Responses to reviews of "Actionable reporting of CPU-GPU performance comparisons: Insights from a CLUBB case study"

Gunther Huebler, Vincent E. Larson, John Dennis, and Sheri Voelz

We thank both reviewers for their careful reading of the manuscript and for their constructive comments.

Reviewer comments are repeated below in blue italics, with our responses interspersed in normal font. Representative changes to the manuscript are shown in orange.

1 Responses to Reviewer 1

The paper advances the science by introducing two new metrics, Peak Ratio Crossover (PRC) and Peak-to-Peak Ratio (PPR), which allow for a better comparison between CPU and GPU performance of a given application. The paper is well written and the claims are well supported by experiments and profiling data which naturally drive the conclusions. I would recommend the publication of the manuscript.

Thanks for the review and the positivity.

The GPU in the title is a broad topic while most of the findings are based on NVIDIA GPUs experiments. Would be interesting to see if all the conclusions still hold on MI250X (given different warp size and different generated assembly code). That is more for the future, not needed to be added for the current manuscript.

A broader cross-vendor GPU comparison would definitely be interesting, especially given the ever-evolving landscape of GPU architectures and compilation/optimization methods. This concern of presenting unfair/incomplete comparisons of available hardware was a large part of why this paper focuses on the performance metrics and reporting methods rather than on the specific performance comparisons between different hardware configurations.

No changes were made to the manuscript in response to this comment, but the OpenACC vs OpenMP section was modified slightly to make it clearer that the results presented there are exclusive to the

NVIDIA/nvhpc stack.

Line 107: "This setup .. closely mirrors the execution model.." - This is usually only partially true because a standalone has a smaller memory footprint in terms of the instruction cache.

Good call. This statement was meant to be specifically about the execution model, which does indeed mirror the execution model when CLUBB is embedded in a GCM, but could easily be interpreted as a more general statement about performance extrapolation. We kept the original phrasing about the execution model, but added a clarification immediately afterward to make clear that absolute timings may not extrapolate well to host-model runs.

"The overall batch size is therefore the number of MPI tasks multiplied by the number of columns per task, and the runtime we report corresponds to the wall-clock time of the full mpirun job — effectively the greatest runtime that any MPI task takes to complete a batched run. This setup both ensures effective hardware utilization and closely mirrors the execution model used when CLUBB is embedded in a GCM. However, absolute timing results may not extrapolate directly to host-model runs, because host models have a greater overall memory footprint and differ in the amount of non-CLUBB work performed by each MPI task."

Fig 3 / page 9 could be improved if the plots with the same number of levels would have the same colour, since the vertical levels are in focus here.

Excellent idea. This makes the plots much easier to interpret. We revised the figure to use the same color scheme for results which use the same number of vertical levels in both Fig. 3 and Fig. 4.

Section 5.4 OpenACC vs OpenMP - Firstly, the experiments were done with the NVIDIA compiler which is known to favour OpenACC. Secondly, Fortran + OpenMP is known to perform better than Fortran + OpenACC on AMD hardware, so without a similar experiment on AMD, I would rephrase the findings as limited to NVIDIA hardware using NVIDIA compiler family.

Agreed, the wording did seem too general. We revised this section so that the OpenACC-vs-OpenMP comparison is explicitly framed as applying to the NVIDIA GPUs and nvfortran compiler stack used in this study. We also clarified that the OpenMP directives were obtained through Intel's migration tool from the OpenACC source – which could also have an impact on the performance of the OpenMP version.

"Our metrics contextualize the seemingly "mixed" outcome at small and large batch sizes. Because our metrics focus the comparison on batch-size ranges where GPUs are beneficial, they show that OpenACC has an overall advantage over OpenMP for this configuration. In this study, the OpenMP

directives were generated from the OpenACC code using Intel's migration tool and compiled with the NVIDIA nvfortran compiler, so the comparison should be interpreted as specific to NVIDIA hardware and its compiler ecosystem."

Since the manuscript is not anonymised, maybe the authors can write their full names in lines 518-520.

Full names here do feel more appropriate. We replaced the author initials in the Author Contributions section with full names.

"The GPU port, timing measurements, performance profiles, and majority of the text for this work was completed by Gunther Huebler. The source code of CLUBB is maintained by a research group led by Vincent E. Larson at the University of Wisconsin Milwaukee. Vincent E. Larson, John Dennis, and Sheri Voelz provided invaluable guidance on how to approach and present the results of this work."

2 Responses to Reviewer 2

This paper gives a detailed performance analysis of the CLUBB cloud and turbulence parameterization, including a new GPU OpenACC based port. The performance work described in this paper is excellent. The paper is well written with clear arguments.

Thank you for the positive review.

Section 2: the wording of the sentence containing the Sun et al. (2023) reference could be clarified. At first reading, I assumed the authors were saying that the column loop changes in CLUBB were described in Sun et al. (2023), but I think the CLUBB changes are due to this work and Sun et al. (2023) was discussing similar work in a different parameterization (PUMAS).

Good call, the original wording was ambiguous. We revised the sentence to make clear that Sun et al. (2023) describes an analogous restructuring in the separate PUMAS codebase, not the CLUBB changes mentioned in this paper.

"A similar restructuring process, applied to the separate PUMAS codebase rather than CLUBB, is described in (Sun et al., 2023)."

Conclusions: most readers will be interested in how the CLUBB performance numbers will impact

global atmospheric models, in typical regimes that are either running at the limit of strong scaling to get the maximum possible throughput, or running on fewer nodes to maximize efficiency and maximize ensemble throughput. I think it would save the readers a little time if the authors added a short discussion explaining the relation between PPR and PRC and these strong scaling or maximum efficiency global model configurations.

This is an excellent point. The way we've defined and used the metrics does relate to the strong-scaling vs weak-scaling analyses that are common in the global model community, but we hadn't explicitly made this connection in the manuscript. The weak-scaling connection is pretty direct – the PPR compares each device at its own most favorable workload, and assumes we can keep the devices fully utilized, which is exactly what you would want to do in a weak-scaling analysis. The connection to the strong-scaling analysis is a little more subtle, because that usually involves subdividing a fixed problem size across different numbers of cores/devices, which is not exactly the same as subdividing a fixed problem size into different batches, but is still pretty close.

In the revised manuscript, we added this interpretation in two places: first in the introduction, where PPR and PRC are initially defined, and again in the summary sentence following Fig. 2, where the paper discusses which metric is most applicable in different situations.

"Because it compares each device at its own most favorable workload, PPR more fairly quantifies the speedup relevant to weak-scaling or ensemble use cases, where the problem size can grow to keep a device well filled with work and maximize sustained throughput. ... This makes PRC more useful for fixed-problem-size questions, as it more fairly determines which device is more performant, a common concern in strong-scaling scenarios where the goal is to minimize the runtime of a given problem by considering how best to subdivide it."

"Another useful way to think about metric choice is through a scaling lens. The PRC is more valuable to someone analyzing a fixed-problem-size scenario in a strong-scaling analysis, where the total problem size is fixed but the number of cores, devices, or batch sizes used to subdivide the work can vary. In contrast, the PPR is more useful in a weak-scaling analysis, where the problem size can grow by adding additional copies of the same underlying workload component to keep the device fully utilized."