We thank the reviewer very much for going through this paper and making some great suggestions. We have hopefully addressed all of them and in doing so, have further improved the manuscript

Reviewer 1:
This paper presents a utility for controlling the execution and initial evaluation of an application (the ParFlow model) running in a (primarily) HPC framework. There are two levels to the framework described: the "run control framework" (or RCF) which itself utilises a more generic JUBE benchmark framework as a workflow engine. Essentially these provide a method of systematically defining, running and analysing some benchmarks - the authors also suggest it would be suitable for use for production simulations as well.

The paper is a heavily modified resubmission, and was previously titled: "Best practice regarding the three P's: profiling, portability and provenance when running HPC geoscientific applications".

The authors have made considerable efforts to respond to the previous reviews, and the paper is much improved, however there are still a number of issues. I think it could appear in GMD provided these issues are addressed to the satisfaction of the editor.

Most of my issues now reside in the front material, the body of the paper describing the use of the tool with paraview is much improved, and I am happy for that part to appear more or less as is.

Major issues

1. The goal of the paper is still not clear - but only because it is still obscured by what still feels like an excessive emphasis on motivation. I would ask the authors the following question: "If you read the abstract, what would you expect to find in the body of the paper?" Half the abstract is motivation, which seems wrong. Results of using this tool with ParaFlow are not even mentioned. Along the same lines, it is page 4 before the introduction gets round to telling us the bulk of the paper is about the RCF. These issues could relatively easily be addressed by reworking the abstract and either removing much of the existing introduction, or moving much of it to a motivation section immediately following.

We have removed some of the material from the introduction to shorten it and we've altered the abstract.

2. It is good to see there is now a discussion of other tools in section 2.1, but the material is not well connected, misses the point in a number of crucial ways, and (I would assert) wrong in some of the statements. There is no pre-existing taxonomy of tools in this space, and it would be unreasonable to expect the authors to have real experience with these tools, so getting the level of discussion right is not trivial, but

1. Page 5, line 9 mashes a description of JUBE into a description of other tools. At the very least this is a new paragraph.

We've now made this into a new paragraph.

2. It is not clear why all the emphasis on XML. All XML provides is a syntax (which is obviously useful) but the statement that cylc has "its own scripting" (line 15) is mixing action (scripting) with the syntax definition (XML). Cylc actually uses INI and Jinja2 for syntax, what is interesting about the differences is not whether one uses XML or INI, but what semantics exist in the configuration. What can they do?

This was trying to make the point that we've now already set up everything in xml and so we can easily switch to ecflow in that case so we've altered these lines to state this more explicitly.

3. The discussion about the platform constraints on submission belongs in its own paragraph (but I would ask why, with 2 hour limits, they can't run cylc - or any other tool - on a third system and simply poll through the login nodes using ssh tunnelling).

We would prefer to use something that runs natively on the machines we are using than having to set up a server to continuously run for the duration of our production job chains. Our production runs take 10 days of compute time (or more). We have tried to emphasize this point more.

3. No one expects such a comparison to assert that cylc or any other tool is better or worse than their tools, simply stating different capabilities is all that is really required. Clealy the JUBE RCF roadmap will differ from those tools, and it would be fine for them to describe and build other tools, even if they had the same or poorer functionalty - which is clearly not the case, there are some real advantages to describe here! The problem is that some of those advantages become clearer once the use case is fully described. It might be that this comparison could go in a paragraph preceding the conclusions, but whatever is done, all tools have strengths and weaknesses, this paragraph currently reads like a "defense" rather than a "comparison".

This also relates to the points below and above. We are trying to assert that we could use any of the tools mentioned- we chose JUBE as we had access to the developers and it is designed to run natively on the machines we use. The strengths in this case are that we do not have to run a separate server which continuously tunnels in over the duration of the run. We have tried to emphasize this but feel that listing all the strengths and weaknesses are beyond the scope of the paper as the focus is on the framework we built rather than the tools we use.

5. (Some of those advantages are actually in the semantics of what the configuration files are
set up to do, which in practice really means, "in the logic of the tools" ... not in the use of XML per se.)

Yes this is the point we were trying (unsuccessfully) to make. So we've tried to emphasize this- we've altered those lines and also altered the first sentence of 2.2.

4. I cannot find any assertion in Manubens-Gil 2016 that cylc is more complicated when building workflows. If that statement really exists, then fine, but otherwise I'd remove this sentence.

Quote: Cylc provides a cycling pattern that can be over just a sequence of integers or a powerful date-time cycle based on standard ISO 8601. It also uses the Jinja2 language to provide scripting features enhancing its workflow definition capabilities, although at the cost of a reduced readability. ecFlow's workflow definition language is not based on cycles, but it allows to define repetitions at job or family level. Moreover, it provides a Python API that makes the workflow definition easy, robust and powerful.

5. I don't understand the first sentence of 2.2. and the statement "merely tools for task submission" ... (particularly in the case of cylc, when ROSE is used with it). (I am not here trying to argue in favour of cylc in any way, but simply to point out that one cannot describe these other things without being accurate about them!).

We were trying to point out that it is really the RCF rather than the workflow engine that provides the useful infrastructure needed for complex geoscience applications. We have altered this sentence to try and reflect that.

3. It appears that the magic sauce that makes this RCF/JUBE framework so useful is really in the layout and structure that is described in Figure 2 and within the various parameters defined in XML and understood and actioned on by JUBE and the RCF. The authors have moved material from the appendix to the main body, and that is helpful, but currently it reads

like documentation, not an explanation of the functionality exposed. I think if the authors could find a better and more succinct way of explaining the functionality exposed by these configuration files the paper (and tools) would be vastly more interesting to prospective readers (and users).

1. E.g. page 22 states that "automatic archiving is performed", surely that is important functionality, and somehow configured ...

We're unsure what is meant by this point. We intended to describe the framework in the methods and then show the functionality in the case study.

Minor Issues

1. The list of ways of generating compute demand (bottom of page 1) is still somewhat idiosyncratic: bundling data assimilation in with ensemble members doesn't make sense to me ... (especially since data assimilation appears to be an important option for ParaFlow - see end of page 14).

Ensembles generate demand by needing to run several instances of the model at once. We have added this in.

2. The paragraph on page 2 beginning line 11 doesn't seem really relate to the topic of the paper. If I were one of the authors I'd be arguing to remove it. The key points are in the next paragraph (but as I Minor Issues said above, I can see an argument for removing much of this entire section).

We've removed a lot of this section based on the advice given above.

3. The paranthetical comment on line 10 page 3 "(see article acknowledgement)" is not obviously pointing at the previous reference (I initially looked for something in the acknoweldgements of this manuscript, rather than the previously referenced paper). In any case, it's not just the size of the team, it's the time they spent as well.

The acknowledgements referred to are in that citation. The people mentioned in the acknowledgements are part of the development team. However to avoid confusion we have removed that comment.

4. I can't really see the segue between the last two paragraphs of section 1, probably because the authors have not really made clear to me what distinction exists between a workflow engine and a run control framework. I think I understand the that this tool is something which makes it a specialised workflow engine ... (the same issue exists with the last sentence on page 4).

We tried to make the distinction clearer between a RCF and a workflow engine in the last paragraph of section 1.

The long paragraph which begins page 5 covers so many different things ... and introduces ParaFlow where generalised statements would be more appropriate.

We've shortened this paragraph and made it more general.

6. Page 12/13 Eaton et al describes CF, not CMOR, and is inappropriately positioned at the end of the sentence (by appearing there it appears to be applying somehow to ParaFlow, not CF).

We have included the additional reference for CMOR and moved the position of the citation.

7. The second sentence of section 3.5 could be improved ... it's a very difficult sentence to deconstruct :-), and I think the use of the word exascale doesn't add anything.

We have tried to improve this sentence.

##

We thank the reviewer very much for going through this paper and making some great suggestions. We have hopefully addressed all of them and in doing so, have further improved the manuscript

Reviewer 2:

P2 11 – p3 3: the references in this section are all relatively old and a little out of date (there is nothing beyond 2014 in a rapidly developing area). It would be useful to include some more recent references. For example, GP-GPU accelerators are now well beyond “10s of cores” (p 2 25) and their performance is currently in the 10s of TFLOP/s range (rather than 1 TFLOP/s, p2 24 - e.g. see the NVIDIA Titan – and that’s a year or two old now!).
Also, a major current trend towards developing exascale machines is the exploitation of FPGAs for acceleration of both applications and communication (e.g. the recently funded EU EuroEXA project, https://euroexa.eu/, among others).

We’ve shortened this paragraph and included more recent references as well as mentioning the FPGAs.

P4 9: please make it clear here that the ParFlow case study will focus on its execution on the Juqueen machine at JSC.

We’ve added this in.

P6 3: change to -> “The directory structure for the RCF run harness…”

We’ve made the change.

P9 12: it would be useful to more clearly link the section starting here on “health examination” to the follow up in Sec. 3.3. I suggest adding a subtitle (unnumbered) “Health Check Protocol” to make the link explicit.

We’ve added the subtitle.

P9 24/25: for completeness, define the symbols used in eqn 1. (i.e. T_1 and T_N) – other symbols in equations are defined.

We’ve defined the symbols used.

P13 Fig. 3 I spotted that coll_message_size does not appear in Appendix B!

Thanks for picking this up, we’ve added this in.