MESMER v1.0.0: Consolidating the Modular Earth System Model Emulator into a Sustainable Research Software Package

Review file

Referee Comments 1

Comment 0

This paper presents the modular climate model emulator (MESMER v1), its components, and strategy for integration, and serves as detailed documentation in addition to the online documentation for users. The authors also demonstrate that this rewritten emulator now meets sustainable research software standards. The output of the emulator includes annual and monthly mean temperature and several climate extreme indicators, such as maximum annual temperature. The paper provides a thorough description of the emulator, improvements made as well as performance and sustainability assessment. However, prior to publication, the following comments and questions should be addressed:

Response

Thank you for the positive evaluation of our study and the helpful comments. We address the comments made below and explain the modifications we made to the study.

Comment 1 - Motivation of the study

The motivation of this study appears to focus mainly on making the emulator accessible and providing documentation on how to use it (as inferred from the introduction: Lines 30-31). However,

users may also want to be able to read and understand the model source code, extend and modify its functionality, and contribute to maintenance (if possible, e.g., by reporting bugs) in order to improve the overall software sustainability of the emulator. Therefore, I recommend revising the motivation around lines 30–31 to more fully expand on these aspects.

Response

Thank you for this comment, we agree the motivation in the paragraph is too narrow.

Changes to the manuscript

We added the following in line 31:
The emulators' source code should be easily available, understandable and extensible in order to open possibilities for new applications and further democratize climate information.

Comment 2 - State of the old MESMER software

What is currently not clear to me is the state of the old (legacy) MESMER software. You only hint at this with statements such as organic growth of the code base (Section 1: Introduction, lines 36–38) and development of software by multiple researchers without a focus on software design (Section 3: Design of MESMER v1, lines 98–99). Was documentation available for the old software? Was the old software easily accessible by users? Is the code of the legacy MESMER difficult to read and understand? What is the state of code comments in the old code? Were the separate components of old MESMER all integrated as shown in this paper? Did the old software have a well-defined software architecture? I expected to see a section that answers and expands on these points.
This would also be beneficial during your assessment (Section 3.6: Assessment of the redesigned code base), as it would help to understand how much the rewritten code improves upon the old one. Additionally, did the state of the old code require rewriting everything from scratch (or only parts of it)? This also highlights the importance of making software sustainable, as complete rewriting requires significantly more time and effort. This is an issue that is now slowly gaining attention in academia.

Response

We now added a subsection “State of the legacy Software” to section 2, which should answer these questions. We did not have to rewrite MESMER from scratch. Mostly, we reused the basic statistical functionalities, but split them up into modular bits, built new APIs for them, wrote wrappers for the new data-structure, and unified common functionalities between the different components. For the manuscript however, we prefer to emphasise the key features of the new

software for potential users, and somewhat less on the shortcomings of the legacy code or the process of the rewriting itself.

Changes to the manuscript

We added section 2.6 State of the legacy Software:

Previously, the individual MESMER components listed above were published in separate repositories, or, in the case of MESMER-X, not accessible for the public at all. Overall, there was no overarching software architecture. While all components were written in Python, the structure varied across components. Some components were written in an object-oriented style, others were a collection of scripts that had to be executed in the right order, and others were collections of function definitions and execution code in single Jupyter Notebooks. Across all four components, there were about 12'000 lines of code in 60 files.

External documentation for MESMER-M, MESMER-M-TP and MESMER-X was limited to the model description papers (Nath et al., 2022; Schöngart et al., 2024, Quillcaille et al, 2002, 2023), whereas MESMER already had a documentation webpage with an API reference. Around 60 % of public function and class definitions were documented with docstrings in the legacy code. The code quality of the legacy components was mixed, scoring pylint scores (Goodger and Rossum, 2011) between 0 and 8.03, were 0 is the smallest (worst) score possible and 10 the highest (best).

The comprehensibility of the software was further hampered by a rather in-transparent data structure consisting of nested dictionaries, where data was stored as plain numpy arrays (Harris et al., 2020). To navigate the data one had to know the dictionary structure, the appropriate keys, and data dimensions. Lastly, MESMER-M, MESMER-M-TP and MESMER-X had no testing suite, ensuring correctness and stability of the Code. As of the release of MESMER v0.8.3 in Hauser et al, 2021, MESMER had a test coverage of around 78 %.

Comment 3 - Performance Assessment

Performance assessment (Section 5) was mainly focused on runtime. It would be helpful for the paper to also include some visualized plots showing the reproduction of old results or even improvements over previous results. For example, in Section 4.3, Switch to a cyclo-stationary AR(1) process for monthly emulations, you state that the cyclo-stationary AR(1) represents variability better than the AR(1) process. However, the corresponding plot is not shown.

Response

We added the plot showing the comparison variability estimations in the Appendix.

Changes to the Manuscript

We added the plot showing the comparison variability estimations in the Appendix.

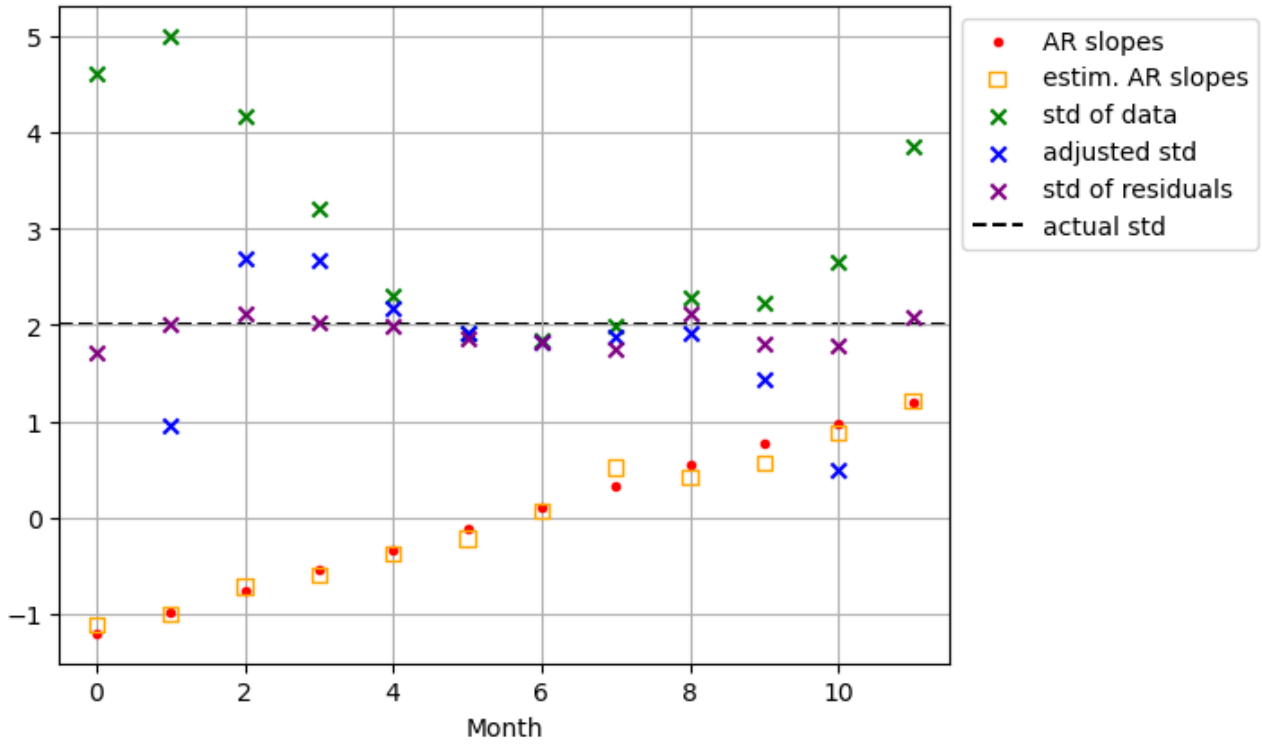

Fig A1: Comparison of methods to estimate the driving noise variance of data generated by a cyclo-stationary AR(1) process. Data was generated for 12 months using AR slopes indicated by red dots and a standard deviation (std) of 2 (actual std - gray stippled line). Yellow squares mark the estimated AR slopes retrieved by MESMER's fitting algorithm. Green crosses indicate the standard deviation estimated from the raw monthly data, blue crosses the one estimated by adjusting the standard deviation with the retrieved AR parameters, and purple crosses the one estimated on the data residuals of the AR process.

Specific Comments

Section 1: Introduction

1. Line 15: Please provide citation(s)

Section 3: Design of MESMER v1

2. Line 102: Please check the citation. Should it be Anzt et al., 2021?
3. Lines 112-113: Is mesmer.proba the same as mesmer.distrib in Figure 2? If so, could you please make it consistent?
4. Lines118-119: For data handling functionality, I don't seem to see mesmer.anomaly.calc_anomaly in Fig. 2. (mesmer),
5. Lines 121-124: Making this workflow for calibration and emulation creation readily available would already benefit users and improve reproducibility. Is this workflow already available?
6. Lines 154-156: Please provide citation.
7. Line 206: Should this be 238 instead of 269 as in Table 2?
8. Line 276-278: " … We were able to fix these issues by switching to more precise routines", This line is unclear to me. Which routines were used to address the numerical stability problem?

Section 5: Performance assessment

9. There are several instances where runtimes are mentioned as referring to Table 3, but Table 3 is not actually referenced. For example, in Lines 367–368 and 390–391. Please revise these references to make them clear
10. Line 355: Should this be 3 minutes rather than 4 minutes?
11. Line 400: Should this be 30 seconds rather than 1 minute?

Response

1. We added the citation Tebaldi, C., Selin, N., Ferrari, R., and Flierl, G.: Emulators of Climate Model Output, Annual Review of Environment and Resources, 50, 709–737, https://doi.org/10.1146/annurev-environ-012125-085838, 2025.
2. Yes, thank you, we corrected all mentions of the wrong citation.
3. Yes, thank you, we corrected all instances of mesmer.proba to mesmer.distrib in the text (the figure was correct and remains unchanged).
4. It is at the very bottom, in the middle in Figure 2.
5. This workflow is demonstrated in the MESMER tutorials. We refrained from adding high level functions that execute these workflows in one go, to keep flexibility and modularity of the software high. We added this text in line 124: *This workflow is demonstrated in the tutorials delivered with MESMER and on the documentation webpage.*

6. We added the citation Rew, R. and Davis, G.: NetCDF: an interface for scientific data access, IEEE Computer Graphics and Applications, 10, 76–82, https://doi.org/10.1109/38.56302, 1990.
7. Thanks for spotting this, we adjusted the number in the text.
8. We added the following text:
   We were able to fix these issues by switching to more precise numpy routines in the computation of the transformed data (namely numpy.expm1(x) and numpy.log1p(x) that yield higher precision for small x than numpy.exp(x)-1 and numpy.log(1+x), cf. https://numpy.org/doc/stable/reference/generated/numpy.expm1.html, last accessed: 12.3.2026).
9. We added the missing references in the text.
10. Yes, thank you, we adjusted the number in the text.
11. Yes, thank you, we adjusted the number in the text.

Changes to the manuscript

1. Line 16 in the latexdiff file: (Tebaldi et al. 2025) and corresponding reference in the Bibliography
2. Lines 123 and 264 in the latexdiff file: changed Loewe at al.,2021 to Anzt et al., 2021 and adjusted the corresponding reference in the bibliography
3. Lines 134, 136 and 137 in the latexdiff file: replaced mesmer.proba with mesmer.distrib
4. No changes
5. Line 146 in the latexdiff file: "This ~~restructuring is workflow~~ is demonstrated in the tutorials delivered with MESMER and on the documentation webpage. The restructured workflow is more in with…"
6. Line 154 in the latexdiff file: "It adopts Unidata's self-describing Common Data Model on which the network Common Data Form (netCDF) (Rew and Davis, 1990) is built."
7. Line 233 in the latexdiff file: "On average, source code files in MESMER v1 have ~~269~~ 248 lines of code…" The correct number changed from 238 as mentioned in the comment to 248 due to ongoing developments in the code base. The number now aligns with Table 2.
8. Lines 305-308 in the latexdiff file: "We were able to fix these issues by switching to more precise numpy routines in the computation of the transformed data (namely numpy.expm1(x) and numpy.log1p(x) that yield higher precision for small x than numpy.exp(x) – 1 and numpy.log(1+x), cf. https://numpy.org/doc/stable/reference/generated/numpy.expm1.html, last accessed: 12.3.20206).
9. Added reference to Table 3 in Line 398, 406, 422, 424, and 433 in the latexdiff file.
10. Line 385 in the latexdiff file: "…and ~~4~~ 3 minutes for the 24-member ensemble (Tab.3)."
11. Line 431 in the latexdiff file: "…takes about ~~one minute~~ 30 seconds,…"

Moreover, we realised another incoherence in the text with Table 3 that we corrected in Lines 424-425: "Drawing monthly samples for 100 realizations using the MESMER-M setup takes about ~~3 minutes~~ 2 minutes (Tab.3). This is about ~~9~~ 9-10 times slower than…"

Other questions

Is there a containerized solution for the new MESMER, such as a Docker container, to enable running it on different computers and facilitate reproducibility?

Response

No, we do not provide a Docker container, but we do test on multiple operating systems and python versions and provide detailed documentation on our dependencies.

Referee Comments 2

Comment 0

The authors present a consolidated version of MESMER, a fast climate model emulator based on a previously published statistical methodology. This is a well-written technical contribution that I enjoyed reading. I commend the effort of the team in making the code more accessible, with a focus on software engineering aspects that (as the authors also note) are often overlooked in research code. I also appreciate the effort of bringing the different flavors of MESMER together, which I think could prove useful in making the emulator more widely used by a larger community of researchers and/or stakeholders that are hungry for climate information. I have a few minor comments on the paper that could be worth addressing, and some food for thought for future releases of MESMER given the current state of the model.

Response

Thank you for the positive evaluation of our study and the helpful comments. We address the comments made below and explain the modifications we made to the study.

Minor Comments to the authors

1. L56: “land surface annual mean temperatures” – I’m assuming this is actually surface air temperature over land (tas in CMIP6 jargon), and not the actual land surface temperature. Worth rephrasing for clarity?
2. On a similar note (L343): is there a reason why you chose to exclude the ocean from your emulation? It seems that the main audience of the code is researchers, and I can think of a lot of research problems where the ocean response matters significantly (evaporation, hydrological cycle changes, low cloud feedbacks and boundary layer stability, SST pattern effect, dominant modes of internal variability [ENSO/NAO/…], ocean heat content and TOA radiation,....) . Even for general stakeholders, things like marine heat waves are becoming more prominent and relevant for impact assessments, and knowing the ocean response seems quite key. Maybe worth including (not necessarily in this paper, but in the future)? It would probably even simplify the code to some extent as no masking would be necessary.

3. Figure 4 caption and elsewhere: I just wanted to flag the possibility for any web link to be broken in the near future (few years timescale), whereas the paper is usually thought to be a much longer-term stand-alone contribution that will likely outlive the current web links (e.g. if you decide to change domain, not maintain mesmer anymore, or other possible unforeseen circumstances the links will be broken with no real possibility of updating them, since the paper after publishing is usually not updated anymore). Not necessarily suggesting to remove links (I see why they could be useful), but something to think about. I'm flagging this because I've recently read a few papers from the 2010s with links embedded in them that are now broken and not really useful anymore.
4. L383: can you be a little more precise on what you mean by "annual maximum temperature"? i.e., what time resolution is used to calculate the maximum (for example, is that the annual maximum monthly temperature, the annual maximum daily temperature or the annual maximum hourly temperature). Might be worth using CMIP6 jargon if it is defined in CMIP6 and it is not a postprocessed variable.
5. I tried to pip install the mesmer-emulator package and run the "Emulating near surface temperature on land with MESMER" tutorial, but it looks like the mesmer.example_data file was not included in the pip install (I believe along all the other *.py files in the mesmer/mesmer directory, e.g. anomaly.py, etc). I am using an Ubuntu machine and just followed the installation commands in the doc page, i.e. ran python -m pip install mesmer-emulator[complete] in a virtual environment. In my experience, a research software package becomes quite successful if the user experience is frictionless, meaning that it works out-of-the-box without much debugging to be done on the user side. Small details like this one might discourage users to keep digging into mesmer, which would undermine lots of great work that could be really useful to a lot of people. Just a suggestion to pay attention to these details and extensively test with different people, machines, …
6. On a similar note to #5, I wonder if it could be useful to develop an out-of-the-box version of mesmer (sort of like an old-fashioned executable, or maybe some high-level wrappers) that produces some basic output based on a few input parameters (e.g. N realizations of gridded tas for a specified experiment and model, where N, model, experiment are input parameters). In other words, I found the tutorials a bit too detailed (there is still a lot of processing, data handling, …) for the casual user that just wants to try the model out. Or perhaps there could be a range of tutorials, from a really simple one for the casual user, to the more processing/data handling ones that could be geared towards developers and people that want to dig more into mesmer. Not necessarily for this paper but it's something that might be worth thinking about for the future if the goal is to make mesmer a widely used emulator in the community.

Response

1. Yes, thank you. We changed the instances of "surface temperature" to "surface air temperature" and specified "annual mean of daily land surface air temperature ("tas")" in section 2.1.

2. Thank you for this question. We agree that the ocean response is vital to the future climate. The impact of the ocean response on the land is still included in our emulations, since we train on fully coupled ESM output. MESMER applications to date were focused on terrestrial climate impacts (e.g. Schöngart et al. 2025). It should be possible to fit and emulate ocean grid-cells with the MESMER code (but this is not something that we tested). However, more grid-cells will lead to a more severely rank deficient covariance matrix, requiring stronger regularisation or making the use of multiple ensemble members for the calibration even more important. We also want to note that we tested the behaviour of MESMER for strongly non-linear climate responses, like for example AMOC tipping.
3. Thank you for flagging this. We removed the web links in the caption of Figure 4, as they also impeded readability. As for other weblinks in the manuscript, we decided to leave them as they are either tightly linked to the content of the paper, like the github page or readthedocs page, or are the only references were no doi exists, like the numpy documentation or the GNU license.
4. Yes, we specified "annual maximum of daily maximum land surface air temperature ("txx")" in Section 2.1., but still refer to it as annual maximum temperature elsewhere for improved readability.
5. Thank you for testing the MESMER software! We updated MESMER to automatically download the tutorial data when executing the tutorials (https://github.com/MESMER-group/mesmer/pull/846).
6. For MESMERv1, the focus was primarily on improving the software so that it is a good starting point for further developments. We wanted to make the software flexible for users to extend and apply it to their own project by having understandable documentation and in-depth examples. However, we agree that there is a use case for out-of-the-box emulations, which we also want to facilitate, e.g. by providing pre-calibrated parameters. Our envisioned future for easy emulation is a light weight web-interface for MESMER, where a limited set of variables can be emulated at the click of a button. Thank you for your feedback! It is very useful for us to see that there is interest for such an application!

Changes to the Manuscript

1. Lines 49-50 in the latexdiff file: "~~land surface annual mean temperature~~ annual mean of daily land surface air temperature ("tas")" Moreover, we added "air" to "surface temperature" in lines 60, 167, 367, 368, and 384 of the latexdiff file.
2. No changes
3. Caption Figure 4: removed link, "the landing page on readthedocs (a),"
4. Lines 54-55 in the latexdiff file: "~~maximum annual temperature~~ annual maximum of daily maximum land surface air temperature ("txx")".
5. Updating the code to include automatic downloads of the example data led to changes in the most common code standard violations in MESMER. We therefore adjusted the corresponding text in Lines 245-248 in the latexdiff file: "The most common violations of the standards in MESMER are: (1) lines with more than 100 characters (mostly example-data file hashes), (2) functions having more than 5 arguments, which is sometimes

unavoidable, ~~(2)~~ and (3) missing module docstrings, as we do not document modules but only functions, classes and methods ~~,and (3) left over TODO comments in the code, these are kept for future developments and not urgent~~ (not shown)."

6. No changes

Additional changes to the manuscript

Some of the numbers in Table 2 changed slightly, due to ongoing developments in the MESMER Code base, see https://mesmer-emulator.readthedocs.io/en/latest/changelog.html#v1-0-0-unreleased and screenshot of latexdiff of Table 2 in the manuscript, below. The corresponding information in the text has been updated as well. These changes do not change any conclusions of the paper.

Target	Metric	Value
Modularity	Average number of lines of code per object	~~38~~ 37
	Number of lines of code per file (avg [min, max])	~~238~~ 248 [7, 1219]
Code quality	Pylint score	~~8.74~~ 8.64
Data structure	Numpy arrays to xarray data structure	Completed
Documentation	% of public API documented	~~100~~99 %
	% of private API documented	53 %
Testing	% of lines covered by tests	99 %

Ensuing changes in lines 231-255 in the latex diff file:
"Code is modular if one unit of code (e.g. a function, method or class) is only dedicated to one task (Trisovic et al., 2022; Nyenah et al., 2024). Thus, it is reasonable to assume that modular code units are short. On average, source code files in MESMER v1 have ~~269~~ 248 lines of code (Table 2). […] The longest source code file in MESMER v1 has 1219 lines. While this file exceeds the recommendation the second largest file has ~~629~~ 642 lines. In addition to lines of code per file, we also assessed number of lines per object definition, i.e. per defined function, class or method. In total, the MESMER v1 code base about ~~8.000~~ 8.200 lines of code in total and ~~214~~ 219 defined objects, making for an average of about ~~38~~ 37 lines of code per definition (Table 2), which we argue is reasonably short to understand. Furthermore, we employ the pylint score, to analyze the adherence of the MESMER v1 code to coding standards (Goodger and Rossum, 2011), as a metric of how comprehensible the MESMER v1 source code is. The score liesbetween 0 and 10 with 10 being perfect compliance. A score above 6 is considered satisfying (Molnar et al., 2020; Nyenah et al., 2024). The refactored code base reaches a score of ~~8.74~~ 8.64 (Table 2). […] MESMER v1 provides documentation for ~~100~~ 99 % of its public API and 53 % of private definitions (Table 2)."

Moreover, we released a new set of persisting calibrated parameters for emulations which we incorporated in section 3.3 (Lines 180 to 189 in the latex diff file) and the Outlook:
“Together with MESMER v1 we publish pre-calibrated parameters for the emulation of ~~annual mean surface temperature following the approach of the original MESMER (Beusch et al., 2020b) on Zenodo (Bauer et al., 2025)~~ multiple variables (Quilcaille et al., 2026): annual mean air temperature, monthly mean air temperature, annual maximum air temperature, annual average of the soil moisture, annual minimum of the monthly average soil moisture, annual maximum of the Fire Weather Index, seasonal average of the Fire Weather Index, annual number of days with extreme fire weather, length of the fire season. For information on the Fire Weather Index please refer to Quilcaille et al (2023a). ~~These~~ The parameters are calibrated for 58 CMIP6 models (Eyring et al., 2016) post-processed according to Brunner et al. (2020), using all available initial condition members of SSP1-1.9, SSP1-2.6, SSP2-4.5, SSP3-7.0, SSP4-6.0, SSP5-8.5, and SSP5-3.4-OS for each model. We calibrate using only global mean surface air temperature as a predictor. ~~This set of parameters~~ The pre-calibrated parameters allow~~s~~ users who are ~~only~~ interested in emulations of ~~annual mean surface temperature~~ the listed variables to omit the calibration step.”

Lines 441-443 in the latex diff file: For the first time, we also provide pre-calibrated parameters for multiple climate variables (annual and monthly mean temperature, annual maximum temperature, soil moisture and several Fire Weather related indicators) to enable users to skip the manual calibration process (Quilcaille et al., 2026).

During review period a new MESMER application was developed which is not yet integrated in the code base. We cite the according preprint in the Outlook (Line 470-471 of the latexdiff file):
“Immediate further developments include the integration of MESMER-M-TP and a novel extension of MESMER-X for extreme precipitation (Pierini et al., 2025) into the code base.”

Lastly, we added the proper references to the Code & Data Availability Section and updated the dois for the new versions of the pre-calibrated parameters and analysis scripts.