The authors present a heterogeneous CPU-GPU implementation of a horizontal advection module in the EPICC air quality model. I commend the authors' efforts to increase air quality modeling efficiency towards timely high-resolution forecasting to protect human health and wellbeing. While generally comprehensive, the presentation at times drifts away from the work's novelty and omits evaluating the horizontal advection module when it is coupled back to the full model.

Response: We appreciate the editor for reviewing our manuscript and for the valuable suggestions, which we will address point by point in the following.

General comments

- **A 1:1 CPU:GPU ratio with only 10 CPU cores does not necessarily reflect operational runtime configurations. Considering heterogeneous compute environments and hardware availability, would CPU:GPU ratios greater than 1 lead to resource competition and/or serialize kernel calls, compromising efficiency gains?**

Response: Your question is both accurate and pertinent. Allocating a single GPU to multiple CPU processes may lead to competition in data transfer and increase the risk of GPU memory overflow in high-resolution applications with large data scales. Therefore, designing a more complex CPU-GPU matching mechanism is required to achieve pairing between multiple CPU processes and a single GPU. Currently, we opt for the relatively easier-to-implement scheme of matching a single CPU process with a single GPU. In the future, on one hand, while avoiding data transfer competition between CPU and GPU, we will consider designing a more sophisticated mechanism for matching multiple CPU processes with a single GPU. On the other hand, drawing on Cao et al. (2024), we plan to introduce an OpenMP shared-memory parallel scheme into the EPICC-Model. Through multi-level hybrid parallelism, while porting computationally intensive modules to the GPU for parallel computing, other modules running on the CPU will utilize OpenMP multithreading parallelism, thereby fully leveraging CPU computing resources. We have modified this part in **lines 771-781**, which are as follows:

Finally, in current heterogeneous architecture supercomputing systems, the number of CPU processes within a computing node typically exceeds the number of GPUs. Employing the current matching scheme of one CPU process to one GPU acceleration card results in the waste of

remaining CPU computing resources. In the future, on one hand, while avoiding data transmission competition between the CPU and GPU, we will consider designing a more sophisticated mechanism for matching multiple CPU processes with a single GPU card. On the other hand, drawing on Cao et al. (2024), we plan to introduce an OpenMP shared-memory parallel scheme into the EPICC-Model. Through multi-level hybrid parallelism, while porting computationally intensive modules to the GPU for parallel computing, other modules running on the CPU will utilize OpenMP multithreading parallelism, thereby fully leveraging CPU computing resources.

- **Considering the comparison between implementations summarized in Table 4, Section 4.2.2 seems somewhat out-of-scope, more in line with evaluation of the EPICC model itself than the performance of the accelerated horizontal advection implementation, which is this work's novelty.**

Response: Thanks for your constructive suggestions. Following successful offline validation of GPU-HADVPPM4HIP V1.0, we have supplemented this revised manuscript with an additional set of comparative experiments. Using identical input data and model configurations, we compared the differences in computational results between the HIP-Opt2. version of EPICC-Model and the original Fortran version after a 24-hour integration. The HIP-Opt2. version is the heterogeneous version formed by coupling GPU-HADVPPM4HIP V1.0 into EPICC-Model, with optimizations in communication, thread and cooperative indexing, and hybrid parallelization. Although the differences in computational results of the HIP-Opt2. version remain within acceptable limits, its computational stability on heterogeneous clusters and its simulation performance in real-world scenarios have not yet been evaluated. To address this, we conducted a one-month real-world case simulation using the HIP-Opt2. version. This serves both to test the integration stability of the model on the China's domestic heterogeneous clusters and to evaluate the model's simulation performance in comparison with observational data. The reasons for this are explained in **lines 492–503** of the manuscript., which are as follows:

After parallelizing the HADVPPM program in the air quality model CAMx on China's domestic GPU-like accelerators, Cao et al. (2024) conducted comparative analyses of simulation results between NVIDIA GPUs and domestic GPUs through offline and coupled testing

approaches. Although both experimental results indicated smaller computational errors introduced by China's domestic GPUs, the study did not validate the discrepancies between CAMx simulation results and actual observational data, particularly regarding model performance in real-case scenarios. To address this gap, the current study integrates GPU-HADVPPM4HIP V1.0 into the EPICC-Model V1.0 and performs one-month real-case simulations following the experimental configuration described in Sect. 4.1. This serves dual purposes: firstly, to verify the computational stability of EPICC-Model V1.0 in cross-architecture heterogeneous cluster environments using China's domestic hardware, and secondly, to evaluate the model's pollutant simulation performance through observational validation.

- **The statements claiming a 25.0x efficiency gain seem to refer to a baseline early heterogeneous implementation that was less efficient than the standard CPU implementation. This seems to inflate the actual efficiency gain, which is confusing or potentially misleading.**

Response: We apologize for any confusion caused by our previous description. In fact, we restructured the Fortran source code of the advection module using the standard C programming language and adapted it for domestic GPUs via the HIP API, resulting in GPU-HADVPPM4HIP V1.0. However, simply coupling GPU-HADVPPM4HIP V1.0 into EPICC-Model without any further optimization led to extremely low computational efficiency on the domestic heterogeneous cluster, where simulating one hour took approximately 5.4 hours. Performance analysis revealed that the efficiency of data transfer between the CPU and GPU was one of the primary factors contributing to this poor performance. Therefore, we improved data transfer efficiency by reducing the communication frequency between the CPU and GPU. This optimization resulted in a 25.0x increase in the model's computational efficiency. By significantly enhancing computational efficiency on the heterogeneous cluster through reduced CPU-GPU communication frequency, we aim to emphasize the fundamental architectural differences between CPUs and GPUs. In recent years, the substantial improvement in GPU computational performance has opened new directions for accelerating numerical models. However, influenced by the historical development of modeling frameworks, early numerical models were primarily designed for CPU

architectures. Achieving efficient parallel computation of numerical models on GPUs cannot be accomplished simply through direct code translation or the use of heterogeneous programming interfaces alone. Rather, it requires not only adapting the model to GPUs but also redesigning computational workflows, loop structures, and logic in accordance with GPU architectural characteristics, so as to fully leverage the powerful parallel computing capabilities of GPUs. We have modified this part in **lines 642-653**, which are as follows:

By significantly improving the computational efficiency of the model on heterogeneous clusters through reducing the communication frequency between the CPU and GPU, we aim to emphasize the fundamental architectural differences between CPUs and GPUs. In recent years, although the substantial increase in GPU computational performance has opened up new directions for enhancing the efficiency of numerical models, the historical development of modeling frameworks means that early numerical models were predominantly designed for CPU architectures. Achieving parallel computation of numerical models on GPUs cannot be accomplished merely through straightforward code translation or by relying solely on heterogeneous programming interfaces. Instead, beyond adapting the model to GPUs, it is necessary to redesign the computational workflow, loop structures, and logical organization in accordance with GPU architectural characteristics, so as to fully leverage the powerful parallel computing capabilities of GPUs.

- **I would be interested in an accuracy evaluation of the fully coupled implementation. I wonder if the different hardware's handling of floating precision operations could lead to greater discrepancies than presented in the offline evaluation.**

Response: Thanks for the constructive comment. Accordingly, we have supplemented a set of comparative experiments. Using identical input data and model configurations, the differences in simulation results between the HIP-Opt2 version and the original Fortran version were evaluated in a real-world case study. The HIP-Opt2 version refers to the heterogeneous parallel version formed by coupling GPU-HADVPPM4HIP V1.0 into the EPICC-Model, along with optimizations in communication, thread and block coordinated indexing, and hybrid parallelization. In these comparative experiments, the model's input data and configurations were set as described in

Section 4.1, with the simulation period covering 24 hours from 00:00 to 23:00 UTC on July 1, 2021.

Figures 1 and 2 present the simulated results of gases (e.g., HONO, SO_2, NO_2, NH_3) and aerosols (e.g., BC, $PM_{2.5}$, ASO_4, ANH_4) after a 24-hour integration by both the HIP-Opt2. and the original Fortran versions, along with the absolute errors (AEs) between the two model versions. As visually evident from the figures, the results from HIP-Opt2. after the 24-hour integration closely match those from the original Fortran version. For the vast majority of grid points, the AEs for gases and aerosols are within ±0.1 ppbV or ±0.1 $\mu g \cdot m^{-3}$.

To assess the scientific applicability of HIP-Opt2., we followed the methodologies of Wang et al. (2021) and Cao et al. (2024) by introducing two metrics: the root mean square error (RMSE) and the standard deviation (std). The ratio of RMSE to std was calculated to evaluate the scientific usability of HIP-Opt2. Here, RMSE represents the error between HIP-Opt2 and the original Fortran version for different species, while std denotes the standard deviation of the respective species in the Fortran version. Taking NO_2 as an example, if the RMSE/std ratio is very small, it indicates that the computational deviation introduced by heterogeneous parallel acceleration is negligible compared to the intrinsic spatial variability of NO_2 itself. This suggests that such minor computational errors do not affect the model's utility in scientific research. Table 1 lists the RMSE, std, and their ratios for gases (e.g., HONO, SO_2, NO_2, NH_3) and aerosols (e.g., BC, $PM_{2.5}$, ASO_4, ANH_4). The RMSE/std ratios for these species range from 10^{-5}% to 10^{-2}%. Specifically, BC exhibits the smallest ratio at 4.8×10^{-5}%, while ASO_4 shows the largest ratio, yet only at 7.0×10^{-2}%. The RMSE/std ratios for all species in HIP-Opt2. are comparable to those reported by Cao et al. (2024), demonstrating that the simulation results from the HIP-Opt2 version are fully suitable for scientific research. We have modified this part in **lines 449-489**, which are as follows:

By coupling GPU-HADVPPM4HIP V1.0 into the EPICC-Model and implementing optimizations—such as reducing the communication frequency between CPUs and domestic GPUs, adopting thread and block coordinated indexing, and employing "MPI+HIP" hybrid parallelization—we developed the HIP-Opt2. version. Using identical input data and model configurations, the differences in simulation results between the HIP-Opt2. version and the original Fortran version were compared in a real-world case study. The model input data and configurations were set as described in Section 4.1, with the simulation period covering 24 hours

from 00:00 to 23:00 UTC on July 1, 2021.

Figures 1 and 2 present the simulated concentrations of gases (e.g., HONO, SO_2, NO_2, NH_3) and aerosols (e.g., BC, $PM_{2.5}$, ASO_4, ANH_4) after a 24-hour integration by both the HIP-Opt2. and the original Fortran versions, along with the Absolute Errors (AEs) between the two model versions. As visually evident from the figures, the results from HIP-Opt2. after the 24-hour integration are in close agreement with those from the original Fortran version. For the vast majority of grid points, the AEs for gases and aerosols remain within ±0.1 ppbV or ±0.1 $\mu g \cdot m^{-3}$.

To further evaluate the suitability of HIP-Opt2. for scientific research, we followed the methodologies of Wang et al. (2021) and Cao et al. (2024) by introducing two metrics: the root mean square error (RMSE) and the standard deviation (std). The ratio of RMSE to std was calculated to quantify the scientific applicability of HIP-Opt2. Here, RMSE represents the error between HIP-Opt2. and the original Fortran version for different species, while std denotes the standard deviation of each species in the Fortran version. Taking NO_2 as an example, a very small RMSE/std ratio indicates that the computational deviation introduced by heterogeneous parallel acceleration is negligible compared to the inherent spatial variability of NO_2. This implies that such minor computational errors do not compromise the model's utility in scientific research. Table 1 lists the RMSE, std, and their ratios for the aforementioned gases and aerosols. The RMSE/std ratios for these species range from 10^{-5}% to 10^{-2}%. Specifically, BC exhibits the smallest ratio at 4.8×10^{-5}%, while ASO_4 shows the largest ratio, yet only at 7.0×10^{-2}%. The RMSE/std ratios for all species in HIP-Opt2 are comparable to those reported by Cao et al. (2024), demonstrating that the simulation results from the HIP-Opt2 version are fully suitable for scientific applications.

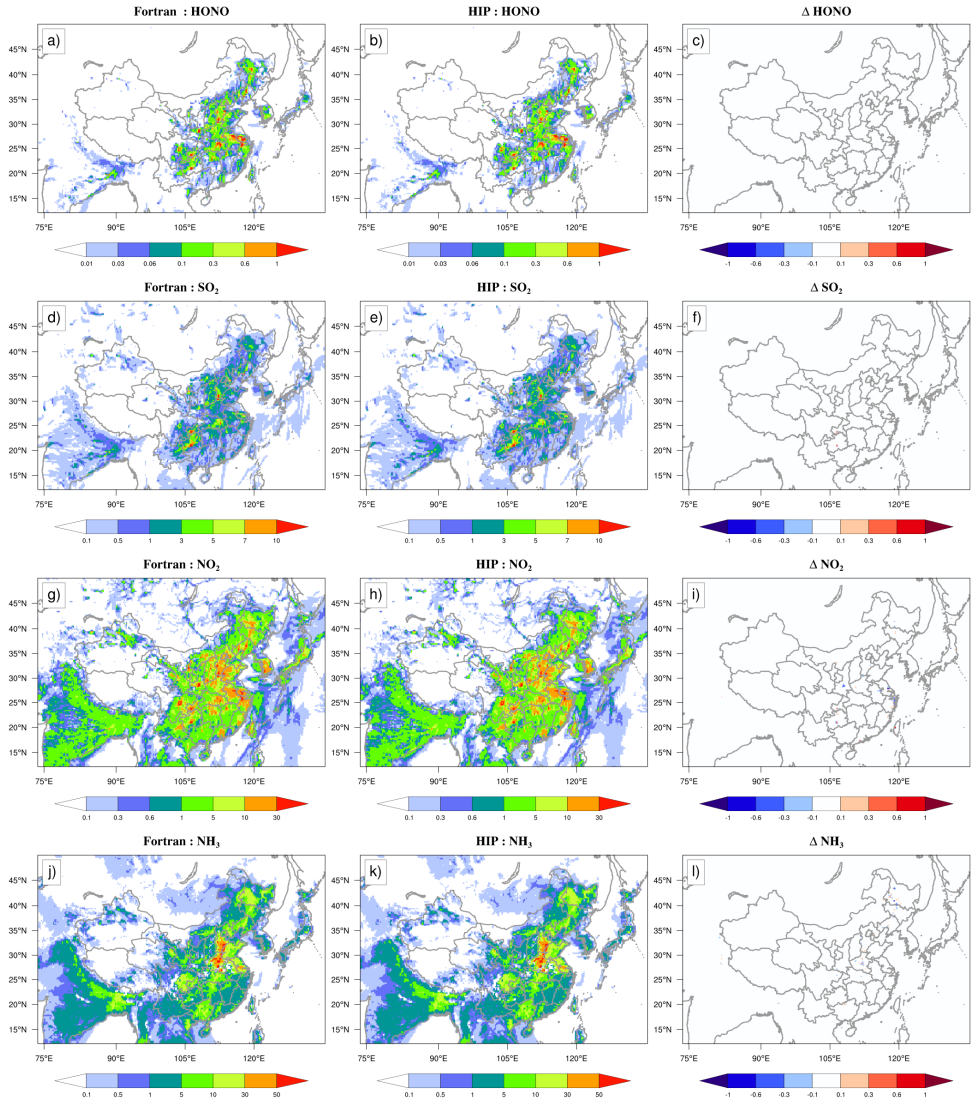

Figure 1. *HONO, SO_2, NO_2, and NH_3 concentrations outputted by the EPICC-Model for the Fortran and HIP-Opt2. versions. Panels (a), (d), (g), and (j) are from the Fortran version. Panels(b), (e), (h), and (k) are from the HIP-Opt2. version. Panels (c), (f), (i), and (l) are the absolute errors (AEs) between the Fortran and HIP-Opt2. Versions.*

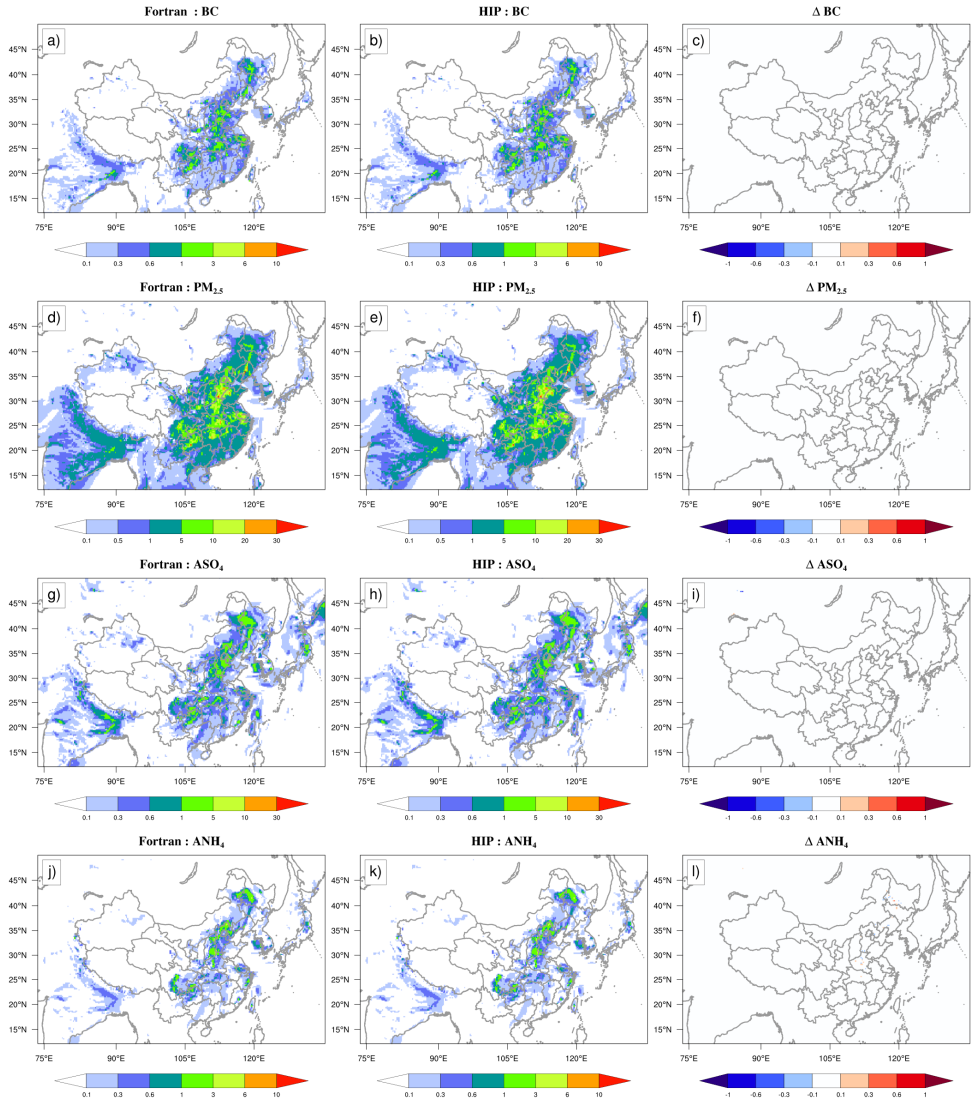

Figure 2. *BC, $PM_{2.5}$, ASO_4, and ANH_4 concentrations outputted by the EPICC-Model for the Fortran and HIP-Opt2. versions. Panels (a), (d), (g), and (j) are from the Fortran version. Panels(b), (e), (h), and (k) are from the HIP-Opt2. version. Panels (c), (f), (i), and (l) are the absolute errors (AEs) between the Fortran and HIP-Opt2. versions*

Table 1. *The root mean square error (RMSE) between the Fortran and HIP-Opt2. versions, standard deviation (std) of the Fortran version, and the RMSE and std ratio.*

	RMSE	**std**	**RMSE/std (%)**
HONO (ppbV)	2.1×10^{-5}	0.1	2.1×10^{-2}
SO_2 (ppbV)	3.3×10^{-5}	0.7	4.5×10^{-3}
NO_2 (ppbV)	2.8×10^{-4}	3.4	8.3×10^{-3}
NH_3 (ppbV)	9.6×10^{-5}	4.1	2.4×10^{-3}

BC ($\mu g \cdot m^{-3}$)	9.7×10^{-8}	0.2	4.8×10^{-5}
$PM_{2.5}$ ($\mu g \cdot m^{-3}$)	4.4×10^{-6}	1.9	2.3×10^{-4}
ASO_4 ($\mu g \cdot m^{-3}$)	1.7×10^{-4}	0.2	7.0×10^{-2}
ANH_4 ($\mu g \cdot m^{-3}$)	1.1×10^{-4}	0.2	5.8×10^{-2}

Specific comments

- **Line 390: Mean over what dimensions?**

Response: We sincerely apologize for any lack of clarity in our previous statements. In this study, we compared the offline computational result differences between the Fortran and standard C programming language versions of the HADVPPM program (F-to-C), between the standard C language version of HADVPPM and the GPU-HADVPPM4HIP program (C-to-HIP), and between the Fortran version of HADVPPM and the GPU-HADVPPM4HIP program (F-to-HIP). For the offline result comparison, a dedicated Fortran program was developed to generate the same input dataset, which consists of 100 double-precision floating-point numbers. After the advection computation, the output from this dataset also comprises 100 double-precision floating-point numbers. Taking the comparison between the Fortran and C language versions of HADVPPM as an example, the mean absolute error (AE) and mean relative error (RE) refer to the average AE and RE calculated from the 100 double-precision floating-point results produced by the Fortran version and the C version after the advection computation. The meanings of the mean AE and mean RE for other version pairs are the same, i.e., they represent the average AE and RE between the 100 double-precision floating-point results from the two corresponding versions. We have revised this part in **lines 402-409**, which are as follows:

To ensure input consistency, a dedicated Fortran program was developed to generate identical input datasets, including 100 double-precision floating-point numbers, for all three implementations. Each implementation executed a complete advection integration computation, with subsequent output recording and analysis. Therefore, the absolute errors (AE) and relative errors (RE) presented in Table 4 represent the average values computed from the 100 double-precision floating-point results produced by the original Fortran code, restructured standard C code, and HIP-accelerated code of the HADVPPM program after performing the advection solution computation under the given input conditions.

■ **Line 481: Units on 10^7?**

Response: We sincerely apologize for the lack of clarity in our previous statement. What we intended to express is that the offline test results indicate that when the HADVPPM program was ported from the CPU to the domestic GPU, the computational discrepancies introduced by the heterogeneous porting process are extremely minor, on the order of 10^{-7}. That is, differences in the computational results only begin to appear at the seventh decimal place. We have revised this part in **lines 549-551**, which are as follows:

Specifically, the HADVPPM program exhibits small discrepancies on the order of 10^{-7} when ported from CPU to domestic GPU-like accelerator architectures. That is, differences in the computational results only begin to appear at the seventh decimal place.

Technical corrections

■ **Lines 145-146: "Initial conditions (BC)" should be "initial conditions (IC)"**

Response: We apologize for the error. The abbreviation for "initial conditions" has been corrected to "IC" throughout the manuscript.

■ **Line 447: Correlation coefficient should be lower-case "r".**

Response: We apologize for the error. The abbreviation for "correlation coefficient" has been updated to a lower-case "r," and all corresponding instances have been corrected throughout the manuscript.

■ **For flow, "China's domestic GPU-like accelerators" could be simplified to just "accelerator" or similar after the first occurrence.**

Response: Thanks for your valuable suggestion. Upon its first mention in the manuscript, the full term "China' s domestic GPU-like accelerators" is introduced, and thereafter it is standardized to

"GPU-like" for consistency. This adjustment has been applied throughout the manuscript.

Reference

Cao, K., Wu, Q., Wang, L., Guo, H., Wang, N., Cheng, H., Tang, X., Li, D., Liu, L., Li, D., Wu, H., and Wang, L.: GPU-HADVPPM4HIP V1.0: using the heterogeneous-compute interface for portability (HIP) to speed up the piecewise parabolic method in the CAMx (v6.10) air quality model on China's domestic GPU-like accelerator, Geosci. Model Dev., 17, 6887-6901, 10.5194/gmd-17-6887-2024, 2024.

Wang, P., Jiang, J., Lin, P., Ding, M., Wei, J., Zhang, F., Zhao, L., Li, Y., Yu, Z., Zheng, W., Yu, Y., Chi, X., and Liu, H.: The GPU version of LASG/IAP Climate System Ocean Model version 3 (LICOM3) under the heterogeneous-compute interface for portability (HIP) framework and its large-scale application, Geosci.Model Dev., 14, 2781–2799, https://doi.org/10.5194/gmd-14-2781-2021, 2021.