Reply on RC2

Hui Wan (on behalf of all authors)

We thank the referee for the very positive assessment and the helpful questions and suggestions. Our responses to the specific comments are listed below.

Referee comment: The manuscript indicated that the version 1 of this tool has implemented in EAM with several successful use cases. EAM is the target host model for now, but it is flexible to be used in other AGCMs. Some possible improvement to make the paper/tool more comprehendible for the general audience:

Elaborate on terminology that is specific to EAM/CAM, just to name few: physics state, physics buffer, cam_in, cam_out, tphysbc vs tphysac...

Author response: Thanks for the suggestion. A new subsection is added to Section 2, "Host model features", to explain the terms related to EAM/CAM-specific code structure and data structure.

Referee comment: The zenodo doi linked two versions of EAMv1 with and without CondiDiags, but it is not straightforward to see the codes of CondiDiags and how it interfaces with EAM. Not sure about the best approach, but I wonder if the authors can supply with an additional package with standard CondiDiags and a template that could be used as an example for external users?

Author response: We plan to add to the Zenodo archive a third, very small tar ball which contains only the source files that were either newly added or had to be revised during the implementation of CondiDiag1.0. A copy of the relevant EAMv1 files before the implementation will be included in this third tar ball as well.

Referee comment: Is there a general guideline about adopting ConndiDiags with other AGCM as a host model?

Author response: Thanks a lot for the interest. A new subsection (4.4, "Portability") is added to the revised manuscript to discuss this.

Referee comment: It is not explicitly indicated that if CondiDiags has already be available in EAM code base and readily to be used for EAM developers?

Author response: We developed CondiDiag1.0 using branches in E3SM's public GitHub repository at `https://github.com/E3SM-Project/E3SM`. For example, the revised EAMv1 code shared on Zenodo corresponds to a branch named `huiwanpnnl/maint-1.0_cnd_diag1.0rc`. In recent months, some EAM developers and users expressed interests in using the tool, so we ported it to the code versions they were using (which were typically development versions between v1 and v2). Given the positive feedbacks from the colleagues, we plan to discuss with E3SM's code integrators a pathway to get CondiDiag onto the `main` branch of the E3SM repository. Before that merge happens, our development branches are publicly available in the E3SM repository, and we would be happy to help port the tool to other colleagues' working branches.

Referee comment: Technical corrections: In session 2.1, the definition of "process" and "component" seem ambiguous. Ln 117-119: indicate that deep convection contains two sub-components, with the parameterization of impact of convection on temperature and humidity, the parameterization of convective momentum transport. Is process B a more frequent case than processes without sub-processes? It might be useful to have some definition on process and component?

Author response: Thanks for the feedback and questions. Nomenclature has been a challenge not only for this manuscript but also in our work on numerical process coupling in EAM which motivated the development of CondiDiag. Oftentimes, we use "processes" to refer to physical phenomena and "components" to refer to sections of the source code (for the latter, "compartment" might be a better name). Because the EAM/CAM code is largely modularized by the simulated physics, processes and components (or compartments) often have close relations. In the revised manuscript, we have added a new subsection to describe the data and code structures in EAM. Hopefully, the new text helps to clarify that EAM has a hierarchically modularized code structure in which we often see sub-processes or even sub-sub-processes.

Referee comment: Ln 130. Is there a relationship between the location of where outfld is called to check-points?

Author response: This is a very insightful question. The locations of `outfld` calls and the checkpoints are related to some extent, as both `outfld` and our `cnd_diag_checkpoint` subroutine have dummy arguments that use data structures dependent on EAM/CAM's column-chunk-based horizontal domain decomposition as well as EAM/CAM's vertical grid. A *single* call of `outfld` is designed to copy (to the history output infrastructure) the values of a *single* physical quantity stored in a *single* integer or floating-point Fortran array. In contrast, `cnd_diag_checkpoint` is designed to obtain values of a (runtime-specified) *collection* of physical quantities using the EAM/CAM-specific *derived-type* variables like `state` and `pbuf` etc. From this perspective, if the values sent to the history output infrastructure are also saved to the derived-type variables, then one could use a single `cnd_diag_checkpoint` to replace many scattered `outfld` calls.

Referee comment: Is the location of outfld process-depended?

Author response: We are not quite sure what is meant by "process-depended" in this comment. If it means every physical process has its own set of `outfld` calls that are placed inside the parameterization or immediately after a parameterization is invoked, then yes, the location of `outfld` is process-depended.

It is perhaps useful to mention that `outfld` calls are seen before and after many parameterizations and in multiple levels of subroutines. If an `outfld` call copies the values of a physical quantity from an array that is local to the subroutine and these values are not saved to persistent derived-type data structures or passed to other subroutines, then the `outfld` call needs to be placed in the subroutine where the physical quantity is calculated. On the other hand, if the values being written out are saved and hence persist for a while within a time step, then one will have more options for the location of the `outfld` call.

Referee comment: For the inactive checkpoint, are they off by default but can be turned on easily with name list change?

Author response: All checkpoints are inactive by default. A checkpoint is turned on (i.e., become active) when the user mentions its name in the namelist. Re-compilation of the model source code is not needed for requesting different sets of checkpoints.

Referee comment: Ln 237: "If a value of 101 (moist) or 102 (dry) is used, the the. . . ." Shoud be ". . . , then the"?

Author response: Corrected. Thanks.

Referee comment: Session 5.2.4 The process of turning on vertical integral and assign moist/dry air mass for mass-weighting is a little hard to comprehend. Maybe try simplifying if possible. . .

Author response: We have heard similar feedback from other colleagues and have rewritten this part. Hopefully the new version is easier to understand.