Authors' response to referee #2: egusphere-2024-169: "Towards a real-time modeling of global ocean waves by the fully GPU-accelerated spectral wave model WAM6-GPU"

Hi, the anonymous referee,

Thank you for your constructive comments on our manuscript entitled 'Towards a real-time modeling of global ocean waves by the fully GPU-accelerated spectral wave model WAM6-GPU'. We feel indebted to you for your time on this manuscript. In the response below, all the comments and concerns are replied point by point, and the revised manuscript is attached as PDF supplements.

Kind Regards,
Ye Yuan, on behalf of the co-authors.

Summary
The authors have fully ported the spectral wave model (WAM6) to GPU using OpenACC with a substantial amount of code refactoring. On a GPU cluster with 32-core Intel Xeon6326 and 8 NVIDIA A100 GPUs, the WAM6-GPU code achieved a speed-up of 37x when utilizing all the resources on a node. As a result, they achieved around 90% reduction in power consumption.

This is an important study that would enable century-long global simulations with a stand-alone wave model and also facilitate the integration of wave models into Earth system models. However, before accepting this manuscript, the authors need to address the following issues thoroughly. Main:

1. In the abstract, the authors need to state the speed-up value based on a node comparison e.g., 32-core intel Xeon6326 and 8 NVIDIA A100.

Response: Adopted. Line 4-6 in the abstract has been rephrased as,
[Line6-9 in the revised manuscript] "*The power of GPU computing has been unleashed through substantial efforts of code refactoring, which reduces the computing time of a 7-day global 1/10 wave modeling to only 7.6 minutes in a single-node server installed with 8 Nvidia A100 GPUs. Speedup comparisons exhibit that running the WAM6 with 8 cards can achieve the maximum speedup ratio of 37 over the dual-socket CPU node with 2 Intel Xeon 6236 CPUs.*"

2. Looking into the code, I saw that most of the subroutines/modules were refactored. A rough estimate of how much the original CPU code has been refactored should be discussed within the manuscript.

Response: Adopted. As shown Figure 1 by shaded boxes, to eliminate expensive CPU-GPU data transfer, we have ported all computing modules to GPUs. During the study, we bear in mind that code refactoring can be a 'two-edges sword'. Too much code refactoring may be detrimental to code readability in terms of its physical concept. So based on the full absorption of model physics, we tried to achieve better speedup mainly through loop reordering and collapse, which haven't altered the code

structure in most cases. Substantial code refactoring mainly occurs in the source term of nonlinear interaction (SNONLIN), as well as MPI library (wam_mpi_comp_module.f90).

The above concern has been added to Line 151-154 in Section 3.2 of the revised manuscript.

"*It should be noted that too much code refactoring may be detrimental to code readability in terms of its physical concept. Based on the full absorption of model physics, loop reordering and collapse are the main strategies for better speedup, which haven't altered the code structure in most cases. Heavy code refactoring only occurs in computing source term of nonlinear wave interactions, as well as in some MPI interface functions.*"

3. One important thing missing from this paper is the structure of the WAM code. The authors should include a skeletal code structure of both CPU and GPU versions of some parts of the code. This would greatly improve the manuscript for readers, especially for understanding the SNL optimization explained in line 245-255.

Response: Figure 1 only gives a top-level flowchart of the WAM package. As stated in Line 71-72, the full description of the model structure and numerical schemes can be referred to ECMWF (2023), Gunther et al. (1992), and Behrens and Janssen (2013).

In Section 4.3, we have included the pseudocode for the SNL optimizing experiments, and more discussion has been made. I hope it will improve the readability of the manuscript in this section. Details please refer to Section 4.3 and Figure 6 in the revised manuscript, which is also shown in below.

- *Exp1: Placing the IJ loop to the innermost (Not adopted; **0.214 s**): It leads to too much overhead for serially launching thousands of kernels in a single time step.*

- *Exp2: Placing the IJ loop to the outermost and accessing global coefficient and index arrays directly (Not adopted; **0.252 s**): It leads to lower GPU occupancy and higher GPU latency due to spilling of local memory, and frequent access of global arrays within a kernel is detrimental to performance.*

- *Exp3: Placing the IJ loop between the M (frequency) and K (direction) loops (Adopted; 0.171 s for loop collapse on IJ and K, and 0.151 s for parallelism on IJ and sequential execution on K): It overcomes the shortcomings of the above experiments. Besides, actually two tests have been conducted in Exp3. By reorganizing the code substantially, we managed to collapse the IJ and K loops at first. As the second test, we did not do the loop collapse, and simply inserted <acc loop seq> before the nested K loop. Surprisingly, the second test took 0.02 s less time. Although loop collapse on IJ and K may increase code parallelism, it seems that the reorganized code leads to increased overhead.*

```
(a) $S_{nl}$ Exp1

 ! $S_{nl}$ : define the transfer of energy among wave components.
 ! index_array: 2D arrays defining indexes for interacting frequencies
 ! coef_array: 2D array used for computing wave interactions
 1  for each M;   ! Unable to parallalize
 2   index_1 = index_array(1, M) ... index_n = index_array(n, M)
 3   coef_1 = coef_array(1, M) ... coef_array(n, M)
 4  for each KH;  for each K;   ! Unable to parallalize
 5    ! defining indexes for interacting wave components
 6    MM, MP, MM1, K1, K2, K21 ...
 7   !$ACC KERNELS LOOP
 8   for each IJ;
 9     ! K' and M' defined by index_array
10    $S_{nl}(IJ, K, M) = f( coef\_n, S_{nl}(IJ, K', M'), ... )$
11    $S_{nl}(IJ, K1, M) = f( coef\_n, S_{nl}(IJ, K', M'), ... )$
12       ...
13    $S_{nl}(IJ, K21, MM) = f( coef\_n, S_{nl}(IJ, K', M'), ... )$
14       ...

(b) $S_{nl}$ Exp2

 1 !$ACC KERNELS LOOP
 2  for each IJ;
 3   for each M;   ! Unable to parallalize
 4    for each KH; for each K;  ! Unable to parallalize
 5     MM, MP, MM1, K1, K2, K21
 6     !Noting that accessing the global coef_array directly instead.
 7     $S_{nl}(IJ, K, M) = f( coef\_array, S_{nl}(IJ, K', M'), ... )$
 8     $S_{nl}(IJ, K1, M) = f( coef\_array, S_{nl}(IJ, K', M'), ... )$
 9        ... ...

(c) $S_{nl}$ Exp3

 1  for each M;
 2   index_1 = index_array(1, M) ... index_n = index_array(n, M)
 3   coef_1 = coef_array(1, M) ... coef_array(n, M)
 4   for each KH;
 5    !$ACC KERNELS LOOP COLLAPSE(2)
 6    for each IJ; for each K;
 7      K1 = ...   ! defining indexes for interacting wave components
 8      $S_{nl}(IJ, K1, M) = f( coef\_n, S_{nl}(IJ, K', M'), ... )$
 9    !$ACC LOOP COLLAPSE(2)
10    for each IJ; for each K;
11      K2 = ...;  $S_{nl}(IJ, K2, M) = f( coef\_n, S_{nl}(IJ, K', M'), ... )$
12         ... ...
```

Figure R1. Pseudocode of optimizing experiments on source term describing nonlinear wave interaction (S_{nl})

4. The use of two CPU-only HPC clusters is confusing. Given that the study focuses on GPU and not the optimization of the CPU code on the CPU, I think there is no need to run the CPU code on two CPU-only HPCs. Since the NMEFC's GPU server does not have more than one node needed for scalability of the GPU code, the authors should only keep the NMEFC's HPC cluster for comparing resource usage needed to achieve the GPU execution time.

Response: The Beijing Super Cloud Computing Center's (BJSC's) cluster is equipped with AMD EPYC 7452 CPUs with 64 cores, which was released in 2020 (Noting that Nvidia'a A100 was released in 2020), while the NMEFC's cluster uses older Intel Xeon 2680v4 CPUs with 28 cores (year of 2016-2017). As shown in Figure 5, to achieve a comparable performance with 8 A100 cards, the required processors for AMD and Intel are 2048 and 2688 cores, respectively. Sometimes there is no fair comparison between CPU and GPU performance. Besides, some readers may concern about the WAM's performance on different x86 CPU brands (Intel and AMD). So we decide to include the

metric for both CPU clusters in the manuscript. We can remove the content about the BJSC's cluster if the reviewer insist. This will not affect the completeness of the paper.

5. Fig. 7: The authors should show the spatial difference between the output parameters generated by the WAM6-GPU and the CPU version. Mean difference (Fig. 8) sometimes averages out the spatial difference between, if any.

Response: Below I have posted a log contour plot of spatial difference for significant wave height, wave mean period and wave directions between GPU and CPU versions. Generally, the absolute difference falls below 10^{-6} in most areas. This is normal since the WAM6 runs in single-precision. I didn't include the figure into the manuscript at this time, as the spatial distribution doesn't show any recognized patterns. In the reviewer suggest we do so, we will include it into the manuscript.

For other output parameters, the reader may want to run the GPU and CPU versions by themselves. Both CPU and GPU code packages are available at Zenodo. Besides, I have updated a new version WAM6-GPU v1.1 not long ago, which support grid nesting on GPU and fix some bugs. However, this version has not been checked thoroughly. Using this version should be cautious.
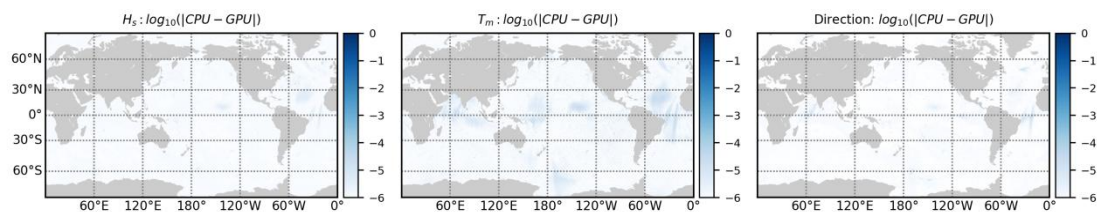

Figure R2. Logarithmic contour plot of the absolute difference between the modeling results of CPU and GPU version (a: Significant wave height; b: Mean wave period; c: Wave direction in rad).

6. Apart from running on the NVIDIA A100 GPU, are there any other further optimization strategies to improve the WAM6-GPU code on H100?

Response: Sorry, we have had no access to NVIDIA's H100. H100 is prohibited to sale in China Mainland at this time.

7. Just curious. Considering this study started in 2020, I wonder if the authors used P100 and V100. If so, what were the achieved speed ups?

Response: We indeed started to develop the WAM6-GPU with a server with 4 NVIDIA's early product V100. The theoretical computing power of the V100 is half of that for A100. The related metrics are shown in Figure R3.
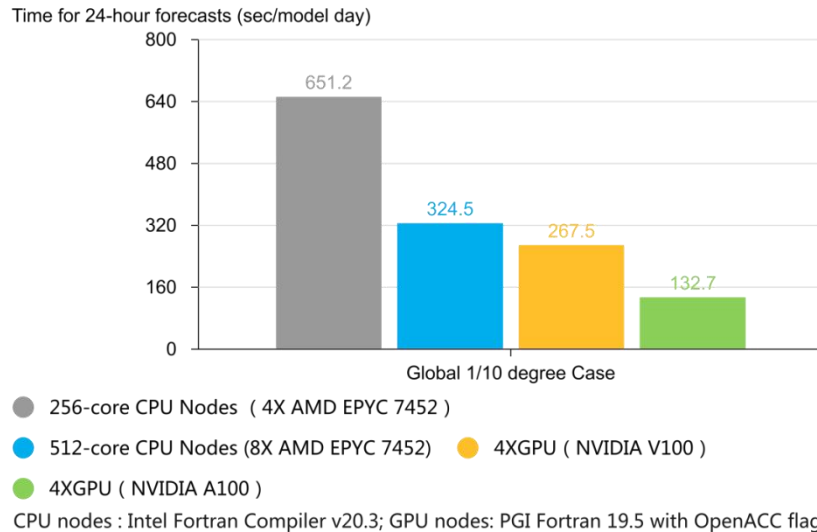
Figure R3. Computing time of the WAM6-GPU for 1-day global 0.1◦ wave hindcast on NVIDIA's V100 and A100 GPUs.

Minor:

Line 11: This is a scientific dataset. Cite Cavaleri et al., 2012 as in Line 20

Response: Adopted. Please refer to Line 15 in the revised manuscript.

Line 13: Check citation format.

Response: Corrected. Please refer to Line 17 in the revised manuscript.

Line 33: The new U.S. Department of Energy (DOE) Energy Exascale Earth System Model (E3SM) has also included WW3 as part of the default component. Cite Ikuyajolu et al., 2024 and Brus et al., 2021

Response: Adopted. However, Ikuyajolu et al. (2024) can not be added as Bibtex can not include ref that is in press. We will do it as the paper's information is available. Please refer to Line 35 in the revised manuscript.

Line 228: Define all terms in the equation

Response: All terms are defined now.Please refer to Line 240 in the revised manuscript.

Figure 6: Check caption for incorrect latex degree symbol

Response: Corrected. Please refer to Figure 7's captain in the revised manuscript.