Responses to Reviewer #1

We thank the reviewer for taking the time to review our paper and for the constructive comments. The page and line numbers that we quote for indicating where we changed the manuscript refer to the revised marked-up version.

(1.1) The code can readily support sectional and per-particle models. In principle, TChem- atm can be adapted to support more mechanisms than the two curently supported (UCI, CB05). However, it should be made clear (also in the abstract) that modal host models need to be adapted to support TChem-atm, and that architecute-specific performance tuning is required (for example a factor of 2 improvement is reported for the NVIDIA H100) to achieve higher efficiency.

We followed the reviewer's suggestion by revising the abstract and the introduction.

- In the abstract (P1L11): "TChem-atm enables performance-portable execution across CPUs and GPUs, though optimal efficiency may require modest architecture-specific tuning (e.g., team and vector sizes), with up to a twofold improvement on the NVIDIA H100. It directly supports sectional and particle-resolved host models, while modal aerosol schemes require minor adaptation to provide particle-scale quantities such as representative diameters."
- In the introduction (P3L74): "TChem-atm is compatible with a wide range of aerosol representations, but the degree of direct interoperability differs across host models. Sectional and particle-resolved frameworks map naturally onto TChem-atm's computational-particle abstraction. Modal aerosol schemes, however, typically do not expose particle-scale quantities such as representative diameters or per-species mass vectors, and therefore require modest adaptation to supply these inputs. Once provided, TChem-atm can evaluate multiphase chemistry within modal frameworks in the same manner as for other aerosol representations."
- In the introduction (P4L92): "Although TChem-atm achieves performance portability across CPUs and GPUs without architecture-specific code, performance is further improved when tunable parameters (e.g., Kokkos team and vector sizes) are optimized for the target hardware. As demonstrated in Section 3.2, architecture-aware tuning yields up to a factor-of-two speed improvement on the NVIDIA H100 relative to the Kokkos default configuration."

(1.2) The current validation focuses on a box-model scenario using the CB05 mechanism and SIMPOL partitioning. While this is adequate as a proof-of-concept, it falls short of a scientifically realistic demonstration (e.g., regional-scale or global model simulation) that would better showcase TChem-atm's capabilities and its potential to speed-up atmospheric chemistry-aerosol production simulations. Again, in the abstract (and elsewhere) it would be more fitting to describe the results presented here as "proof-of-concept".

We agree with the reviewer that our tests represent a proof-of-concept demonstration rather than a full scientific application. The goal of this initial paper is to validate correctness, establish interoperability with an existing host model (PartMC), and quantify performance portability across architectures. We have revised the abstract, the introduction and the beginning of Section 3 to emphasize this.

- In the abstract (P1L7): "In a proof-of-concept integration with the particle-resolved model PartMC, TChem-atm reproduces the existing PartMC–CAMP implementation within solver tolerances and delivers substantial GPU speedups, especially for large particle populations."
- In the introduction (P3L83): "The present study is intentionally scoped as a proof-of-concept implementation that validates correctness and establishes performance characteristics of the new multiphase chemistry framework. Because the integration of CAMP chemistry with TChem's performance-portable backend is novel, our immediate priority is demonstrating

functional equivalence to existing implementations and quantifying solver and hardware scaling. Application within regional- or global-scale models is a natural next step, but lies beyond the scope of this initial paper."

- In section 3 (P16L374): "Because the goal of this work is to establish and validate the TChem-atm framework, the tests presented here focus on controlled box-model scenarios that isolate chemical behavior and computational scaling."

(1.3) If possible, the manuscript could benefit from a brief profiling summary (e.g., fraction of time spent in kernel execution vs. memory transfer) to strengthen the discussion.

We appreciate this suggestion. We performed lightweight profiling on the timing runs reported in Section 3. For these idealized box-model experiments, the Kokkos kernels accounted for $> 99.9\%$ of the wall-clock time, with $< 0.1\%$ spent in host–device memory transfers. We have added a short discussion to the manuscript clarifying this point.

In Section 3.2 (P22L486): "We also conducted lightweight profiling during the timing experiments to separate kernel execution from host–device memory transfer. For the idealized box-model runs presented here, more than 99.9% of the wall-clock time occurs inside the Kokkos kernels, with memory transfers accounting for only the remaining fraction. This behavior is expected in the present setup, where particles share identical initial conditions, the RHS evaluation is uniform across particles, and memory movement is minimized. In more realistic host-model applications, such as full PartMC–TChem-atm simulations, additional data exchange between the host model and TChem-atm, more heterogeneous particle states, and increased stiffness may increase the relative cost of memory transfers and solver iterations. Thus, while our profiling confirms that kernel execution dominates in these proof-of-concept tests, the performance breakdown will necessarily depend on the host-model coupling pattern and the chemical and microphysical variability of the simulated system."

(1.4) Finally, it is not clear why the authors evaluated the performance of direct linear solvers for dense (full) Jacobians for atmospheric chemistrykinetics (where Jacobians are generally sparse, as is the case for the mechanism tested here). Thus, the reported results in terms of memory limitation, and longer time-to-solution are to be expected and almost trivial. And of course, preconditioned iterative methods are preferrable for the large, stiff ODE systems of atmospheric chemical kinetics because they require significantly less memory than direct solvers. It is suggeted to refactor the disucssion around this.

We thank the reviewer for this important clarification. We agree that dense direct solvers are not appropriate for large atmospheric chemistry systems, where Jacobians are sparse and memory requirements scale poorly. Our intention in including dense solvers was not to suggest that they are viable for large, stiff mechanisms, but rather to provide a performance baseline and to illustrate the regime in which they are competitive—namely, very small systems (e.g., few computational particles, reduced mechanisms, or modal/sectional models with low dimensionality).

To address this concern, we have revised the discussion in Section 3.2 to state explicitly as follows (P22L474): "To interpret these performance trends, it is helpful to clarify the role of direct versus iterative solvers. We include dense-direct linear solvers in our performance evaluation not because they are expected to be optimal for large atmospheric chemistry systems—indeed, the Jacobians arising from CB05 and similar mechanisms are sparse—but because they provide a well-defined performance baseline for small to moderate system sizes. Dense solvers incur higher memory costs and scale poorly as the number of particles grows, so their performance degradation is expected. However, these solvers remain competitive in regimes with few computational particles or simplified aerosol representations (e.g., small sectional or modal systems), where the Jacobian dimension remains modest and overhead is low. Including them therefore helps delineate the transition point at which preconditioned iterative methods such as GMRES become the

clearly superior choice. Our intention is not to advocate dense solvers for large, stiff atmospheric chemistry problems, but rather to provide a complete performance map across solver categories and hardware backends.

For realistically sized atmospheric mechanisms, preconditioned iterative Krylov methods are preferable because they exploit sparsity and dramatically reduce memory demands. The results in Section 3.2 are therefore consistent with expectations and are included to contextualize solver performance across operational regimes."

(1.5) The reaction mechanisms in Fig.3 do not appear correctly in the preprint. Please check.

We fixed this, thank you.

(1.6) Line 136 and elsewhere: Please be consistent with capitalisation and code names e.g. "Sundials CVODE" or "SUNDIALS", etc.

We fixed this.

(1.7) Sec. 2.5 only discusses the partitioning of semi-volatile species. It's not clear how other cases of heterogeneous reactions are handled.

At present, TChem-atm implements only one class of heterogeneous processes: semi-volatile gas–aerosol partitioning following the SIMPOL formulation. Other heterogeneous processes—such as reactive uptake, surface-mediated chemistry, or condensed-phase reactions—are not yet included in the current release.

We have revised Section 2.5 (now Section 2.6) to state this explicitly and to clarify that the TChem-atm framework is designed to accommodate additional heterogeneous process types: new processes can be incorporated by extending the mechanism specification and adding the corresponding source-term kernels, following the CAMP-style modular process abstraction. These extensions constitute planned future work.

We added the following paragraph in Section 2.6 (P11L244): "This section focuses on semi-volatile partitioning because this is the only heterogeneous process type currently implemented in TChem-atm. The framework, however, is extensible: heterogeneous reactions, reactive uptake parameterizations, or condensed-phase chemical processes can be incorporated by defining the corresponding rate expressions in the mechanism file and adding the associated source-term kernels. These capabilities follow the CAMP abstraction and will be implemented in future extensions of TChem-atm. At present, semi-volatile gas–aerosol partitioning represents the primary heterogeneous pathway supported."

(1.8) Fig. 5: It's clear that there's agreement as also reported in the text, so this figure offers little for the reader. Please consider plotting the differences if possible between runs instead, to reveal more architecture-related information.

We added the following figure, right after Figure 5:

We updated the text accordingly (P17L409): "To more clearly quantify the agreement between the CPU, GPU, and legacy PartMC–CAMP implementations, Figure 6 shows the relative differences between the CPU, GPU, and PartMC–CAMP results. The discrepancies remain small throughout the simulation, with a modest temporal drift for some species that reflects normal numerical variability rather than any architecture-specific effect. All deviations remain well within solver tolerances."

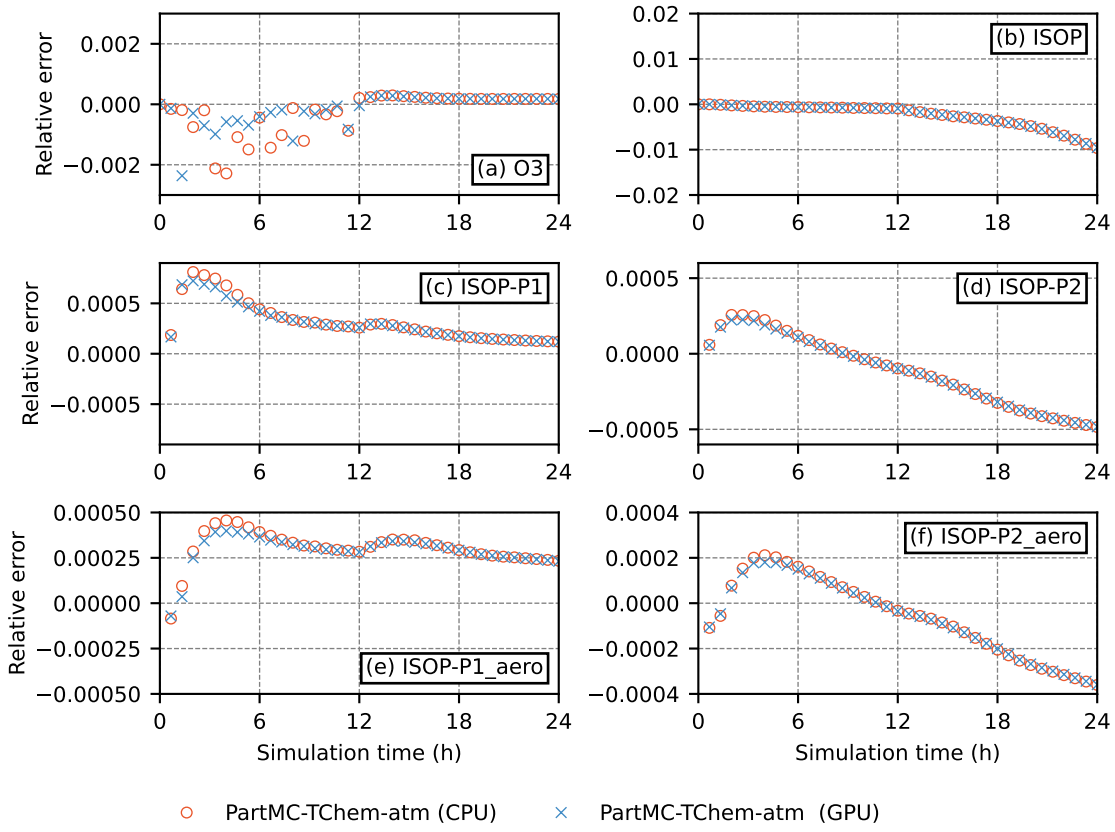

Figure 1: Relative differences between CPU, GPU, and PartMC–CAMP mixing ratios for the species shown in Figure 5.

(1.9) Line 326: The reference for SIMPOL is wrong. Please correct.

Thanks for catching this, we fixed it.