The software architecture of climate models: a graphical comparison of CMIP5 and EMICAR5 configurations

By **K. Alexander and S. M. Easterbrook**

Response to Reviewers

This document is colour-coded as follows:

- Comments by reviewers are in **blue**.
- Our responses are in **black**.
- Blocks of text we have added to the manuscript are in **red**.

Anonymous Referee #2

Received and published: 21 February 2015

Summary:
The authors analyze the source code of several climate models and determine their structure, i.e. how components are separated and interact from a software point of view. They visualize these structures in simple diagrams. They argue that these diagrams offer insights into the similarities and differences between models.

Recommendation:
This is an interesting and novel idea to look at code structures, a well written manuscript, and I recommend publication with minor changes in the discussion and implication sections. I'm still unsure whether we can learn anything about such analysis from a climate point of view (see details below), and how much this analysis affects how people think or work on models, and the authors have not demonstrated that. The discussion should be sharpened here. But the question of how such models are built, documented, tuned, how information is shared between modeling groups, and therefore how ensembles of models should be interpreted, is a very important one. Few people have looked at those broader questions, and it is great to see contributions from the software engineering community in this field. I am sure this manuscript will stimulate discussions, and in the long run help people to understand both how models are built, and how to interpret the zoo of models out there.

Specific comments:
(2.1) Abstract, page 352, line 11: "These diagrams offer insights into the similarities and differences between models, and have the potential to be useful tools for communication between scientists, scientific institutions, and the public.": I would argue the diagrams offer insight in the software structure, but not in the behaviour of the model. Two models can have similar structures but behave very differently, in fact even the same code with perturbed parameters can. In the end all of those models describe the same physical system, and whether a sea ice model is a subcomponent of the ocean or a separate component called from a coupler should not make any difference to the simulation of the climate, if things are done properly. The results from a model should be determined by the set of equations used (plus some parameterizations and numerical assumptions of course), and should be largely independent from the software framework, or the language used.

Model architecture may not have a direct impact on scientific output, but it can influence development pathways (e.g. highly modularized code is easier to edit and/or swap with another component) which then has a second-order effect on which processes get incorporated into the model, the parameterizations used, and hence ultimately the output. Architecture and past development pathways are tightly coupled. Also (as for Referee #1) our paper is not just about “the quality of science that we get out of an Earth system model”, we are also interested in the software architecture that enables efficient development of the science code.

We have modified the wording in the abstract to clarify this:
These diagrams offer insights into the similarities and differences in structure between climate models, and have the potential to be useful tools for communication between scientists, scientific institutions, and the public.

(2.2) The implication of this view is that such visual representations may be a useful communication tool for those who build and use climate models, but not necessarily for the public, and apart from the size of the components we cannot infer much (if anything) about how the model will behave, or how similar it will be to reality, or to other models. I do think the information provided here is valuable, but it is limited to software, and largely irrelevant for climate. It would be appropriate to state that clearly I think, unless the authors convince me that there is more value that I have missed so far.

We argue that these diagrams are useful to the public, who might otherwise picture climate models as black boxes that magically spit out climatology, and not stop to think about how the models are set up and the engineering challenges involved. We have anecdotal evidence that even experienced policymakers engaged in climate negotiations tend to think of climate models as fancy ways of computing linear regression over climatic trends. While our diagrams alone cannot correct such misperceptions, we believe that a visual representation of the architecture, along with some sense of the complexity of the code, can go a long way in helping explain to a lay audience what really goes into a climate model.

(2.3) In this case it would be good to see some specific examples and applications where such representations have been used, or have influenced something, or where people could infer climatic behaviour from the structure of the software.

We have a growing body of informal evidence of the value of our diagrams, but do not consider it suitable for inclusion in the current paper without a more rigorous empirical evaluation. Please see our more detailed response to point (1.14) made by Referee #1.

(2.4) Page 353, line 1: “We argue that these differences in module size offer a reasonable proxy for scientific complexity of each component”: It would be useful to define complexity early here (it comes in a parenthesis statement on page 361 but not clearly). The statement suggests that by complexity the authors mean “amount of stuff”, i.e., number of processes, or degree of detail considered. One could also interpret complexity by how rich the results are in terms of behavior. The equations of motion or a Lorentz system could be described as complex in terms of their behaviour, even though they don’t need many lines of code. Increasing the resolution of a model could massively increase the complexity in terms of the simulated patterns or structures of clouds, fronts, storms, while keeping the exact same code.

We have added wording in the introduction to clarify what we mean by complexity, and we have also added wording to section 4.3 to emphasize that we don’t include resolution of the model in this definition. The relevant sentence of the introduction now reads:

We argue that these differences in module size offer a reasonable proxy for the *scientific complexity* of each component, by which we mean the number, variety and sophistication of the set of geophysical processes included in the simulation with respect to a given part of the Earth system.

The relevant wording in section 4.3 is:

These observations allow us to treat line count as a proxy for scientific complexity, by which we mean the number and sophistication of physical processes represented in the model. Over the development history of a climate model, the line count tends to grow linearly. For example, Easterbrook & Johns (2009) showed that the UK Met Office model grew steadily from 100,000 lines of code in 1993 to nearly 1 million by 2008. The bulk of this code growth is due to addition of new geophysical processes to the model, and an increase in sophistication of the processes that are already incorporated. Note that we exclude changes in resolution of the model from our definition of scientific complexity, as changes to the grid size and timestep don't necessarily entail addition of new code.

(2.5) Page 361, line 25: the land component is not just about carbon cycle, but also about land surface processes, vegetation, and hydrology.

We have updated this sentence as follows:

In other models, particularly CESM (Figure 1) and IPSL (Figure 6), the land component is relatively substantial (although not the largest component in the model); this may reflect the growing interest by the modelling community in terrestrial carbon cycle feedbacks, land surface processes, vegetation, and hydrology.

2.6) Page 364, line 16: I'm not sure the interpretation of this factor of 20 is appropriate and useful. One might compare lines of code between different models of the same class (e.g. AOGCMs) but in a hierarchy of models the lines of code will obviously differ. An energy balance model can be coded with just handful of lines, and is also a climate model in some sense, but of course it has a very different purpose.

We have updated the text to include this measurement restricted to the six GCMs (factor of 7), and to clarify that the factor of 20 requires comparing a GCM to an EMIC. We have chosen to retain the factor of 20 statement because we believe it is an interesting and useful illustration of how different GCMs and EMICs are in terms of total complexity, and should help a non-expert reader understand the distinction between these classes of model:

Finally, climate models vary widely in complexity, with the total line count varying by a factor of 20 between the largest GCM and the smallest EMIC we analyse (Figure 9). Even when restricting this comparison to the six GCMs, there is still a factor of 7 variation in total line count.

5) Page 364, line 17ff: I like the comparison here but would add a few things. The amount of expertise may determine the amount of development of individual code, but does not imply similarity or differences to other codes a priori. In practice it may be the case, but it's a hypothesis (which is hard to test, and I'm not asking for a test).

This is true. We have updated the wording in this paragraph to indicate this is a hypothesis. Please see our response to reviewer 1, point (1.13).

Second, it would be good to emphasize that these two approaches take very different but complementary approaches. One looks at the structure of the code, how the model is built, and the other looks only at the model output, without knowing anything about the code.

We have added a sentence to the conclusions to emphasize this point:

Furthermore, the two analyses are complementary: while our analysis looks at model code without considering its outputs, Masson and Knutti (2011) and Knutti et al. (2013) analyze model outputs without looking at the code.

Finally, there is an update on the model clustering for CMIP5 by Knutti et al., GRL 2013, doi:10.1002/grl.50256.

We have added references to this publication wherever we cite the original clustering paper by Masson and Knutti, 2011.

We would like to thank Referee #2 for this thoughtful review, which led to direct improvements in our manuscript.