We thank the Referee for his/her time and his/her constructive comments. We have complied with most of the proposed changes. In the following, the comments made by the Referee appear in black, while our replies are in blue.

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.

We agree that Section 4 does not demonstrate the "physical realism" of the simulations or "the mechanisms involved in the formation of small-scale wind gusts". However, a quantitative comparison with observations and a detailed study of the physical processes are beyond the scope of the paper. As stated in the same section: "While the detailed processes involved are currently investigated in separate studies, the results illustrate that different types of storms are realistically simulated by Meso-NH and benefit from the combination of large domain and high resolution." Thus, we implemented two changes:

(1) We rephrased the first paragraph to make it clear that the purpose is to illustrate possible applications of high-resolution simulations on a large grid, without levering expectations about an assessment of their forecasting skill or a detailed examination of the involved physical processes, which are left for future studies.

(2) We added a new figure to show the benefits of high-resolution simulations (now Fig. 10 in the paper, and the left panel of Fig. 1 below): "Focusing on Atlantic storm Alex prior to landfall over Brittany (Fig. 9, top right panel), Fig. 10 shows the scale cascade as spectrum of kinetic energy in the middle of the boundary layer. The Meso-NH simulation (blue curve) exhibits three distinct ranges: the mesoscale for $\lambda >$10 km, the inertial subrange approaching the theoretical slope of the Kolmogorov spectrum (grey line) for $\lambda <$1 km, and an energy accumulation range in between for 10$>$ $\lambda >$1 km. Specifically, the energy accumulation range includes the fine-scale wind structures at the origin of gusts illustrated in Fig. 9. At smaller scales, the drop in energy for $\lambda <$400 m in Fig. 10 indicates that the effective resolution of the simulation reaches 4Δx." In addition, we compare the Meso-NH simulation with the highest-resolution operational model data available: "In contrast, the AROME operational analysis that provides the initial and lateral boundary conditions for the simulation (orange curve) diverges from Meso-NH for $\lambda <$10 km: it captures only the mesoscale and misses the energy accumulation and inertial subrange." We prefer this approach to using the same amount of computing power on CPU-only simulations, as suggested by the Reviewer, which would lead to a large energy consumption for limited added value. Finally, the right panel of Fig. 1 shows the energy spectrum at a lower level: while the

divergence with AROME for λ <10 km is similar, the Meso-NH does not clearly follow the slope of the inertial subrange for λ <1 km. This is due to the smaller scale of eddies closer to the surface, which would require even higher resolution to be represented explicitly. For the sake of brevity, only the left panel showing the spectra in the middle of the boundary layer is included in the paper.

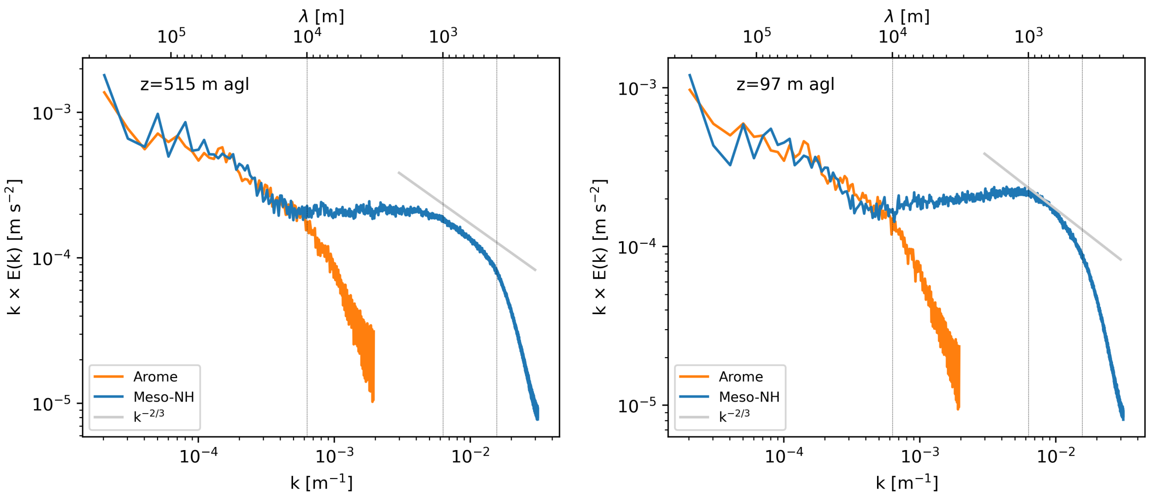

Figure 1: Energy spectrum for Atlantic storm Alex at 515 m agl (left panel) and 97 m agl (right panel) in the Meso-NH simulation (blue curve) and the AROME operational analysis (orange curve). The grey line shows the theoretical slope of the inertial range, while the vertical lines indicate the approximate wavelengths of the lower limit of the mesoscale ($\lambda \approx$10 km) and the upper limit of the inertial subrange ($\lambda \approx$1 km), as well as the effective resolution of the simulation ($\lambda \approx$400 m).

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.
We are sorry to hear that you are having trouble compiling Meso-NH. The problem with the NetCDF library sometimes arises because the compilation of the NetCDF library included in the Meso-NH package may conflict with other NetCDF libraries already compiled on the Linux PC, or because other utility libraries (compression, ...) are missing. It is not possible to take account of each user's specific system environment during the installation process. In the case of compilation issue, the user usually mails his/her problem to Meso-NH support, who will help solving it.

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.

We use the radiative code described by Gregory et al. (2000), Revision of convection, radiation and cloud schemes in the ECMWF model, Quart. J. Roy. Meteor. Soc., 126, 1685-1710, https://doi.org/10.1002/qj.49712656607. This reference is now added Line 98. It describes the radiation scheme used at ECMWF before the ecRad scheme was included in the IFS code. As noted in the text, no attempt has been made to port the radiation scheme to the GPU, so it is included in the performance scaling results, along with other components (Line 286, Figure 3 and "Others" in Figure 4).

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. As you write, we added run flags and other technical elements to show the practical application of porting the Meso-NH code to GPUs. In section 4.2, we have refrained from using these details in the revised version. We prefer to keep lines 106 to 114 in sub-section 2.1, as they are not limited to sub-section 2.2, but include the modifications described in all the following sub-sections. When describing the verification of bit-reproducibility, we prefer to keep the indication of the inline compiler flags, as they are essential to the execution of the MPPDB_CHECK library.

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? We agree that no code is bug-free. Here, we state that the implementation for massively parallel executions on CPU or GPU supercomputers is bug free. To clarify, the sentence is now "To our knowledge, Meso-NH is the only atmospheric (or oceanic) model offering bit-level reproducibility. This outstanding capability guarantees that Meso-NH is parallelization bug-free, i.e. there are no bugs in its implementation for parallel executions on CPUs and GPUs."
Line 427: "the use of **single** precision" Changed