Review of "sympl and climt — Towards a flexible framework for building model hierarchies in Python"

Manuscript authors: Joy Merwin Monteiro, Jeremy McGibbon, Rodrigo Caballero

Summary of manuscript

The authors introduce the `sympl` and `climt` open-source python packages for creating models of Earth's climate (or those of other planets) and/or its subcomponents. `sympl` is a general purpose tool for formally linking and running these models, while `climt` is a specific instance of a model built using `sympl`.

Summary of my review

(Note that I deliberately did not review the source code of either package nor have I attempted to use either package, and I only briefly skimmed the documentation. I thought this would be useful for the sake of providing a review, since I suspect most readers will be in the same position.)

These kind of open-source, carefully designed, richly documented simulation and analysis tools are sorely needed in climate science and related fields, and I sorely hope that one day using them becomes standard operating procedure. I found especially compelling the prospect of changing a single column model to a full dynamical model with only a few lines of additional code.

However, my initial excitement quickly faded while reading as I found it very difficult to follow the manuscript. Still at this point, I am confused about some fundamental aspects of the packages. I think this could be addressed through the use of more concrete examples and some reorganization.

Therefore, though I look forward to this manuscript being published eventually, I recommend major revisions at present. My remaining comments are below.

Signed,
Spencer Hill
UCLA and Caltech

Major Comments

Lack of concrete examples

Most of the manuscript describes the packages in abstract terms, in such a way that I really struggled to understand what was meant much of the time. The concrete examples that are given are mostly not until the latter part of the manuscript, by which point I was already confused. I think it would be useful to move these earlier. This may require some considerable reorganization of the manuscript, but I think that would helpful. In general it's much easier for the reader to get a concrete example first, and then use that to discuss a more general principle.

Alternatively (or in conjunction with the existing examples), another potentially useful approach would be to create a toy example or two (or, if more, as few as possible) that are continually referred back to as concrete examples when more abstract issues are presented. At least one should use `sympl` without `climt`, while at least one should use `climt`.

For example, in Section 5, as described the `Prognostic` and `Implicit` types are so similar that it's not clear why they are treated separately. A specific example of each type would make that much clearer. And if the same examples were used repeatedly or gradually built out throughout the manuscript, it would facilitate the reader making connections across sections.

Scope of `climt` not clear

It would be easy to understand if `climt` was simply a particular model that was generated using the `sympl` framework. But instead, as I understand it, `climt` is itself a framework, albeit a more concrete one in which physical processes are explicitly represented, from which models can be generated. That may well be a useful approach (I suspect it is), but for the purposes of this manuscript, it makes understanding the packages more confusing.

For this reason, I reiterate the comment above that introducing a minimal, concrete example of a model developed using `sympl`, and separately one developed using `climt`, would be valuable.

Also, given that `climt` has been around for a long time, but previously in a very different guise (i.e. Rodrigo's `CliMT-legacy` Github repo), it would be worth noting this history somewhere.

Platform independence

The manuscript referred primarily to computing environments akin to institutional clusters or supercomputers, i.e. those with multiple processors (e.g. Section 3.3). But, especially for the simple models being espoused in the introduction, laptops and tablets can be sufficient. Separately, there is the coming era of web-based computation (e.g. Amazon Web Services; the Pangeo project). How do `sympl` and `climt` handle these different environments, what configuration etc. is required to go from one to another, etc.? Some discussion of this would be useful.

Line-by-line and minor comments

Page 1

L1 Consider replacing "represent" with a stronger word like "are"

L4 and elsewhere Oxford comma; I like it (i.e. I would put a comma after "inter-operable", but it is ultimately personal preference.

L4 and elsewhere hyphen in "fine-grained"

L7-11 I'm not sure listing the different data structures is useful in the abstract. Consider a more compact, higher-level description instead, more like the subsequent paragraph.

L8 Implicit should be `code-formatted` like the other components.

L13 I don't think that Cython is well-known enough to be referenced in the abstract without introduction. Consider just omitting this clause.

L13 The meaning of "aims" is ambiguous. Does that refer to planned future work, or is it the package's philosophy/design principle?

L14 I found "trade-off" confusing here. Is this a comparison amongst the "different APIs"? Or between the APIs collectively and something else? I think it's the former, but it still could be made clearer.

L14 and elsewhere consider replacing "which" with "that"

L16 "is" -> "are" and needs another verb, e.g. "are performed"

L17 hyphen in "Python-based"

L17-19 I don't really understand this sentence. What is the connection between modularity and "using online data analysis"? And what does "using online data analysis" even mean? It just feels like a weak sentence to end on.

L21 "mutually" redundant; consider omitting

L22 comma after "chemosphere)"

Page 2

L2 Omit "slightly"; it is sometimes the (fortunate) case that a very simple model very clearly elucidates the behavior of a much more complex one.

L5-6 Part of the emphasis of the Jeevanjee et al paper was on the fact that there is no single hierarchy that is universally applicable. As such, I recommend against referring to "the" modeling hierarchy; consider "the specific model hierarchy suiting their needs." This is also in line with the packages' goal of enabling users to plug-and-play components as they see fit.

L7 Consider "...this hierarchy remains a challenge..."

L12-14 I am unfamiliar with ESMF, and it isn't clear from the text how it provides the fine grained control referenced.

L15 hypen in "framework-based"

L15 I would start a new paragraph here.

L15-17 `sympl` and `climt` are being referred to before they have been introduced in the main body.

L17-20 "we believe ... our design to be unique" is too vague to be convincing. What are the salient differences?

L22-27 I would put this before the preceding sentences that refer to sympl and climt. As written, it jumps back and forth from motivation (modularity) to the packages, back to another motivation (reproducibility), back to the packages again.

L30-32 If you choose to preview the manuscript in this way (which I support), it is better to be explicit about what is in each section. E.g. "...frameworks have to solve (Section 2), ..."

Page 3

L15 the operate -> that operate

L24 Section title uses "taxonomy", but text here (and elsewhere) uses "ontology". I would pick one or the other to keep down the jargon.

L27 I don't understand this sentence, mainly because I don't know what is meant by "display". What does it mean to display different behaviors consistently?

L29-30 and subsequent example of where concrete examples would be extremely helpful

Page 4

L5-6 I don't understand what point is trying to be made here. Examples would help.

L10-19 Examples would help.

L21-24 Here's a paragraph where, thankfully, an example *is* utilized. But I don't understand the example given. What does it mean to "build a full model" in this context? Outputting radiative tendencies *requires* some treatment of radiative transfer, however simple or complicated. Does "interact" simply mean "output results"?

Page 5

L12-14 This feels out of place. The rest of the section is about general principles, not `sympl` and `climt`. Move to next section?

L16-24 I don't see the relevance of these autobiographical/historical details. Consider opening this section more directly, with a brief summary of the design decisions that you will subsequently detail in the rest of the section.

L23 "object orientation" -> "object oriented"

L22-30 Examples would help.

Figure 1

- Names don't match those in Section 3
- Why does "Sympl+CliMT" bracket on the left not extend to the runscript?
- It is possible to reduce the number of intersecting lines in the bottom half; as presented here it feels somewhat unfair (the more intersections, the more complicated it looks).

Page 7

L3 For xarray cite *Hoyer and Hamman* (2017)

L6-7 Consider instead "This exposes the powerful analysis capabilities of `xarray`, allowing users..."

L8-11 What is meant by low-level operations being "simple"? Simple for the user? What is the connection to memory layout? It also might help to put a paragraph break before this; it feels like a separate topic from the higher-level `xarray`-related stuff discussed above it.

L12-17 This recommendation for how users should use the API feels out of place in the "Design Decision" section... it would seem to be better placed in the next section

L26 Is `Diagnostic` used by `Prognostic` and `Implicit`? Since they both also refer to outputting diagnostics.

L27-29 This has the start of a useful example re: the CFL criterion, but it's not detailed enough to be useful. In other words, even with this example, I still can't tell why `ImplicitPrognostic` is needed in addition to the other two.

Page 8

L6-7 I don't understand this description of a `Wrapper`. Why are these needed in addition to the other components?

L24 Probably worth mentioning that this is important for simulations starting at year 0 or 1 (or really long simulations) for which the limited range of numpy datetime supported dates otherwise becomes a real pain.

Page 9

L3 Models don't have to have all three spatial dimensions, right?

L9 I would omit "to perform sensitivity experiments". There are cases (including some of my own) wherein varying planetary parameters like rotation rate is the *main* focus, not a sensitivity test.

Page 10

Fig. 3 A more descriptive caption would help. So would explicitly populating the `__init__` and `__call__` methods.

Page 12

Fig. 4 Panel labels and a legend would help a lot. Also, I know I keep saying this, but I just don't really understand a lot of this figure (apart from panel b). Take panel (a): what is the meaning of the arrows from Current State and Diagnostics converging at a cross and then joining into one? Why are the two Prognostic boxes in panel (c) in their own black outlined box?

L2-5 I think it would help to first state the main focus of `climt`, namely providing actual usable scientific components that can be used to generate physically meaningful models. Then mention the other peripheral aspects: configuration options, helper functions, additional attributes.

L7 "typically" is confusing; doesn't a spectral core step a model forward in spectral space by definition?

Page 13

Footnote 6 Consider omitting; doesn't seem important.

L12 Why separate dimensions into the "core" ones and this `extra_dimensions` category?

L20 What does "by hand" mean?

L24 Is Section 7 supposed to be Section 6.3?

L28-29 incomplete sentence

Page 14

all citations and/or URLs for these different projects/components would be nice (e.g. CDMIP, OpenMP). For the insolation, grid scale condensation, and ice modules, did you code these up from scratch? If yes this should be noted. If not their original source/predecessor should be cited.

Page 15

L2 and elsewhere in paper capitalization typo

L10 regression tests are mentioned; are there unit tests also?

Figure 5

L1 PEP8: lines 1-2 would fit on a single line

L37 60000 is a magic number; define as a constant i.e. `NUM_TIMESTEPS = 60000`

L41 Don't understand what "This order is arbitrary" means in this context. Could have run Implicit first and then TimeStepper? As written, it's counterintuitive to time step first and then call the physics.

L44 another magic number

L49 Feels weird to increment the time by hand like this, when everything else is so boxed into the `sympl` data structures. Why doesn't `TimeStepper` do this?

Caption What code was omitted? If at all possible include all the code, simplifying the example if necessary; otherwise it feels like a black box.

Page 17

- It would be really nice if the output in Figure 6 came directly from the model shown in Figure 5. Is that the case?

L8 a idealized -> an idealized

L8-10 Can you provide the runscripts for each? Potentially as supplmental material?

Page 18

L5-6 Don't understand this, and an instance c.f. major comment above that gives the impression that these tools aren't easily ported to non-HPC contexts

Page 19

L8 There was little mention in the paper of the documentation.

L19-21 Ending with this limitation doesn't do you any favors. Consider adding an additional sentence or two emphasizing that, despite this limitation, these tools are functional/valuabel/etc.

`climt` online documentation

- A more welcoming main page would be helpful; xarray's docs (`http://xarray.pydata.org/en/stable/`) are a great model to follow.

References

Hoyer, S., and J. Hamman (2017), Xarray: N-D labeled Arrays and Datasets in Python, *Journal of Open Research Software*, *5*(1), doi:10.5334/jors.148.