

Discussion response to reviewer 1:

Thank you very much for the review and your helpful comments. Given that the discussion phase for the manuscript was extended yet another time, we thought it would be nice to already respond now. Since this is the response in the discussion phase and not the formal reply for the manuscript revision, we aim to describe more on what we could or intend to do. The revision will of course also depend on the comments of other reviewers, which are not yet available. Please see more details below:

This manuscript describes the most recent release of the PDAF data assimilation software system. PDAF is widely used and provides valuable functionality for Earth system science and so the material is a welcome addition to the literature. The first three sections do an admirable job describing the newest release, in particular how it can be interfaced to models and what capabilities it provides. I felt that sections 2 and 3 are quite mature and do not need significant modification before final publication.

I did find the examples in section 4 to be a bit less useful. I'm not sure exactly what readers were being targeted by this. The simple model results do little to provide evidence of the quality of the PDAF software. The level of detail provided in the description is too little to be helpful without additional reference to the PDAF documentation. Personally, I think the manuscript would be better without these simple model results.

The section is mainly motivated by GMD's usual requirement that each paper includes an application. Thus, it might not be possible to omit this. We have chosen the simple model because it is particularly easy to understand and it is not specific to any discipline. E.g. using a real ocean model would involve the complexity of the ocean model, e.g. handling of restart and forcing files, to demonstrate the cycled data assimilation. To avoid running such a specific model we could only consider the offline-coupled data assimilation using restart files from an ocean model for a single analysis step, i.e. without cycling. This would avoid running the actual model, but would not allow us to demonstrate the similarity of the implementation and functionality for online and offline coupled data assimilation and it would be specific for oceanography. We can check with the Editor what he recommends. A possibility could be to include more examples of the diagnostics that PDAF provides, of choices of the different DA methods, and perhaps also results from scaling experiments (see also our response to the next comment). For the latter, the simple example can be easily configured with a large model grid and more observations and can then be run on a bigger compute cluster.

I was a bit concerned about the lack of any description of the computational performance. A single sentence suggests that it is expected to be the same as earlier versions. I would hope that a revised manuscript could provide a little bit more information about the performance for some reasonably sized Earth system model.

We plan to include information on the performance, perhaps mainly on what was described in previous publications. To name an example, in Nerger et al. (2020) we used the coupled

atmosphere-ocean model AWI-CM with ECHAM6 as atmosphere at T63 resolution and the finite-volume ocean model FESOM with varying resolution around one degree in a global configuration. Assimilating satellite sea surface temperature observations, the analysis step usually took below 1 second using 72 processes (MPI tasks) for ECHAM and 192 processes for FESOM. Running the ensemble with up to 46 states fully parallel in the online coupled setup led to very small overheads of 10-15% in total run time when comparing to the model without data assimilation. However, we are considering to include experiments using the simple model used in Section 4, but with a larger model grid to show the computational performance and scalability.

The manuscript did not discuss how easily PDAF can be compiled on a variety of computing platforms. Many Earth system models require extensive effort to compile and run with compiler and hardware variants. Can the authors give any insight into how easy it is to port PDAF?

We overlooked adding such information because the compilation of the PDAF library is extremely easy. PDAF itself only needs an MPI library, and the BLAS and LAPACK libraries. With commercial compilers, like the compilers by Intel or NEC, the numerical library is provided by the vendor in some optimized form (e.g. Intel's MKL library). PDAF itself is standard Fortran 2003 (as e.g. checked setting the corresponding compiler flag) and we never saw compile issues. When we tested the compilation on new computers, e.g. NEC's Aurora vector computer or compute nodes using ARM64 processors it was merely minutes to specify the compiler and the libraries. Afterwards the compilation worked without any issues and we didn't see performance issues. This was for offline coupling. In case of online coupling, PDAF is still compiled as a library (easy as described above), but the model-specific user code is compiled with the model and the model, user code, and PDAF library are then linked into the executable. For compiling the user code one usually needs to add these to the compilation of the model. Whether this is complex depends on the build process of the model. To this end the complexity of the compilation is rather in the model than PDAF. We plan to include information on the ease of compiling the PDAF library and on compiling for the online coupled case in the revision.

I was a bit surprised by the small volume of code required to implement a new model with arbitrary grids. Accurate interpolation on unstructured grids can often require sophisticated specific computation. This can be particularly important to avoid systematic bias when computing forward operators for Earth system models. This problem is more common in the vertical but can also occur in the horizontal for grids with spatially varying grid density. Could the authors provide a little bit more detail about how the default PDAF tools do interpolation for supporting forward operators?

The provided example code is specific for the regular 2-dimensional model grid and observations that are located at model grid points. The PDAF-OMI system provides functionality for an observation operator with linear interpolation. The actual observation operator just uses interpolation coefficients and, using these, it can combine values from any number of grid points. The interpolation coefficients are initialized before the observation operator is applied (because the operator is applied for each ensemble state using the same coefficients for each). Here, PDAF-OMI provides routines to determine interpolation coefficients for rectangular and for (unstructured) triangular model grids. In the latter case, a barycentric interpolation is used e.g.

for the FESOM model which uses an unstructured triangular model grid. More complex or sophisticated cases, e.g. for tri-polar model grids, are not yet provided, basically because in our applications we didn't have such a case yet. However, even for such model grids one could provide the functionality in the PDAF library, such that the user code remains short. We noted that details on the available observation operators are actually missing in the manuscript. To this end we plan to add these in the revision.