongly dependent on specific problems, such as model complexity (high or low resolution), observation type (dense or sparse), primary requirement (accuracy or speed). These factors determine, for a specific application, which part is the most time-consuming, and which part is intrinsically sequential. Thus, no unified approach for efficient acceleration of all the applications can be found. Although the successful approaches in other applications cannot be directly employed in volcanic ash forecasts, their success do stress the importance of designing a proper approach based on the computational analysis of a specific assimilation system. Therefore, the computational cost of our volcanic ash assimilation system will be first analyzed. Then, based on the computational analysis, we will investigate a strategy to accelerate the ensemble-based data assimilation system for volcanic ash forecasts.

This paper is organized as follows. Section 2 introduces the methodology of volcanic ash data assimilation. Section 3 analyzes the computational cost of the conventional volcanic ash data assimilation system. In Section 4, the mask-state algorithm is developed for acceleration. The discussions on the mask-state algorithm is in Section 5. Finally, the last section summarizes the concluding remarks of our research.

2 Methodology of the volcanic ash data assimilation system

In this study, the ensemble Kalman filter (EnKF) (Evensen, 1994) is employed to perform ensemble-based data assimilation. EnKF is typically a sequential Monte Carlo method (Evensen, 2003), according to the uncertain state estimate with \( N \) ensemble members, \( \xi_1, \xi_2, \ldots, \xi_N \). Each member is assumed as one sample in the distribution of the true state. It has been proposed that for operational applications, the ensemble size can be limited to 10 – 100 for cost effectiveness (Nerger and Hiller, 2013; Barbu et al., 2009). Thus, in this study, an ensemble size of 100 is used due to the high-accuracy requirement of the volcanic ash forecasts to aviation advices as mentioned in Section 1.

To simulate a volcanic ash plume, an atmospheric transport model is needed. In this paper, the LOTOS-EUROS (abbreviation of LOng Term Ozone Simulation – EURopean Operational Smog) model is used (Schaap et al., 2008) with model version 1.10 (http://www.lotos-euros.nl/). The LOTOS-EUROS model (Schaap et al., 2008) is an operational model focusing on nitrogen oxides, ozone, particular matter, volcanic ash. The model configurations for volcanic ash were discussed in details by Fu et al. (2016). For volcanic ash simulation, the model is configured with a state vector of size 180\( \times \)200\( \times \)18\( \times \)6 (the dimensions correspond to longitude, latitude, vertical level, ash species), the size of model state is thus calculated as \( \sim 10^6 \).

The experiment in this study starts at \( t_0 \) (09:00 UTC, 18 May, 2010 for this study) by considering an initial condition from previous LOTOS-EUROS conventional model run (see Fig. 1a). In the second step (the forecast step) the model propagates the
ensemble members from the time $t_{k-1}$ to $t_k$ ($k > 0$, the time step is 10 minutes):

$$\xi_j^f(k) = M_{k-1}(\xi_j^a(k-1)). \quad (1)$$

The operator $M_{k-1}$ describes the time evolution of the state which contains the ash concentrations in all model grid-boxes. The state at the time $t_k$ has a distribution with the mean $\bar{x}^f$ and the forecast error covariance matrix $P^f$ given by:

$$\bar{x}^f(k) = \frac{\sum_{j=1}^{N} \xi_j^f(k)}{N}, \quad (2)$$

$$L^f(k) = [\xi_1^f(k) - \bar{x}^f(k), \ldots, \xi_q^f(k) - \bar{x}^f(k)], \quad (3)$$

$$P^f(k) = [L^f(k)L^f(k)^T]/(N-1), \quad (4)$$

where $L^f$ represents the ensemble perturbation matrix. In this study, the forecast step is performed in parallel because of the natural/common parallelism of the independent ensemble propagation, which is a trivial approach when employing ensemble-based data assimilation (Liang et al., 2009; Tavakoli et al., 2013; Khairullah et al., 2013).

When the model propagates to 09:40 UTC, 18 May, 2010, the volcanic ash state gets sequentially analyzed by the data assimilation process through combining real aircraft in situ measurements of PM$_{10}$ and PM$_{2.5}$ concentrations until 11:10 UTC. The measurement route and values are demonstrated in Fig. 1b–1c and the details are described in (Weber et al., 2012; Fu et al., 2016). The observational network at the time $t_k$ is defined by the operator $H_k$ which maps the state vector $x$ to the observational space $y$ by

$$y(k) = H_k(x(k)) + v(k), \quad (5)$$

where $y$ contains the aircraft measurements and $v$ represents the observational error. $H_k$ selects the grid cell in $x(k)$ that corresponds to the locations of the observation. When measurements are available, the ensemble members are updated in the analysis step using

$$K(k) = P^f(k)H(k)^T[H(k)P^f(k)H(k)^T + R]^{-1}, \quad (6)$$

$$\xi_j^a(k) = \xi_j^f(k) + K(k)[y(k) - H(k)\bar{x}_j^f(k) + v_j(k)], \quad (7)$$

where $K$ represents the Kalman gain, $R$ represents the measurement error covariance matrix and $v_j$ represents the realization out of the observation error distribution $v$. After the continuous assimilation ending at 11:10 UTC, the forecast at 12:00 UTC is illustrated in Fig. 1d, for which the forecast accuracy has been carefully evaluated as significantly improved compared to the case without assimilation (Fu et al., 2016).

The EnKF with above setups is abbreviated as “conventional EnKF” and used in this study for the computational evaluation. Note that in the study we don’t use covariance localization as proposed by Hamill et al. (2001) for reducing spurious covariances. This is because although localization is possible, the ideal case is not to use it in order to have the correct covariances in a large (converged) ensemble. It is crucial for localization that when unphysical (spurious) covariances are eliminated, physical (correct) covariances can be well maintained (Petrie and Dance, 2010). If the “filtering length scale” for localization is too
long (i.e., all the dynamical covariances are allowed), many of the spurious covariances may not be eliminated. If the length is too short, important physical dynamical covariances then may be lost together with the spurious ones. Therefore, essentially deciding an accurate localization is a challenging subject (Riishojgaard, 1998; Kalnay et al., 2012) especially for accuracy-demanding applications. Therefore, in this study we choose the ensemble size of 100 to guarantee the accuracy and avoid large spurious covariances.

3 Computational analysis for volcanic ash data assimilation

3.1 Computational analysis of the total runtime

Ensemble-based data assimilation is a useful approach to improve the forecast accuracy of volcanic ash transport. However, if it is time-consuming, it cannot be taken as efficient due to the high requirement on speed for volcanic ash assimilation (see Section 1). Based on this consideration, we need to analyze the computational cost of a conventional volcanic ash assimilation system.

As introduced in Section 2, the total execution time of conventional EnKF comprises four parts, i.e., initialization, forecast, analysis and other computational cost. The initialization time includes reading meteorological data, initializing model geographical and grid configurations, reading emission information, initializing stochastic observer for reading and transforming observations to the model grid, initializing all the ensemble states and ensemble mean, and so on. The forecast time is obtained from Eq. (1), while the analysis time corresponds to the computational sum from Eq. (2) to (7). The other computational time includes script compiling, setting environment variables, starting and finalizing data assimilation algorithms, etc.

The evaluation result of the conventional EnKF is shown in Table 1 (the middle column). It can be seen that the total computational time (4.36 h) is relatively large compared to the simulation window (3.0 h, i.e., from 9:00–12:00 UTC, 18 May, 2010), which is too much in an operational sense. Therefore, in this study, we aim to accelerate the computation to within an acceptable runtime (i.e., requires less runtime than the time period of the data assimilation application).

It can be also observed from Table 1 that the main contribution to the total execution time is the analysis step. Compared to the initialization and forecast time, the analysis stage takes 72% of the total runtime. Due to the expensive analysis step, although some approaches (such as MPI-parallel I/O (Filgueira et al., 2014), domain decomposition (Segers, 2002)) can potentially accelerate the initialization and forecast step, the effect to the final acceleration of the total computational cost is little. Therefore, to get acceptable computational time, the cost reduction in the analysis step is the target. One may wonder that since the amount of observations is small, why does analysis takes so much time? The large state vector seems to be left responsible for the problem. To know the exact reason, the detailed computational cost of the analysis step must be evaluated.
3.2 Cost estimation of all analysis procedures

We start with the formulations of the analysis step. The analysis step is represented by Eq. (7), which can be written in a full matrix format with Eq. (8),

$$A_{n×N}^a = A_{n×N}^f + K_{n×m}(Y_{m×N} - H_{m×n}A_{n×N}^f), \tag{8}$$

where the subscripts represent the matrix’s dimensions. $A^f$ and $A^a$ represent the forecasted and analyzed ensemble state matrix, and are respectively built up from $\xi^f$ and $\xi^a$ with $N$ ensembles. The measurement ensemble matrix $Y$ is formed by an ensemble of $y + v$ (see Eq. (7)). $H$ is the observational matrix, which is used to select state variables (at measurement locations) in the full ensemble state matrix corresponding to the measurement ensemble matrix $Y$. $n$ is the number of model state variables in a three-dimensional (3D) domain, i.e., $\sim 10^6$ in this study (see Section 2). $m$ is the amount of measurements at one assimilation time, which depends on the measurement type. For aircraft in situ measurements used in this study (see Fig. 1c), two measurements are made at each time by one research flight, so that $m$ is 2 here. $N$ is the ensemble size and is taken as 100 in this study. As described in Eq. (3), the ensemble perturbation matrix $L^f$ in EnKF can be re-written as

$$L_{n×N}^f = A_{n×N}^f - \bar{A}_{n×N}^f = A_{n×N}^f(I_{N×N} - \frac{1}{N}1_{N×N}) = A_{n×N}^fB_{N×N}, \tag{9}$$

where $I$ is an $N×N$ unit matrix and $1$ is an $N×N$ matrix with all elements equal to 1. Thus, $L^f = A^fB$ where $B_{N×N}$ is introduced to represent $(I_{N×N} - \frac{1}{N}1_{N×N})$. So that, $HL^f = O^fB$, where $O_{m×N}^f$ is used to represent $(HA^f)$. Here we explicitly express $L^f$ and $HL^f$ in the form of $A^f$ and $O^f$, respectively. This is because in our volcanic ash assimilation system, $A^f$ and $O^f$ are two of the three inputs (another one is the measurement ensemble matrix $Y$) for the analysis step. These are the three inputs used for actual computations in the analysis step. As shown in Fig. 2a, $A^f$ is obtained from the forecast step, $O^f$ and $Y$ are acquired from our stochastic observer module (see Fig. 2a) which is used for a volcanic ash transport model to integrate geophysical measurements. With the input $Y$, the measurement error covariance $R$, as introduced in Eq. (6), can be then computed with

$$R_{m×m} = \frac{1}{N-1} (Y_{m×N} - \bar{Y}_{m×N})(Y_{m×N} - \bar{Y}_{m×N})' = \frac{1}{N-1} (YB)(YB)'. \tag{10}$$

Based on previous definitions and Eq. (2) to (7), the analysis step can be reformulated as followings,

$$A_{n×N}^a = A^f + K(Y - HA^f)$$

$$= A^f + P^fH'(HP^fH' + R)^{-1}(Y - HA^f)$$

$$= A^f + \frac{1}{N-1} L^f(HL^f)'\left[\frac{1}{N-1}(HL^f)(HL^f)'+ \frac{1}{N-1}(YB)(YB)']^{-1}(Y - HA^f) \right.$$  

$$= A^f + A^fB(O^fB)'[(O^fB)(O^fB)'+ (YB)(YB)']^{-1}(Y - O^f)$$

$$= A^f[I + B(O^fB)'[(O^fB)(O^fB)'+ (YB)(YB)']^{-1}(Y - O^f)]$$

$$= A_{n×N}^fX_{N×N}, \tag{11}$$
where
\[ X_{N \times N} = \{ I + B(O^TB)'[(O^TB)(O^TB)'+(YB)(YB)']^{-1}(Y-Of) \}. \] (12)

Eq. (11) shows how the analysis step is performed in a volcanic ash assimilation system. In order to accelerate the analysis step, the most time-consuming part must be reduced. Fig. 2b shows estimations of the computational cost for each procedure in the analysis step. Considering that the state number \( n \) (\( \sim 10^6 \)) is significantly larger than the measurement number \( m \) (\( m=2 \) here) and the ensemble size \( N \) (\( N=100 \)), thus the most time-consuming procedure in the analysis step is the last one, that is \( A^a = A^fX \) with computational cost of \( O(nN^2) \). Therefore, in our volcanic ash assimilation system, this part is the most time-consuming part in the analysis step. Note that the procedure \( [(O^TB)(O^TB)'+(YB)(YB)']^{-1} \) for Singular Value Decomposition (SVD) in our study is not time-consuming, which is quite different from some other applications, such as reservoir history matching (Tavakoli et al., 2013; Khairullah et al., 2013). This is because the SVD procedure costs \( O(m^3) \), and due to the measurement size in the order of the size of the state in those cases, SVD procedure thus requires a huge computational cost for reservoir assimilation.

4 The mask-state algorithm for acceleration of the analysis step

4.1 Characteristic of ensemble state matrix \( A^f \)

Analysis in the previous section shows that \( A^a = A^fX \) is most expensive in the analysis step. Each column of \( A^f \) is constructed from a forecasted ensemble state, thus the dimension of \( A^f \) is \( n \times N \). In each column, the element values correspond to volcanic ash concentrations in a 3D domain. Fig. 3 shows the coverage of all ensemble forecast states at a selected time 10:00 UTC 18 May, 2010, without loss of generality. A common phenomenon can be observed, that is only a part of the 3D domain are filled with volcanic ash. The ash clouds only concentrate in a plume which is transported over time. This is because volcanic eruption is a fast and strong process. The advection dominates the transport, and the volcanic ash plume is transported with the wind. This is a particular characteristic for volcanic ash transport, in contrast to other atmospheric related applications such as ozone (Curier et al., 2012), \( \text{SO}_2 \) (Barbu et al., 2009), \( \text{CO}_2 \) (Chatterjee et al., 2012). For those applications, the concentrations are everywhere in the domain, the emission sources are also everywhere, and observations are available throughout the domain too (especially for satellite data). Whereas for application of volcanic ash transport, the source emission is only at the volcano, thus usually only a limited domain is polluted by ash. As shown in Fig. 3, in the 3D domain with grid size of \( 3.888 \times 10^6 \), the number of grids in the area with volcanic ash is counted as \( 1.528 \times 10^6 \), whereas the number of no-ash grids is \( 2.36 \times 10^6 \). Note that shown in the figure are accumulated ash coverages of all ensemble states, thus in the no-ash grids, there are no ash for all the ensemble states. Thus a very large number of rows in \( A^f \) are zero corresponding to the no-ash grids. These zero rows in \( A^f \) have no contributions to \( A^a = A^fX \), because a zero row in \( A^f \) always results in a zero row in \( A^a \). Therefore, for the case of Fig. 3, \( \frac{2}{3} \) of the computations are redundant and can be avoided. To realize this, one may think to limit the domain for the entire assimilation steps, then the number of zero rows certainly would be largely reduced. This is actually incorrect, because
these zero rows are changing along with the transport of ash clouds, and not constant at each analysis step. So the full domain must be considered and it should be adaptive (choose different zero rows according to different $A_f$ at different analysis time).

4.2 Derivation of the mask-state algorithm (MS)

Here we introduce item $n_{noash}$ to represent the number of zero rows in the ensemble state matrix $A_f$, and use $n_{ash}$ to represent the number of other rows (also $n_{ash}$ represents the grid size of ash plume). When computing $A_a = A_fX$, to avoid all the computations related to $n_{noash}$ rows with zero elements, the index of other $n_{ash}$ rows must be first decided. This index is meant to reduce the dimensions of $A_f$. After getting a $A_a$ with a dimension of $n_{ash} \times N$, the index will be used again to reconstruct the full matrix $A_a$ with the dimension of $n \times N$. Based on this idea, we propose a mask-state algorithm (MS) which deals with the time-consuming analysis update. MS includes five steps:

(i) **Compute ensemble mean state $\bar{A}_f$**: The mean state $\bar{A}_{f \times 1}$ can be easily computed by averaging $A_{f \times N}$ along $N$ columns. Due to all elements in $A_{f \times N}$ corresponding to ash concentrations, thus all elements in $A_{f \times N}$ are larger than zero, so that the index of non-zero rows in $\bar{A}_f$ is equivalent to that in $A_{f \times N}$. The computational cost for this step is $O(nN)$.

(ii) **Construct mask array $z$**: Based on previously obtained $\bar{A}_f$, we search the non-zero elements of $\bar{A}_f$ and record the index into a mask array $z_{n_{ash} \times 1}$. With this strategy, we don’t need to search the full matrix $A_{f \times N}$ and build an index matrix for storage. This is a benefit for saving memory. The computational cost for this step is $O(n)$.

(iii) **Construct masked ensemble state matrix $\tilde{A}_f$**: Using the mask array $z_{n_{ash} \times 1}$ obtained from step (ii), $\tilde{A}_{f \times N}$ can be constructed column by column according to Eq. (13), and the computational cost (overhead) for this step is $O(n_{ash}N)$.

$$\tilde{A}_f(1 : n_{ash}, 1 : N) = A_f(z(1 : n_{ash}), 1 : N),$$

(iv) **Compute $\tilde{A}_a$ by multiplying $\tilde{A}_f$ and $X$**: Perform matrix computation $\tilde{A}_a_{n_{ash} \times N} = \tilde{A}_f_{n_{ash} \times N}X_{N \times N}$. This step is similar to $A_a = A_fX$, as described in Section 3.2, but the computational cost now becomes $O(n_{ash}N^2)$ instead of $O(nN^2)$.

(v) **Construct analyzed ensemble state matrix $A_a$**: With the computed $\tilde{A}_a$ from step (iv) and the mask array $z$ from step (ii), the final analyzed ensemble state matrix $A_a_{n \times N}$ can be constructed based on Eq. (14). The computational cost (overhead) for this step is $O(nN)$.

$$A_a(z(1 : n_{ash}), 1 : N) = \tilde{A}_a(1 : n_{ash}, 1 : N),$$
According to the derivations of MS, the computational cost related to zero rows are avoided. Here the “zero rows” doesn’t equal to “zero elements”. The former corresponds to the regions where there are no ash for all the ensemble members, while the latter also counts the no-ash regions specifically for some ensembles. Certainly the consideration of all “zero elements” can include all the sparsity information of the ensemble state matrix, but extra computations and memories must be spent on searching the full matrix $A^f_{n \times N}$ with a computational cost of $O(nN)$ and storing a mask state matrix with dimensions of $n \times N$. This is expensive compared to construct the mask array in the procedure (ii). Actually, after a careful check on the volcanic ash ensemble plumes, there is no “bad” ensemble which is really different from others. Although the concentration level in ensemble members are distinct, the main direction and the occurrence to the grid cells are more or less same. This means, the “zero rows” actually more or less equals to “zero elements”, but much faster than the way with “zero elements”, which confirms the suitability and advantage of procedure (ii). Probably when there are big meteorological uncertainties, the “zero elements” will be much larger than “zero rows”. In this case, how to make use of the sparsity information in the ensemble state matrix, will be considered in future.

Based on procedures of MS, the computational cost of $A^a = A^f X$ can be reduced. However, without a careful evaluation, we cannot conclude MS is fast, because the algorithm also employs other procedures. If these procedures (i)(ii) (iii)(v) are which explicitly specifies the expected amount of speedup ($S_{ms}$) of $A^a = A^f X$ by the MS algorithm. In this case study, $N$ is taken at 100 and $\frac{n_{ash}}{n} \approx \frac{1}{3}$, so $S_{ms}$ is approximately 3.0.
According to Amdahl’s law (Amdahl, 1967), the total computational speedup \( S_{\text{total}} \) by MS can be predicted by Eq. (16),

\[
S_{\text{total}} = \frac{1}{(1 - p_{\text{ms}}) + \frac{p_{\text{ms}}}{S_{\text{ms}}}},
\]

(16)

where \( p_{\text{ms}} \) is the proportion of the computational cost of \( A^a = A^fX \) in the overall data assimilation computations. It has been evaluated that the computational cost of \( A^a = A^fX \) dominates the analysis step (see Fig. 2b), thus the proportion of the computational cost of \( A^a = A^fX \) approximates the proportion of the analysis step in the total data assimilation computations (i.e., \( p_{\text{ms}} \approx 72\% \) in this case, as described in Section 3.1). Therefore, based on Eq. (16), the maximum (“ideal”) computational speedup can be predicted to be \( \frac{1}{1 - p_{\text{ms}}} \) (i.e., \( \approx 3.57 \) for this case study) when \( S_{\text{ms}} \) approximates infinity. However, this is not the actual speedup because \( S_{\text{ms}} \) is in fact specified by Eq. (15). (Based on discussions above, \( S_{\text{total}} \) can be therefore estimated by Eq. (15) at \( \approx 2.0 \) in this case.)

4.3 Experimental results

Analysis of the algorithmic complexity of the mask-state algorithm (MS) shows MS is an efficient approach to reduce the computational cost of the time-consuming \( A^a = A^fX \). Now MS will be applied in the real volcanic ash assimilation system, to investigate whether in practice it can well speed up the analysis step. We perform MS in the conventional EnKF, which means initialization, forecast steps are all computed as the conventional EnKF. The only difference between MS-EnKF and conventional EnKF is that in the former MS is employed for analysis step, and in the latter is standard analysis step. The result and related specifications are shown in Table 1. As introduced in Section 2, the forecast step has been configured with the conventional parallelization, thus \( N+2 \) (102 here) cores are actually used (one core for the data assimilation algorithm, the other \( N+1 \) cores for the parallel forecast of \( N \) ensemble members and one ensemble mean). It can be seen that MS indeed largely accelerates the analysis step (as expected, by a factor of about 3.0 for this study) which confirms the theoretical cost evaluation. The mask-state algorithm is now experimentally proven as efficient to significantly reduce the computational time for the analysis step during volcanic ash assimilation.

Note that it can also be observed that the computational time for the “other” parts in Table 1 (such as operations for setting environmental variables, starting and finalizing data assimilation algorithms, as mentioned in Section 3.1) is slightly reduced by the MS method (i.e., 0.03 h in this case). This is because in the conventional EnKF, the ensemble mean state \( \bar{A}^f \) is calculated in the “other” parts as an output to finalize the data assimilation algorithms, while in MS-EnKF, the calculations of \( \bar{A}^f \) are needed and directly involved in the “Analysis” part.

The result shows that benefiting from the success of reduced analysis step, the overall computational cost indeed gets significantly reduced. The total execution time is 1.95 h which is less than the simulation window of 3 h (09:00 – 12:00 UTC, May 18, 2010). This result satisfies our goal to accelerate the computation to an acceptable runtime (i.e., requires less run time than the time period of the data assimilation application). Therefore, aviation advices based on the MS-EnKF can be provided as not only accurate, but also sufficiently fast. Note that the result (1.95 h) is obtained after the volcanic ash is transported to the continental Europe. If the assimilation is performed in the starting phase of volcanic ash eruption (when aircraft measurements are available), a more significant acceleration would be obtained. This is because in this case the volcanic ash is only transported
in an area near to the volcano, thus the number of no-ash grid cells will take a large proportion (much higher than $\frac{2}{3}$ for this case study) of the full domain.

Another note is that in this study, we only perform the commonly used ensemble parallelization for the forecast step (already efficient compared to the expensive analysis step), but do not choose model-based parallelization (e.g., tracer or domain decomposition). As specified in Table 1, no parallelization is implemented on the 6 tracers. This is because due to the important aggregation process (Folch et al., 2010), there are big dependencies between different ash components and thus it doesn’t make much sense to parallelize them. As for domain-decomposed parallelization (Segers, 2002), it is not efficient here. This is because volcanic ash is special in the sense that the model is only doing computations in a small part of the domain (i.e., there is no data in a rather large part of domain), but this part is continuously changing. Thus, a fixed domain decomposition is not very useful here because of the changing plume position. In this sense, some advanced approach such as adaptive domain-decomposed parallelization (Lin et al., 1998) might add additional acceleration to the volcanic ash forecast stage. This is an interesting subject for future in case, when a more complicated model is employed, only ensemble parallelization may be not enough for the forecast stage.

5 Discussions

5.1 Applicability

For volcanic ash forecasts, only a relatively small domain is polluted compared to the full 3D domain, so that the mask-state algorithm (MS) can work efficiently. For other applications with similar characteristic (e.g., exploding nuclear plants or factories, chemicals or oil leaking on seas), MS can achieve the same effect on the computations of $A^n = A^f X$. It has been analyzed that when the number of non-zero rows ($n_{ash}$, i.e., the number of ash grids in a 3D domain) of $A^f$ satisfies $n_{ash} < n$, MS can work faster than standard EnKF. For volcanic ash application, because $n_{ash}$ is much less than $n$, the acceleration is thus quite large. Hence in this case, we propose to embed the mask-state algorithm (MS) in all ensemble-based data assimilation methods because it is fast and the implementation using MS is exact to the standard ensemble-based methods, i.e., it doesn’t introduce any approximation in view of MS procedures. Actually this proposal can be extended to all real applications, even if the condition is not satisfied. This is because, in this case the computational cost of MS for $A^n = A^f X$ becomes $O(nN^2)$, which is the same as that of using the standard assimilation (shown in Fig. 2b). Therefore, if the state numbers equal to or close to the number of the total number of grid points in the domain, the added computational cost by using MS is very small (neglectable), so that the computational time with MS is almost the same as the time of using the standard approach. Whereas, when the condition $n_{ash} < n$ is satisfied, MS will accelerate the analysis step. Thus MS is generic and can be directly used in any ensemble-based data assimilation, and this acceleration can be automatically realized for some potential applications, without spending time investigating if the condition is satisfied. In a real (or operational) 3D assimilation system, MS can be easily included, i.e., we only need to invoke the MS module when computing $A^n = A^f X$, without any other change to the current framework.
As stated in Eq. (15), the speedup of the MS method is approximately the inverse of $\frac{\text{mass}}{n}$. So far there is no statistical data on the value of $\frac{\text{mass}}{n}$. Consider the problem of volcanic ash transport, there is only one emission point (at the volcano), all the ashes in atmospheres are transported by the directional wind drive from the same source point. Thus volcanic ash cloud is actually transported in a shape of a plume, which in general doesn’t cover the full but only a small part of the 3D domain. At the start phase of a volcanic ash eruption, $\frac{\text{mass}}{n}$ is much smaller than 1.0 (started from 0). During transport over a long time (one and a half months for this case study), $\frac{\text{mass}}{n}$ increases to approximately $\frac{1}{3}$. Therefore, the speedup of MS on volcanic ash data assimilation will be significant.

5.2 MS and localization

Based on the formulation of MS, one may think it can be taken as a localization approach ((Hamill et al., 2001)). There is indeed a similarity between MS and the localization approach, in a sense that when computing $A^a = A^f X$, both get rid of a large number of cells, and only do computations related to the selected grids. These two algorithms are however functionally different. This is because the localization approach is meant for reducing spurious covariances outside a local region which is built up around the measurement, thus the results with and without localization approaches are different. While, MS is developed for the acceleration purpose. The masked region is discontinuous and independent of locations of measurement, but dependent on the model domain. Thus, there is no difference on the assimilation results between using MS and without using it. Therefore, based on the functional difference, MS cannot be taken as a localization approach.

In this study, we don’t employ the localization strategy in the analysis step, because we use a rather large ensemble size of 100 to guarantee the accuracy, as introduced in Section 2. But for some applications (e.g., ozone, CO$_2$, sulfur dioxide) especially when assimilating satellite data, localization is a necessary approach and has been widely used in reducing spurious covariances (Barbu et al., 2009; Chatterjee et al., 2012; Curier et al., 2012). In these cases, because the localization approach forces the analysis only to update state within a localization region, one may think that localization could replace MS and there would be no significance to employ MS. Actually this is not correct. We explain the reason as follows.

The localization approach is usually realized in Eq. (6) by employing a Schur product of a localization matrix and the forecast error covariance matrix (Houtekamer and Mitchell, 1998, 2001) given by:

$$ K(k) = (f \circ P^f(k))H(k)^T[H(k)(f \circ P^f(k))]H(k)^T + R]^{-1}. $$

(17)

The Schur product $f \circ P^f$ in Eq. (17) is defined by the element-wise multiplication of the covariance matrix $P^f$ and a localization matrix $f$. $f$ is defined based on the distance between two locations, thus it is dependent on the domain and needs information of the full ensemble state locations. In this way, $f \circ P^f$ can contain more zeros than $P^f$, but the dimensions are not changed, so that the computations related to $f \circ P^f$ are actually not reduced. Therefore, we can understand the localization approach in the analysis step as that the state within and outside a local region are both updated with increments, but just the increments outside the region are zero (which seems like not updating). This is also the reason why the localization approach is not meant for acceleration but only for reducing spurious covariances. Now it is clear that localization cannot replace MS. Actually both can be performed together in dealing with the time-consuming part $A^a = A^f X$. The localization approach can
first transfer $A^f$ to a localized matrix with more zero rows. Then MS can be used to accelerate the multiplication of the localized matrix and $X$. In this way, MS is expected to accelerate $A^a = A^f X$ with a high speedup rate, because the computational cost of more zero rows in the localized ensemble state matrix are avoided.

6 Conclusions

In this study, based on evaluations on the computational cost of volcanic ash data assimilation, the analysis step turned out to be very expensive. Although some potential approaches can accelerate the initialization and forecast step, there would be no notable improvement to the total computational time due to the dominant analysis step. Therefore, to get an acceptable computational cost, the key is to efficiently reduce the execution time of the analysis step.

After a detailed evaluation on various parts of the analysis stage, the most time-consuming part was revealed. The mask-state algorithm was developed based on a study on the characteristic of the ensemble ash states. The algorithm transforms the full ensemble state matrix into a relatively small matrix using a constructed mask array. Subsequently, the computation of the analysis step was sufficiently reduced. The mask-state algorithm is developed as a generic approach, thus it can be embedded in all ensemble-based data assimilation implementations. The extra computational cost of the algorithm is small and usually neglectable. The mask-state algorithm currently is only designed for the sequential case. Actually this approach can also be adapted for parallel implementation. This is because the related matrix multiplication can be easily parallelized on multiple processors. Optimization and evaluation on the parallelized mask-state algorithm will be considered in future.

The conventional ensemble-based data assimilation with the mask-state algorithm is shown to successfully reduce the total computational time to an acceptable level, i.e., less than the time period of the data assimilation. Consequently, timely and accurate volcanic ash forecasts can be provided for aviation advices. This approach is flexible. It boosts the performance without considering any model-based parallelization, such as domain or component decomposition. Thus, when a parallel model is available, the mask-state approach can be easily combined with the model to gain a further speedup. It implements exactly the standard data assimilation without any approximation and with easy configurations, so that it can be used to accelerate the standard data assimilation in a wide range of applications.

The use of aircraft in situ measurements is the essential reason why the mask-state algorithm perfectly works. For each analysis step, the number of measurements are quite small, and the procedure of the singular value decomposition (SVD) costs little. However, in other applications when many measurements are assimilated (e.g., satellite-based or seismic-based data), and the number of measurements is of the same order as the number of state variables, the most time-consuming part will be the SVD. In these cases, the contributions of the mask-state algorithm will be limited. The final improvement using the mask-state algorithm therefore cannot be significant, so that the efficient acceleration algorithm for the analysis step must be adjusted and should consider the computationally-expensive SVD in the first place.
7 Data and code availability

The averaged aircraft in situ data used in this study are available from Fig. 1c. The used continuous aircraft data and the model output data can be accessed by request (G.Fu@tudelft.nl). The mask-state algorithm is implemented in OpenDA (the open source software for data assimilation, www.openda.com) and the software can be downloaded from sourceforge (https://sourceforge.net/projects/openda).

Author contributions. Guangliang Fu, Sha Lu and Arjo Segers simulated the volcanic ash transport using the LOTOS-EUROS model. Guangliang Fu, Hai Xiang Lin, Tongchao Lu evaluated the computational efforts. Guangliang Fu, Hai Xiang Lin, Arnold Heemink and Shiming Xu developed the mask-state algorithm. Guangliang Fu and Nils van Velzen carried out computer experiments and analyzed the performance of the mask-state algorithm in OpenDA. Guangliang Fu and Hai Xiang Lin wrote the paper.

Acknowledgements. We are very grateful to the editor and reviewers for their reviews and insightful comments. We thank the Netherlands Supercomputing Center to support us the “Cartesius” cluster for the experiments in our study. We are grateful to Professor Konradin Weber for providing the aircraft measurements.
References


Figure 1. Methodology of ensemble-based data assimilation. a, The initial volcanic ash state at 09:00 UTC. b, Flight route of measurement aircraft. c, Aircraft in situ measurements of PM$_{10}$ and PM$_{2.5}$ from 09:30 to 11:10 UTC May 18, 2010. d, Volcanic ash assimilation result at 12:00 UTC.
Figure 2. Computational evaluation of the analysis step. a. Illustration of the analysis step. b. Computational cost of all sub-part of the analysis step.
Time: May 18, 2010 at 10:00 UTC.

Ash Coverage
- Volcanic ash plume
  - (1.528 * 10^6 ash grids)
- No volcanic ash
  - (2.36 * 10^6 no-ash grids)

Figure 3. Characteristic of volcanic ash state.
Table 1. Comparison of the computational cost of conventional EnKF and MS-EnKF. (The results are obtained from the bullx B720 thin nodes of the Cartesius cluster, which is a computing facility of SURFsara, the Netherlands Supercomputing Centre. Each node is configured with $2 \times 12$-core 2.6 GHz Intel Xeon E5-2690 v3 (Haswell) CPUs and with memory 64 GB.)

<table>
<thead>
<tr>
<th>Case</th>
<th>Conventional EnKF</th>
<th>MS-EnKF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cores used</td>
<td>102</td>
<td>102</td>
</tr>
<tr>
<td>Tracer number ($n_{spec}$)</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>Measurements of tracers (m)</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Ensemble size (N)</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Parallel in forecast step</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Parallel in analysis step</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Mask-state in analysis step</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Initialization</td>
<td>0.42 h</td>
<td>0.42 h</td>
</tr>
<tr>
<td>Forecast</td>
<td>0.65 h</td>
<td>0.65 h</td>
</tr>
<tr>
<td>Analysis</td>
<td>3.14 h</td>
<td>0.88 h</td>
</tr>
<tr>
<td>Others</td>
<td>0.15 h</td>
<td>0.12 h</td>
</tr>
<tr>
<td>Total Runtime</td>
<td>4.36 h</td>
<td>1.95 h</td>
</tr>
</tbody>
</table>

*h=hour, simulation window = 3.0 h, the time is Wall clock time.