We thank the anonymous reviewer for taking the time to read this paper thoroughly and providing the authors with constructive and thoughtful feedback. Addressing this feedback has greatly improved the paper.

Addressing reviewer 2's concerns:

The paper presents the some of the challenges of geoscientific modelling on HPC resources, with emphasis on profiling and processing workflows. In order to address provenance, portability and profiling best practices, a run control framework (RCF) based on JUBE is described. The demonstration of the RCF is conducted in a weak scaling experiment in ParFlow, an integrated watershed model. Presentation of the tests results constitutes a notable - and interesting - part of the paper.

The paper is relatively easy to read in sections, but difficult to read as a whole. It covers several domains of expertise such as HPC, geoscientific modelling, software engineering, run harness and profiling. More work is required to present a coherent and uni-formly detailed experiment in its full context. The profiling element is also substantially more detailed than the others to the point of overshadowing them. Furthermore, profiling results could be much better linked and/or mapped to the very informative “health check” section. As for layout of the “health check”, typical diagnosis tools appended for each item does not convey the specific (overlapping or not) role or features of each software very well.

The suggestion to map the results back to the health check is a good one. We have also moved definitions in section 3 up to section 2 to the health check, and mapped each health check step back to the results presented. We have provided a more in depth description of the profiling tools used and their overlap. In addition we have detailed the function of the RCF in automating the health check workflow to try and provide a coherent experiment with which to demonstrate the advantages of using the RCF.

The authors list in section 3.4 the outcomes of the profiling case study. This entices the reader to believe that profiling will also advance provenance and portability. The article presents only a few elements of future work to support the full scope of the study, at least as the title describes it. It is also difficult for a reader to consider the tests results - the most detailed element of the paper - as sufficient proof of application of best practices in profiling, portability and provenance.

Yes we can see how this might cause some confusion. We have deleted Figure 9 and also the text to surrounding the NetCDF reader/writer to limit confusion for the reader. We have also clarified that the RCF is a best practice approach to profiling, portability and provenance by altering the title and also providing a better explanation of what the RCF is and how it is integrated with the workflow engine, JUBE. We have updated the title to: A run control framework to streamline profiling, porting and tuning, simulation runs and provenance tracking of geoscientific applications

The test case experimental design for ParFlow reads like a completely disjoint section to the rest of the paper. It is only very lightly linked with previous or subsequent sections, for instance on the motivations behind that particular test case and how it leverages the RCF, the workflows or the HPC resources. It is unclear looking at the profiling results what are the implications for the test case or to ParFlow itself. This section offers a great potential to present ParFlow software and models (graphically), to show alternative test configurations or to contrast with real-world scenarios.

Thanks for these suggestions. We have now included the motivations for using this particular test case in order to illuminate bottlenecks using the health check procedure and the reasons why a “real world” test case might obscure certain results such as load balance. We have also shifted the test case model description to section 3 for better continuity.

The paper mentions two other workflow engines (p.3 33) before rapidly shifting to JUBE (p.4 25). What are the advantages, limitations, similarities of JUBE compared to these other solutions? As is, considering that the results analysis of ParFlow provided by the RCF is a large (and interesting) part of the paper, a reader might be tempted to think that JUBE was selected first and pitted against other solutions a posteriori. It might be sufficient to cite major findings for the associated publication.

This was an oversight. We have now included a discussion of the relevant strengths and weaknesses of JUBE, ecFlow and cylc and have stated the reasons for choosing JUBE but have also made it clear that the RCF could be adapted for other workflow engines and they share a lot of the same functionality.

Address specific comments

Specific Comments
1. Some sentences are too long (p.1 8-10, p.3 8-10, p.17 6-10). The messages conveyed by these long sentences is important for the coherence of the whole.

Agreed- we have fixed these sentences.

2. Figure 9 could be changed to a textual list without loss of content. Is there an overlap or interrelation between those developments? Is there development to do outside the scope of ParFlow, for instance in the run harness or workflows? What does “towards exascale” refers to exactly (cite or describe)? At what scale is the system operating at right now? What are the hurdles from terascale/petascale onto exascale that the presented work will limit or remove?

We have followed your suggestion and removed Figure 9. From the profling results it can be seen that due to memory required being greater than memory available, at around 64000 cores, ParFlow as it stands is not approaching exascale. *Exascale* computing refers to computing systems capable of at least one exaFLOPS, or a billion billion calculations per second. ParFlow coupled with p4est as the parallel mesh handler, allows ParFlow to scale to the whole Juqueen machine (petascale). We have made this clearer in section 3 and introduced the exascale concept in the same section.

3. A very large part of the relevant information related to Figure 2 is in its legend. The reader might not care very much about the screen layout of the output. The reader might be interested in numerical values for each result on some occasions, but most values aren’t described or introduced previously in the article. Most of all, the reader will most probably be interested in the metrics themselves and how they relate to profiling - or to some extend to portability or provenance, if applicable.

This is a good idea. The reader would benefit from knowing what each metric is used for. We have tried to explain what each metric means and why it might be useful in table format in appendix A2 and thus reduced the length of the figure caption.

4. Section 3.1 is very short compared to others at this level, which diminishes its impact. It does not help with readability and flow. The section would benefit from an extension of concepts or reinforcement of links with other sections. It may also be merged elsewhere.

We have altered section 2 to include a more thorough description of the accuracy test we used to test compilation flags. We have expanded section 3.1 to include a reference to this test as well as a discussion of the relevant compilation flags used for the specific IBM XL and why we focused on the two specific flags as opposed to others available.

5. Some of the claims in the article are very lightly substantiated, insufficiently nuanced or lacking

details. For instance (p.1 16 and p.19), the author claims that RCF is less time consuming and more robust, but less/more than what? Than without use of an RCF? The article concludes by claiming a more efficient use of HPC resources that
was not clearly demonstrated; it reads like this is only implied because best practices were followed. If that is the case, it is suggested to better define and highlight these best practices.

Yes less time consuming than running by hand or developing a series of specialized scripts which only work for one model without integrated profiling tools. We have tried to make this clearer in all sections of the manuscript where this is mentioned and have updated the conclusions accordingly.

6. The paper also mention in a few places costs notions (“invested effort”, “paid off”, “cost in resources”) without providing any data or basic financial analysis. What was the approximate amount invested by articles cited? How much money or energy was saved? This comment is provided without diminishing the (substantial) presented work or assuming that this information is essential.

We are assuming you mean the references mentioned in page 3, 2nd paragraph and have updated this to include an example (Leutwyler et al. 2016) of speedup values (~3.6) and also the invested effort required (a team of developers)

7. The paper briefly mentions hybrid and heterogeneous architectures, but do not mention cloud computing. While a very different architecture than HPC, a reader might interested to see if any of the work presented can be applied in cloud computing environments (worfklows? packaged code? profiling? tools? models?). Commercial
and scientific offers in HPC-as-a-service might prove an interesting option for the RCF. Absence of cloud computing discussion is not seen here as a limitation of the article, only a potential topic of interest.

Thanks for pointing this out. Yes the RCF could be adapted for cloud computing or a web based interface. However this would be a substantial effort for a technology that is as yet severely limited by bandwidth such that it performs substantially worse than just using one HPC system. So we think this is out of scope for what is currently available to HPC users.

8. There are also almost no mention of standards, except a brief sentence on (p.10 3-5). A reader might expect that such large scale systems with claims on portability and provenance do indeed follow standards instead of reinventing the wheel.

A brief mention of these standards without explanation was because the authors assumed that the readers would be familiar what standards we follow (CF and CMOR Metadata Conventions) in the section on provenance tracking. We've now expanded this section to include a discussion to describe the CF conventions and their purpose. We have also now added to this section a discussion on how Irving 2016's minimum standard points 1-4 are covered by the RCF.

9. The code profiling section (p.6 6) makes it difficult to the reader to separate author contribution - by means of the RCF - to outputs from ScoreP and Scalasa. There is a long list of what software can “examine” those outputs. Why are those outputs compatible with all these software? Is it a standardized file format or structured data?

This is true. We have tried to make this clearer by explaining what ScoreP, Scalasca and Cube are in the text (rather than in the appendix) and to clarify what RCF does (collect and collate the performance metrics by parsing the various reports generated by the tools). There is a standard format for ScoreP and Scalasca output which Cube can parse and read.

10. Alinea Performance Reports and Intel Vectorization Advisor are present in each item of the health examination, but both software weren’t used. Still, they are recommended by the authors. Is the toolset of the experiment sufficient? What is the

additional insight offered by Alinea and Intel’s products that the other tools can’t?

The toolset used was determined by what was available on JUQUEEN. We've now explained this in section 2. There is a lot of overlap with different tools and we have now explained this in the same section. As to what tool one uses it depends on 1. what is available on a given machine and 2. personal preference. For example, Alinea products offer a nice way of presenting coarse-grained metrics for a novice user but do not offer more features than many other tools. And Intel vector advisor additionally can provide some more guidance than other software when it comes to identifying potential loops to be vectorized but this is not part of the initial health check guidelines and so these results are not mentioned.

11. Interoperability is largely undefined throughout the text. There is only a single mention of interoperability (p.10 6) for “extra” features. Interoperability between what and what exactly? Was interoperability a critieria either in the conception of the test, the run harness or the workflow?

What the authors meant by interoperability and did not explain very well, is that CMORized NetCDF files can be used on different architecture (big and little endian) and for different software (for use in various terrestrial systems software and visualization software). In the section we've added about CF conventions we have also mentioned how this facilitates interoperability and what we mean by interoperability in this instance.

12. The subsequent paragraph - a very long sentence - mention download and rerun. Results on reruns would be welcomed if possible, either on JUQUEEN or better, on other infrastructures.

Yes, the profiling results obtained are the result of running the benchmark described 10 times to get an average and make sure the benchmark is running as it should- i.e. there should not be a huge variance between results. And we have now added this explanation to section 3. The authors think that showing how this benchmark performs on other architecture is really beyond the scope of the paper as we are primarily discussing the RCF and how it aids portability, profiling and provenance tracking. Of course discussion of the portability aspect should include what other machines the RCF is running on (good suggestion) so we have mentioned that this RCF is also being run on Julia (a prototype KNL cluster at JSC), Jureca (a general purpose cluster at JSC) and Juropa3 (a prototype testbed system at JSC) in the description of the RCF.

13. The paper states that platform.xml can be easily extended or altered to include new systems. It is always easy to modify XML files, but not trivial to know what constitutes a valid modification and to successfully deploy it on other systems. Is there any tools to help a user, a developer, an administrator? Most discussion on portability revolves around XML files, compiler/linker flags and use of Python language. The paper concludes that the RCF using a workflow engine leads to code that can be ported easily. These conditions are important, but insufficient. A more thorough description of “environment preparation setup” might help a reader to better assess how close this particular run harness is compared to his own environment(s).The article would benefit from a better definition of portability. Ported from where to where? Any example of a second HPC infrastructure in your network? Precise what future work will advance portability. Other topics that could help a reader - this reviewer in particular - to assess portability could include virtual environments, software containers, software repositories and continuous integration frameworks.

This is a valid point. The RCF facilitates the environment set up through the use of loading modules, most if not all HPC systems contain this feature, thus this is one of the reasons why the RCF is portable- we have now mentioned this in section 2. Within the platform XML file, there are structs defined with standard parameters for run time arguments and compilation flags which can then be used for any HPC system with particular versions of compilers and profiling tools. We have added this to the description in section 2.3 (code portability).

14. Table 2 presents efficiencies measured during the weak scaling experiment. The authors states that more in-depth analysis is needed, but no strategy, best practices or future work is offered to the reader. Is this analysis to be conducted by a specialist, is it tool-assisted, what is the state of the art?

Good point. More in depth profiling would be much more effective together with performance analysis engineers. We have added this explanation and have put in an example of future work that we are currently conducting with the aid of performance analysis specialists: vectorization of individual loops.

15. There is an assumption that the directory structure (Figure B1) “allows for run time provenance tracking” and “such that the model can be rerun without using any other external tools”. The directory structure presented is most probably correct as a part of the RCF implementation. Still, it is unclear this is sufficient to insure provenance tracking or rerun. Is a tool set available to explore these directories and/or rerun models? Is it indexed in some form? Is there some other semantic information available that can be used?

What is meant is that whole simulation can be rerun as the code, the forcing data, the environment: modules loaded, including a list of dependencies their versions, all extraneous scripts including job submission scripts and a complete model description are bundled together in one output directory. If a user were to untar an output directory, they would be able to compile and rerun the experiment using the XML file contained in the directory, with JUBE, on the same machine and obtain the same results. We have tried to improve the description on page 3 and in section 3. Also we have mentioned that there are additional features beyond the users control such as different hardware when porting models and ways to ensure the scientific results remain the same.