Review of "Porting the Meso-NH atmospheric model on different GPU architectures for the next generation of supercomputers (version MESONH-v55-OpenACC)"

In this article, the author discuss their successful porting of the Meso-NH model, a mesoscale atmospheric model, from CPU to GPU architecture. The reason is to make use of the faster and more efficient running of certain types of calculations on GPU. The authors indeed report significant improvements in speed and energy efficiency using their new GPU code, and in their article discuss the technical details of the code porting, technical choices that need to be made, bit reproducibility efforts, and at the end a set of demonstrative simulations.

First of all, I would like to congratulate the authors with their new GPU-accelerated code, which no doubt was a big effort. I also like the bit-reproducibility work, which appears crucial in getting reliable results across different hardwares. In reading the article, I came across a few themes that I think need to be clarified, which relate mostly to validation and reproducibility, and a handful of small comments that I list at the end of this review. As my technical knowledge of OpenACC/MPI, compilers, and writing custom libraries is limited I will focus a bit more on the practical side, interpretation, and overal validation.

Major comments

Section 4 shortcomings
After demonstrating technical side of the porting process and the hardware / performance scaling of the model on various architectures using the author's standard validation case, section 4 is set to demonstrate the "physical realism" and even aims to "better understand the mechanisms involved in the formation of small-scale wind gusts" (lines 433-434). However, the discussion of the simulations is limited to qualitative descriptions which ultimately demonstrate little in terms of new understanding or physical realism. At the very least, I would expect comparison to observations here and quantitative measures of skill, for example compared to what is achievable using the same amount of computing power (in time or energy) on CPU-only simulations (which would demonstrate the benefit). Furthermore, to substantiate the claims of "successful cascade of scales" (e.g. line 10), I would expect a power density spectrum of wind and specific humidity at certain levels.

Reproducibility
In light of the previous comment, I thought I'd try and run the code myself on one of the NVIDIA GPU workstations we have available. I use these to run similar GPU-accelerated LES code. After close to an hour, I was not able to compile the library with the documentation supplied in https://zenodo.org/doi/10.5281/zenodo.13759713. Some error code was in French (when you

run `./configure` twice after changing a setting). The top-level README did not guide me through the installation process for a Linux PC, it seemed to be optimized for supercomputer, which is the main purpose, of course. I came across a url for instructions for Linux PCs in the README in the `MESONH-v55-OpenACC` folder, but that url does not work. The compilation seemed almost done, but there were no clear errors at the end - though I suspect an unlinked NetCDF library was the culprit. I don't doubt the compilation process will ultimately be straightforward, but for a user who has never worked with this model before, the instruction for a model as complex as this were too limited given the time I can spend on a review. I don't have access to the supercomputers used by the authors.

Scaling with radiation
Given that this model runs mesoscale domains at LES resolution, I would expected that details in physics parameterizations will start to matter. One example is radiative transfer calculations, see:

Maier, R., Jakub, F., Emde, C., Manev, M., Voigt, A., and Mayer, B.: A dynamic approach to three-dimensional radiative transfer in subkilometer-scale numerical weather prediction models: the dynamic TenStream solver v1.0, Geosci. Model Dev., 17, 3357–3383, https://doi.org/10.5194/gmd-17-3357-2024, 2024.

Veerman, M. A., van Stratum, B. J. H., & van Heerwaarden, C. C. (2022). A case study of cumulus convection over land in cloud-resolving simulations with a coupled ray tracer. *Geophysical Research Letters*, 49, e2022GL100808. https://doi.org/10.1029/2022GL100808

Ukkonen, P., & Hogan, R. J. (2024). Twelve times faster yet accurate: A new state-of-the-art in radiation schemes via performance and spectral optimization. *Journal of Advances in Modeling Earth Systems*, 16, e2023MS003932. https://doi.org/10.1029/2023MS003932

Please clarify:

- What scheme do you use and is it GPU-accelerated? Line 98 should be more specific here. I suspect ecRAD.
- How does the radiation scheme affect the scaling performance of CPU vs GPU code? Line 287 says you call it only every 900s.

Minor comments

Readability of the overal manuscript
As mentioned, I lack the technical know-how of the porting process, and so, feel free to not attribute too much value to this comment. However, if your goal is for the article to be readable to a broader audience, I would advice an approach where the logic and decision making of all steps is written in plain language, with the specific syntax/code not in-line but separate. I understand this may be unavoidable given the topic of the article.

For example, in much of the article, command line options, compiler flags, and run modes are included in parenthesis or in-line in such a way that, for me, the readability of the overal text is challenging and sometimes I lose track of what the purpose of a specific section or paragraph is. In section 4.2, meant to showcase the practical application of the model code, various new technical concepts and run flags are introduced in the first paragraph, and then again at the end of the second paragraph. Lines 106 to 114 may also better be placed in section 2.2? Also the concept of bit-reproducibility can, I think, be explained without the use of inline compiler flags.

Line 58: No code is bug-free, unfortunately. Do you mean that at least single to multi CPU vs GPU will give the same results, and so there are no bugs related to which architecture it runs on?
Line 427: "the use of **single** precision"