Letter to Editor

Dear Editor,

Here we submit the reviewed version of our manuscript entitled “Code accessibility and code quality across phases of the models of the Coupled Model Intercomparison Project” (Manuscript ID: egusphere-2025-6108) submitted to Geoscientific Model Development.

We would like to apologize for the mistake made in the previous submission, when we uploaded the wrong documents for the reply. We have fixed it now.

We appreciate the positive evaluation of the relevance of this work and the suggestions made by the reviewers to improve clarity, transparency, and presentation. A letter with answers to your comments follows.

Thank you for your consideration.

On behalf of all the coauthors,

Dr. Michael García-Rodríguez

EPhysLab, Universidade de Vigo

https://ephyslab.uvigo.es/michael-garcia-rodriguez/

Reply to reviewers

Reviewer 1

In this paper, the authors study the accessibility, understandability, and licensing of climate models used across all CMIP exercises. They found that 18 models out of 121 were accessible, with one being from CMIP3, ten from CMIP5, and seven from CMIP6. Then, with the accessible models, the authors performed code analysis. They found out that all but one were written in Fortran, the exception being NICAM.09. Finally, for the models written in Fortran, they scored the code using the FortranAnalyser, a static code analysis tool that grades Fortran code from zero to ten. Their findings show that the quality of the code has increased with the newer phases of CMIP.

This work is of great importance as it provides an overview of the accessibility of the climate models used in CMIP. Equally important is its discussion regarding the reproducibility of these experiments, which are extremely consequential for advancing our understanding of climate.

I have three critiques for the authors.

1 – In section 5, "Code Analysis," I would like to have a clarification, in general terms, of what the FortranAnalyser analyzes to score the models. Moreover, if possible, give an intuition of what a high score code would be. This is because the authors claim that a score of 4.014 is "relatively high," but I have no reference to compare this value with.

We thank the reviewer for this comment and agree that further clarification is useful.

A detailed and comprehensive description of the metrics, methodology, and scoring system implemented in FortranAnalyser is provided in a dedicated article published in Software Impacts (García-Rodríguez et al., 2024). In that work, the reader can find the full technical explanation of what FortranAnalyser evaluates, including the complete set of static metrics, how they are computed, and how they are aggregated into a final normalized score ranging from 0 to 10. For this reason, the present manuscript includes only a high-level description and cites that companion paper for methodological details.

With regard to the interpretation of the score, we agree that the expression “relatively high” requires additional context. This wording is not meant to imply a high absolute level of software quality in a general software engineering sense. Rather, it is used strictly in a comparative, empirical sense, relative to the distribution of scores obtained for the set of accessible CMIP models analyzed in this study.

Specifically, although the theoretical scale of FortranAnalyser spans from 0 to 10, the scores obtained for CMIP models fall within a much narrower empirical range between 2.7 and 4.3, despite covering several CMIP phases. Within this observed range, a value of 4.014 lies in the upper part of the distribution and is therefore relatively high with respect to other operational CMIP models, including many from more recent phases. At the same time, the fact that no model approaches even half of the theoretical maximum score underlines one of the central conclusions of the study: namely, that there remains substantial room for improvement in the structural quality and maintainability of climate model code, even among the best-performing cases.

We include this specific information in lines 88 to 96:

“FortranAnalyser evaluates code quality using metrics that can be grouped into two scoring categories. Metrics rated on a 0 to 2.0 scale include the use of implicit none, the presence and quality of comments across different code elements (files, variables, functions, subroutines, and control structures), the number of nested loops (as an indicator of structural complexity), and the comment-to-code ratio. In contrast, metrics rated on a 0 to 1.0 scale focus on specific control flow practices, namely the use of the exit and cycle statements. Together, these grouped metrics provide a balanced assessment of code readability, maintainability, and structural quality. FortranAnalyser gives additional information too; number of lines, the cyclomatic complexity, the number of functions declared, the number of variables declared, and the number of calls to subroutines among others. According to the accomplishment of these metrics, FortranAnalyser assigns a score from 0 to 10 (where 0 is the minimum score and 10 is the maximum) to each file analysed and generate a PDF quality report with the results.”

2 – In the annexed table, where all the models are present, there is overlapped text (page 18, page 20).

We thank the reviewer for pointing this out. The overlap in the annexed table is a formatting issue introduced during the compilation of the manuscript. This mistake is solved.

3 – In the references, the last accessed text is written in Spanish.

We appreciate the reviewer’s attention to this detail. The references section is corrected now.

Once again, we thank the reviewer for the constructive feedback, which will help us improve the clarity and presentation of the manuscript.

Reviewer 2

This manuscript introduces a systematic evaluation of code accessibility and code quality across the CMIP phases. The authors tried to access the code for as many climate models as possible that participated in all past CMIP phases. After discussing the code accessibility, programming languages and code quality were examined. This study is very useful to explain the status quo, but also to identify needs for future CMIP phases.

I have three comments:

1 - At the beginning of Section 3 (L.89/99), I am confused about the different numbers. First, it is written about "59 out of the 262 models", and then "18 out of 121". What is the difference? Are the second numbers only about the subset of coupled models? But then in L.114, there are "59 out of the 121 couples models" mentioned. How is this connected? Maybe you can make that clearer.

Now the manuscript includes a clarified explanation about these concepts from line 99 to 105:

"Following the attempts made, successful access was obtained for 59 of the 262 individual models identified across all CMIP phases (see Figure 1 and Table A1). Here, the total of 262 refers to all individual models considered in this study. Within this broader set, a subset of 121 corresponds specifically to coupled climate models, which represent the core systems used in CMIP experiments.

Out of these coupled models, 18 were successfully recovered; one from CMIP3, ten from CMIP5, and seven from CMIP6. Thus, the 59 accessible individual models include these 18 coupled models as a subset. For CMIP1 and CMIP2, no model could be recovered, as their code had not been preserved over time. CMIP7 models were not included in this study because this is the current CMIP phase of the project, and active development and usage are still ongoing."

2- In Section 4 about the programming languages, it would be very interesting to add an outlook or expectation about changes when hybrid models are added to the pool of CMIP models.

Section 4 has been modified by adding our expectations about these changes from line 159 to 172:

"The emergence of hybrid modelling approaches, combining physics implemented to code with machine learning or artificial intelligence techniques, is expected to further increase this heterogeneity. Although such approaches have not yet been integrated within the CMIP phases analysed in this study, their incorporation is foreseeable in future developments. These hybrids models may introduce new programming ecosystems alongside established high performance languages like Fortran. From a software engineering perspective, this transition may significantly impact code structure, dependencies, and development workflows. In particular, machine learning components introduce additional layers of complexity related not only to the implemented code, but also to training data, calibration procedures, and evolving model configurations. This raises important challenges for reproducibility and traceability, as the behaviour of such components may depend on factors beyond the static source code itself. Consequently, evaluating future CMIP models may require extending current approaches to software quality assessment. Traditional static code analysis techniques, such as those employed in this study, may need to be complemented with methodologies capable of addressing data provenance, model training processes, and the interpretability of learned parameterisations. In this context, hybrid models could be more appropriately analysed as systems composed of heterogeneous components, combining multiple tools specialised to a

programming language and tools specialised on the analysis of machine learning usage modelling paradigms in order to obtain an integrated assessment of the full system."

3 - In Section 5, it would be helpful to have some examples of lacks in the code quality which are found by the analysis tool. And which parts or topics should be focused on primarily by the modelling groups to improve their model code?

We add the following information about a real case to show an example of lacks in the code quality by adding from line 203 to 205:

"An example of improved Fortran code by applying FortranAnalyser recommendations is the case of the IPSL climate model, which improved its quality after an assessment performed with the tool. The case study is described in (García-Rodríguez et al., 2024)."