Reply to Reviewer 1

General Remarks:

This manuscript presents results from a modified WAVEWATCH III code, which off-loads the spectral source terms to GPUs. The authors investigate the scaling and two different platforms, and validates the results against a CPU-only run.

This is a very timely, important and well written manuscript. I can recommend that it is accepted for publication after some minor changes. I also want to mention, that the way the problem is broken down and presented step by step made the manuscript easy to follow. Please find my comments and questions below:

We wish to thank the reviewer for their positive comments regarding the manuscript, particularly for the explicit acknowledgment of the writing structure. We also thank the reviewer for the insightful questions and suggestions raised. Detailed answers have been provided to the questions and revision has been done following the suggestions. Below are the original comments and the changes we have made in the manuscript to respond to the comments. Also, we have added a node-level baseline/comparison (using all the CPU cores and GPUs on a node) section to the manuscript. The baseline comparison was done only on the summit machine because KODIAK LANL has been decommissioned.

Specific Comments:

lines 24-25: "despite the growing literature on their in the simulation of weather and climate."

Seems to be missing a word

Yes, we have updated the manuscript to include the missing word. "despite the growing literature on their **importance** in the simulation of weather and climate."

line 71: I'm not sure what this is a reference to, but usually Komen et al. 1994 is used as a WAM reference?

We have corrected the reference. (WAVEWATCH III® Development Group, 2019)

line 86-88: These sentences are slightly confusing, since first we are talking about modules that calculate source terms (right hand of Eq. 1), and then we talk about discretizing (left hand side).

The sentence has been reorganized, and the discussion of discretization has been moved immediately following Eqn (1)

line 94: Can the relative computational intensivness change if we are using defferent propagation schemes. Can extremely small time steps alter/tip this balance? (Probably not, I guess.)

As shown in Fig. 1 (updated Fig. 4 in the manuscript), the source terms approximately take 82% out of 96% of the W3WAVEMD computational time. This means that the remaining computations in

W3WAVEMD (propagation scheme inclusive) ~14%. Thus, different propagation schemes won't alter the balance that much.

Moreover, with the unstructured grid configurations, we are limited to the lowest order scheme (CRD N-Scheme) because it is well-tested and more robust for realistic applications (Abdolali et al., 2020; Roland, 2008). The source term time step is dynamically adjusted based on local wave properties, but when running on a global domain, the effects of the time step dependencies are averaged out.

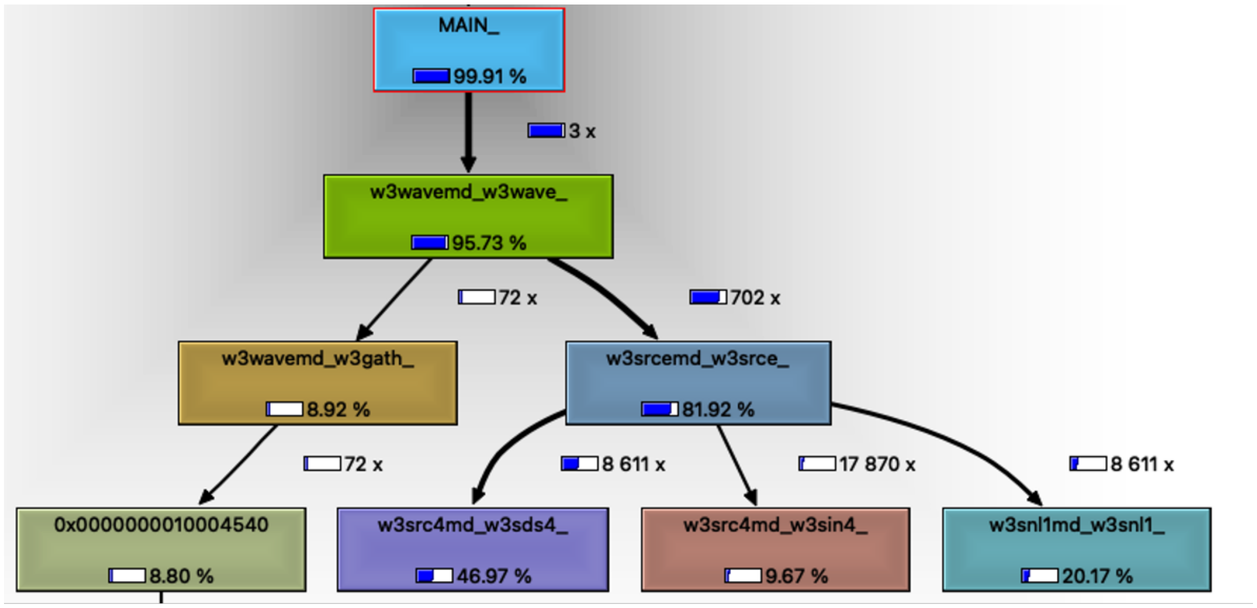

Figure 1: The Callgraph obtained by profiling WAVEWATCH III with 300 MPI ranks. Each box includes the name of the subroutine and its relative execution time as a percentage

line 188-189: I'm a bit surprised that Sin takes half of the source term computational, time, since I would have expected the non-linear interactions to be the heaviest. The cumulative breaking term in ST4 is supposed to be quite resource consuming, while possible not having a large effect on the end results. Is that turned on or off here? If it's turned on, then that would perhaps be an obvious candidate to try to speed up the model (not directly related to GPU porting).

Going back to ST3 would definitely speedup the code on both CPU and GPU. However, it is possible that the GPU gain some speedup over CPU due to the reduction of divergence among threads heavily present in ST4 code. We only focus on ST4 term since it is widely used for global wave modeling.

line 196-198 Is this a hard requirement, or does the lack of communication just mean that the source terms are trivially paralellisizable? The word "suitable" suggests that any communication here would make GPU porting a non-option, but I'm not sure that is the case (altough it probably becomes a lot more complex).

Yes, you are right. It would only make the porting more complex. We have changed the sentence to '**Fortunately, W3SRCEMD does not contain neighboring grid dependencies in both spatial**

and spectral i.e. no parallel data transfers, W3SRCEMD can therefore be ported to GPUs with less difficulty.' – Line 202 -205

line 241-266: very long paragraph. Even though the paper is generally very well written, this aspect could be checked.

We have modified it according to the suggestion. The paragraph has been split into two.

line 275: It seems like the order the figures are presented might be wrong, since Fig. 8 has already been referenced?

We have rearranged the figures.

line 295: Would it be possible to increase occupancy by reorganizing the loops? Now we loop over all grid points, and then loop over one spectrum, but perhaps it would be more efficient to define an array that has both spatial and spectral dimensions (and perhaps slice that up into some blocks, if needed)? Can you comment on this?

Although it is possible, it would require a significant amount of code refactoring, and that would defeat the whole purpose of using OpenACC in the first place, which is to avoid excessive refactoring. However, moving forward, this is necessary to achieve more speedups.

The paper is missing a discussion section. Although it might not strictly be needed in this kind of more technical paper, it would perhaps be interesting for the reader to know what kind of impact these speed-ups might have in practical terms. Several days of wall time was mentioned, but is this a "game changer" to allow for including wave models in ESMs, or do we still need to optimize? I'm also wondering how well the exact non-linear solution might scale (if the authors can comment), since this might have consequences to very basic reasearch into e.g. wave growth that might be affected by the crude approximations of DIA. Finally, would it every be viable to port any other parts of the wave model, such as the propagation, to GPUs, or is the communication needed beween the grid points a complete deal breaker?

Currently, on the CPU, WW3 in E3SM is computationally intensive. As E3SM is expanding other model components' capabilities to run on heterogeneous architecture, and depending on the speed-up achieved for other components, it might be necessary to further optimize WW3 code.
While there is still room for further GPU optimizations, such as pushing down the grid loop counter into the W3SRCEMD, this would require major code refactoring. There are other parts of WW3 code that can be ported to GPU, such as spatial propagation, and GPU communication is not a deal breaker, but it might require more bookkeeping. According to Fig. 1 (Fig. 4 in the manuscript), a significant speed-up should not be expected.

References

Abdolali, A., Roland, A., Van Der Westhuysen, A., Meixner, J., Chawla, A., Hesser, T. J., Smith, J. M., and Sikiric, M. D.: Large-scale hurricane modeling using domain decomposition parallelization and implicit scheme implemented in WAVEWATCH III wave model, Coast. Eng., 157, 103656, https://doi.org/10.1016/j.coastaleng.2020.103656, 2020

Roland, A. Development of WWM II: Spectral wave modelling on unstructured meshes. Diss. Ph. D. thesis, Technische Universität Darmstadt, Institute of Hydraulic and Water Resources Engineering, 2008.