

Response to Reviewer #1:

> Comment:

The authors have satisfactorily addressed most of the concerns raised in the previous round of review. However, they overlooked one of my general comments regarding their methodology.

The main contribution of this paper is the use of automatic differentiation to replace traditional manual gradient derivation in the 3DVar system. Despite this, the manuscript lacks sufficient detail on exactly how PyTorch was implemented to achieve this in the 3DVar assimilation.

Expanding on these technical details would be highly beneficial to the reader. Once this minor addition is made, the paper will be ready for publication.

** Response:**

We sincerely thank the reviewer for this constructive comment. We fully agree that the implementation details of how PyTorch's automatic differentiation (AD) is used to replace manual gradient derivation are essential for the reader to understand and reproduce our method.

In the revised manuscript, we have replaced the original brief description of AD in Section 2.5 with a detailed explanation, while retaining only the opening sentence of the original PY3DVar paragraph ("PY3DVar is developed within the PyTorch framework and can be deployed on both CPU and GPU platforms without code modification."). A simplified pseudo-code has also been added as Fig. 2. These revisions are located in lines 318-339 of the revised manuscript and are highlighted in yellow.

Specifically, the revised Section 2.5 now clarifies the following points:

1. Control variable and differentiable quantities. In PY3DVar, the control variable is the incremental field δz , which is created as a `torch.Tensor` with `requires_grad=True`. All other quantities involved in the cost function — including the background-error transform operator F , the observation operator H , the innovation vector d , and the observation error covariance R — are precomputed before minimization and treated as constants.

2. Forward computation and computational graph. The complete forward mapping from δz to the scalar cost J is encapsulated in a closure function `closure()`. Inside this function, PyTorch's autograd engine records every tensor operation that depends on δz , thereby building a complete computational graph from δz to J .

3. Automatic gradient computation. Once the forward evaluation is complete, calling `cost.backward()` triggers reverse-mode automatic differentiation along the recorded computational graph, yielding the gradient $\nabla J(\delta z)$ without any manually coded adjoint.

4. Coupling with the L-BFGS optimizer. During minimization, the L-BFGS optimizer calls `closure()` at each iteration, which performs four steps: zeroing the gradients, forward-computing J , backpropagating to obtain the gradient, and updating δz . Since L-BFGS invokes the closure multiple times during line search, the computational graph is rebuilt at each call, ensuring that the gradient always corresponds to the current δz .

5. Pseudo-code. A simplified pseudo-code summarizing the above workflow has been added as Fig. 2, which illustrates the forward cost function, the L-BFGS closure, and the iterative minimization loop.

We believe these additions provide sufficient technical detail for the reader to understand and reproduce the AD-based implementation in PY3DVar. We again thank the reviewer for this helpful suggestion, which has significantly improved the clarity of the manuscript.

Response to Reviewer #2:

> Comment:

L228: So the split is done before thinning, right? And the validation stations were not thinned but compared directly?

**** Response:****

We sincerely thank the reviewer for this insightful comment, which has helped us identify a critical flaw in our original procedure. The reviewer is correct: in the original manuscript, the split was performed after grid-based thinning. This was incorrect, because thinning (grid-based averaging) implicitly mixes observations from stations that would later be assigned to either the assimilation or the validation set. As a result, the validation dataset was not truly independent — it implicitly contained information from the assimilation stations through the grid-averaged values.

We have therefore completely revised the observation-splitting procedure in Section 2.2 (see lines 214–229, highlighted in yellow), following the scheme of Hu et al. (2022, Line 780). In the revised procedure:

1. The split is performed at the station level, and no thinning is applied. For each of the six pollutants, stations with valid observations at 00:00 UTC are grouped by model grid cell.
2. If a grid cell contains only one valid station, it is assigned to the assimilation set. If a grid cell contains two or more valid stations, one is randomly selected for the independent validation set, and the rest are assigned to the assimilation set.
3. The station assignment is performed daily and retained for all 25 time steps of the forecast period (from 00:00 UTC of the current day to 00:00 UTC of the next day).

4. Assimilation is performed only at 00:00 UTC; later time steps are used solely for validation. Observations from stations outside the corresponding validation sets are marked as missing and are not used for validation.

With this revision, the validation dataset is completely independent of the assimilation dataset. Based on the revised data-splitting procedure, we have re-run all assimilation and forecast experiments. Although some evaluation metrics have changed slightly, the core conclusions of the paper remain unchanged. The differences between the results obtained with the original and the revised splitting procedures are small, which further confirms the robustness of our conclusions.

> Comment:

L313: is there a reference (or a web page) for the Fortran library?

**** Response:****

We thank the reviewer for this comment. We would like to clarify the origin of the Fortran routines used in FT3DVar. The outer L-BFGS driver routine 'mbfgs' is adapted from publicly-shared Fortran implementations of the standard L-BFGS algorithm (Liu and Nocedal, 1989; Nocedal, 1980), with custom-tailored interfaces for our 3D-Var framework. It is not distributed as part of an official third-party Fortran library. The line-search kernel adopts the well-known Moré-Thuente algorithm, for which we cite its original publication (Moré and Thuente, 1994).

We have clarified these points and added the corresponding references in Section 2.5 of the revised manuscript (see lines 303, 309-311 and 821, highlighted in yellow).

In addition, we have substantially expanded the code and data availability in the revised manuscript. The updated repository now includes: (i) the Fortran 3DVar source code (FT3DVar), (ii) the PyTorch-based 3DVar source code (PY3DVar), (iii)

the observation data from CNEMC used in this study, including the assimilation and validation datasets after station splitting, (iv) the code for the observation splitting procedure, (v) the idealized experiment data with single-station observations, and (vi) the background error covariance modeling code. The previously deposited Zenodo record has been deprecated, and all code and data are now available at the new DOI: <https://doi.org/10.5281/zenodo.23077072>. The "Code and data availability" section (see lines 694-697) and the corresponding entry in the reference list (see lines 847-848) have both been updated accordingly, and both are highlighted in yellow.

> Comment:

Figure 2: The legend box has micrograms per m³ for O₃ but the caption has milligrams.

**** Response:****

We thank the reviewer for this careful observation. We have checked the legend box and caption of the figure, and confirmed that the units are consistent: in both the legend box and the original caption, CO is in mg m⁻³ and O₃ is in µg m⁻³. The apparent inconsistency may have arisen from the layout, where CO and O₃ are placed close to each other in the same subplot (f), making the units easy to confuse.

To eliminate any potential ambiguity, we have revised the caption of this figure as follows (note that this figure is now numbered as Figure 3 in the revised manuscript, because a new pseudo-code figure has been inserted as Figure 2):

Figure 3. Vertical profiles of background error standard deviation for each control variable: (a) EC, NH₄, SO₄, NO₃; (b) ORG, CPO, CL, P25; (c) DUST1, DUST2; (d) ANTHA, SOILA; (e) SO₂, NO₂, CO; (f) O₃.

Specifically, we have removed the unit description from the caption, because the units

are already explicitly labeled in the legend box of each subplot. In addition, we have moved O₃ to a separate subplot (f), while CO remains in subplot (e). This is because O₃ exhibits a distinct vertical profile with high concentrations in the upper troposphere, which differs substantially from the other variables and is better displayed in a separate panel.

The revised caption is shown in lines 389-391, highlighted in yellow.

> Comment:

Figure 2: The standard deviations for NO₂ and SO₂ look very small compared to the analysis increments in Fig. 5. Please check.

**** Response:****

We thank the reviewer for this careful observation. We have checked the data and confirmed that the D matrix itself is correct. The issue was that a unit conversion (from the model's internal units to $\mu\text{g m}^{-3}$) was accidentally omitted when plotting the profiles of the gaseous pollutants. This made the plotted values appear two to three orders of magnitude smaller than the actual ones.

Upon checking all control variables, we found that the same omission affected all four gaseous species (SO₂, NO₂, CO, and O₃), whereas the aerosol variables were unaffected. We have now corrected the unit conversion in the plotting script for all four gaseous species and regenerated the figure. The corrected profiles are now physically consistent with the analysis increments shown in Fig. 6.

The revised figure is now numbered as Figure 3 in the revised manuscript, because a new pseudo-code figure has been inserted as Figure 2.

> Comment:

Section 4.1: It seems like a bit of overkill to show everything (Figs. 5-11) for both systems, since the results are indeed identical for all practical intents and purposes. But I leave this to the authors.

**** Response:****

We thank the reviewer for this suggestion. We agree that the results from the two systems are nearly identical.

However, we have decided to retain the full set of figures, because the near-identical results are precisely the core conclusion of this study — that PY3DVar achieves assimilation accuracy equivalent to FT3DVar while being much faster. All subsequent analyses (iteration behavior, computational efficiency, parallel scalability, and forecast performance) are built on this equivalence. Showing both systems side by side allows readers to directly verify this consistency.

Additional changes

In addition to the revisions requested by the reviewers, we have made the following changes to improve the manuscript:

1. In the Abstract, "traditional Fortran-based systems" has been changed to "FT3DVar" (Line 29), since the term "FT3DVar" has already been defined earlier in the Abstract as "the classic Fortran-based 3DVar system (FT3DVar)". This avoids redundancy and improves conciseness.
2. Re-run experiments with the revised data-splitting procedure. Following the revised observation-splitting method (see the response to Reviewer #2), we have re-run all assimilation and forecast experiments. The updated evaluation metrics are provided in the revised manuscript and are highlighted in yellow. Although some metrics have changed slightly, the core conclusions remain unchanged.
3. A new pseudo-code figure has been inserted as Figure 2. Consequently, all subsequent figures have been renumbered, and the corresponding in-text references have been updated accordingly. In addition to the four locations listed above, all other in-text references to figures originally numbered 2 or higher have been updated accordingly.
4. Figure format. For Figures 3, 5, 6, 7, 8, 9, 10, 13, and 14, the individual subpanels were previously inserted into the manuscript as separate images arranged within tables. In the revised manuscript, these subpanels have been manually assembled into single composite figures and embedded directly into the Word document, without using tables.
5. Author contributions. Following the editorial request, we have used the initials of the authors in the "Author contributions" section (see lines 699-704, highlighted in yellow).

6. In the Introduction, the citation “Key et al. (2024)” has been corrected to “Key et al. (2025)” to be consistent with the reference list. This change is located at line 105 and is highlighted in yellow.

7. The citation “De et al., 2021” has been corrected to “De Luca et al., 2021” to match the corresponding reference entry. This change is located at line 123 and is highlighted in yellow.

8. The citation “Parrish et al., 1992” has been corrected to “Parrish and Derber, 1992” to be consistent with the reference list. In addition, the full name of the NMC method has been clarified as the “National Meteorological Center (NMC) method” in Section 2.6. These changes are located at lines 343 and 344 and are highlighted in yellow.

9. The reference “Carter, W. P. L.: Documentation of the SAPRC-99 chemical mechanism for VOC reactivity assessment, Final Report to California Air Resources Board Contract No. 92-329, University of California, Riverside, 2000.” has been added to the reference list (Line 728-731). The corresponding in-text citation “Carter (2000)” appears in Section 2.1 and is highlighted in yellow (Line 171).