GMD-2023-45 Response to Reviewers

We appreciate the time and efforts one anonymous reviewer and Dr. Tor Nordam dedicated to providing feedback on our manuscript. We have incorporated or responded to all the comments and suggestions made by the reviewers. Please see below, in blue, for a point-by-point response to the reviewers' comments and concerns. The line numbers refer to the tracked version of our manuscript.

Reviewer 1:

Review of "Intercomparisons of five ocean particle tracking software packages" by Xiong and MacCready

In this manuscript, the authors present a new particle tracking code and compare it to three existing offline codes, one online code, and an online dye simulation. They focus on two regional domains on the west coast of the US. They show that the new code compares well in terms of accuracy and efficiency.

The manuscript is generally well-written. However, I have some significant concerns that prevent me from recommending its publication in GMD. Most importantly:

1. It is unclear what this new Lagrangian tracking code adds. What is the advantage of the new code over the previous codes? What does it add/improve on the other codes? That could be much more explicit.

Response: we thank the reviewer for the suggestion and apologize for the lack of clarity. In this study, we introduced a new offline particle tracking code Tracker and evaluated its performance with other offline and online tracking codes and passive dye. The main purpose is to conduct the intercomparisons of some commonly used codes in the same numerical simulations to explore the net effect of the many slightly different choices made by the different developers. The other offline tracking codes have already been rigorously tested by their developers, and we present our own tests of vertical mixing for Tracker. When choosing a particle tracking code to use, modelers have many considerations. Will the code be easy to use with their model output? Will they be able to modify the code for their specific needs, e.g., introducing vertical behavior? Will it run fast enough? Finally, a modeler should have some confidence that regardless of which code they choose the results will be reasonably similar for all the choices. The goal of this intercomparison is primarily to address this final issue of confidence, and to discuss some of the other tradeoffs. We have endeavored to clarify this in the revised manuscript. We now explicitly describe our goal in Introduction (lines 93-100) and also add more details on the implementation of Tracker in Methods (lines 113-152).

2. The title is far too general, and suggests a much wider scope than that the manuscript can deliver. It's therefore not appropriate for this specific manuscript.

Response: thanks for the suggestion. We revised the title to “Intercomparisons of five ocean particle tracking software packages in the Regional Ocean Modeling System” to be more specific.

3. It's not at all clear why the LiveOcean and Hood Canal are such good testcases for Lagrangian models. I would have expected a much more thorough discussion of why these are specifically suited. Now, it seems as if the authors had these models lying around and decided to do the comparison; instead of selecting an optimal case for the comparison.

Response: the motivation of the present study for particle tracking code intercomparison stems from the authors’ research need to decide which particle code to use for our own ROMS model analysis. During our experiments, we found several open-accessed particle tracking codes could be used but some code like Parcels required re-gridding of the original velocity fields, which requires extra work and will likely introduce some errors in re-gridding. Thus, we limited our comparisons among those codes that can directly operate on ROMS native velocity outputs. As suggested above, we revised the title to be specific to the ROMS model. Given that there is a considerable user group for ROMS and particle tracking is a very useful tool in oceanography, we thought our work of intercomparison could be useful to some part of the research community.

The LiveOcean domain includes deep open ocean, continental slope and shelf, and an inland fjord-type estuary with dynamic sills and quiet deep basins, providing diverse environments to test the preservation of the vertical well mixed condition. The saved LiveOcean model output database (2017-present) is convenient to test the offline particle codes but to do online particle tracking and dye experiment, we found it is more practical to implement them in a small model domain (will be explained more in comment 8).

The reviewer is correct in surmising that we “had these models lying around” and this clearly influenced our choice of experiments. While we did our best to select times and places in the models that we felt covered a useful range of parameter space for the coastal and fjordal ocean, there are clearly many cases we did not test, for example shallow intertidal areas with wetting and drying, river plume fronts, and so on. The “optimal case” for such a comparison is likely to be different for different modelers, hence the definition of globally optimal test cases would involve a much larger number of experiments than we could undertake.

4. It's unclear why some of the choices for the tracker code have been made. E.g., why does it employ nearest neighbour interpolation? That is not very customary for Lagrangian codes. And why then also use 4th order Runge-Kutta integration? Why aim for such high accuracy in time, when spatial accuracy is low?

Response: during the development of Tracker, we tested nearest neighbor and bilinear interpolation and found these two methods gave very similar results but nearest neighbor speeds up the computation in our large model grids. We also tested 2nd-order Runge-Kutta integration and found that a higher-order integrator is required to move particles forward in regions with complex shoreline geometries, like the curving channels in the Tacoma Narrow (in the southern Salish Sea). We justified our choice of nearest neighbor interpolation and 4th-order Runge-Kutta integration in lines 140-144.

5. The argument for smoothing the AK_s diffusivity field is unclear; what is the advantage of this?

Response: we added the argument for smoothing the vertical diffusivity in lines 132-133. Smoothing the vertical diffusivity can reduce the potential sharp gradients in vertical diffusivity that could cause particle aggregations (North et al., 2006).

North, E. W., Hood, R. R., Chao, S. Y., Sanford, L. P.: Using a random displacement model to simulate turbulent particle motion in a baroclinic frontal zone: A new implementation scheme and model performance tests. J. Mar. Syst., 60(3-4), 365-380, https://doi.org/10.1016/j.jmarsys.2005.08.003, 2006.

6. The discussion of the Well mixed condition test in section 2.1.1 could be more elaborate. What is the equation that is tested. Why? How is e.g. the non-significant range defined in Figure 2?

Response: thanks for the suggestion. We added more descriptions about the vertical well mixed condition tests in lines 157-186.

7. One of the most difficulty things to do for Lagrangian codes is boundary conditions near land, and avoiding stuck particles. While the strategies of each code is listed in table 1, there is no discussion of how well the tracker code performs near boundaries. This would be important information for potential users, especially in domains like the Hood Canal where there is so complicated topography.

Response: in Tracker, if a particle gets onto land, it will be moved to a neighboring grid cell with a random direction (please see line 471 in https://github.com/parkermac/LO/blob/v1.1/tracker/trackfun.py). The numerical model does not resolve every process in the nearshore region (waves, rip currents, and so on), therefore, this is a practical way to make sure that particles do not get caught in the boundaries or in corners. We added these details in lines 149-152.

8. Why is there no comparison to online floats or dye in the LiveOcean domain of figure 6 and 7? For a complete picture, that would be useful here too.

Response: we thank the reviewer for this suggestion and tried to run online dye and particle in the LiveOcean domain. However, this requires recompiling and rerunning the model. The version of the model used for the LiveOcean domain used a somewhat dated version of ROMS (the Hood Canal model, and the current LiveOcean forecast use an up-to-date version). The result is that it would be a great deal of work, and significant computational effort, to re-run the LiveOcean model with dye.

While we agree that your suggestion would give a more complete picture, we are motivated by the difficulty of accomplishing it to explore whether it is necessary. We see no reason why intercomparisons between offline particle tracking and online particle tracking/dye experiments in the large LiveOcean domain would give significantly different information than that from intercomparisons among them in the small Hood Canal domain, i.e., that offline particle tracking performs as well as the online tracking. Our attempt here to rerun the large LiveOcean model could also be an example why offline particle tracking is more popular than online tracking in applications. Some studies that applied offline tracking can easily use the precalculated velocity fields without the necessity of rerunning the hydrodynamic model, for example, a recent study using offline particles and the global model ECCOv4 to investigate the global overturning circulation (Rousselet et al., 2021). They may not have the resources available to rerun the model.

Rousselet, L., Cessi, P., Forget, G. (2021). Coupling of the mid-depth and abyssal components of the global overturning circulation according to a state estimate. Science Advances, 7(21), eabf5478.

Other minor comments:

- line 9: Make explicit that these numbers (200m and 1000m) are the resolution and not the domain sizes)
Response: thanks for the suggestion. We deleted these two numbers in the abstract to avoid confusion. The small Hood Canal model domain has a uniform horizontal resolution of 200 m but the large LiveOcean model domain has a changing horizontal resolution from 500 m to 3000 m. In the coastal area of LiveOcean model, in which we conducted the particle tracking experiments, the grid resolution is 1000 m.

- line 15: This sentence is very vague; please rephrase in terms of conclusions/outcomes
Response: We edited this sentence lightly for clarity (lines 14-18), but were unable to boil the "tradeoffs" down to the level of conclusions. This is because the conclusion is really a user's choice based on how the tradeoffs affect their particular model and research needs.

- line 26: Explain why offline tracking is more frequently applied?
Response: we added the explanation in lines 29-31.

- Line 35: There are some recent articles that compare different tracking codes in e.g. the Agulhas: https://agupubs.onlinelibrary.wiley.com/doi/full/10.1029/2019JC015753
Response: thanks for pointing out this reference and we added it in line 49.

- line 38: what is meant here with performance? Speed? Memory? I/O? Accuracy? Reproducibility?
Response: revised in line 53.

- line 40: Is LiveOcean really 'well-established'? What does that even mean, when it comes from the developers of the model?
Response: we apologize for the unclear description and have added the development and calibrations of LiveOcean in lines 61-64.

- line 127: 'studied' instead of 'studies'
Response: revised.

- line 145: what was the convenience why the seeding strategy was different for the online particles?

Response: in the online particle experiment, all 10^5 particles were released inside the same model grid cell as other offline particle codes. Rather than specifying a particle distribution of 100×100×10 (longitude, latitude, vertical) inside the selected grid cell, which is easily to do in the offline codes, it requires significant coding to define a particle distribution of 100×100×10 in ROMS's online tracking file, floats.in. Therefore, we slightly adjusted the seeding strategy and specified the particles' location uniformly along the diagonal of the grid cell, which only requires 1 line of code (an example shown below). This implementation gives the same location of mass center in x, y, and z direction as other offline codes. In addition, the vertical thickness of the selected grid cell is 8.7m, about 5% of the total water depth in this location (~170m). These two release strategies should not induce significant difference to the intercomparison. We added the explanations in lines 230-232.

```
! Number of floats to release in each nested grid.  These values are
! essential because the FLOATS structure in "mod_floats" is dynamically
! allocated using these values, [1:Ngrids].

    NFLOATS == 100000

! Initial floats locations for all grids:
!
!   G      Nested grid number
!   C      Initial horizontal coordinate type (0: grid units, 1: spherical)
!   T      Float trajectory type (1: Lagrangian, 2: isobaric, 3: Geopotential)
!   N      Number floats to be released at (Fx0,Fy0,Fz0)
!   Ft0    Float release time (days) after model initialization
!   Fx0    Initial float X-location (grid units or longitude)
!   Fy0    Initial float Y-location (grid units or latitude)
!   Fz0    Initial float Z-location (grid units or depth)
!   Fdt    Float cluster release time interval (days)
!   Fdx    Float cluster X-distribution parameter
!   Fdy    Float cluster Y-distribution parameter
!   Fdz    Float cluster Z-distribution parameter

POS = G, C, T, N,   Ft0,    Fx0,    Fy0,    Fz0,    Fdt,    Fdx,    Fdy,   Fdz

      1  1  1  100000  0.d0  -123.0008327d0  47.5657658d0  -85.281616d0  0.d0  2.673387e-08  1.8018018e-08  8.738766e-05
```

- line 209: The point that dye spreads faster than Lagrangian particles is not new, and could be related to Markovian dynamics (I.e. dye that enters a grid cell from one side can leave it on the other side within a timestep)?

Response: thanks for pointing it out and we added references (lines 302-303) that observed the same faster dye spreading in their comparisons between dye and Lagrangian particles. For Markovian dynamics, this is an interesting topic, we assume it's related to the advection scheme of dye. Could you point us to the reference? Additionally, numerical mixing was found to account for one-third of the total mixing of salinity in the LiveOcean Model inside the Salish Sea (Broatch and MacCready, 2022), which can also contribute to the faster spreading of dye than Lagrangian particles, especially in regions with strong horizontal gradients.

Broatch, E. M., MacCready, P.: Mixing in a Salinity Variance Budget of the Salish Sea is Controlled by River Flow. J. Phys. Oceanogr., 52(10), 2305-2323. https://doi.org/10.1175/JPO-D-21-0227.1, 2022.

- line 256: 'growing differences in location' is slightly awkward phrasing?
Response: revised (line 362)

- line 278: is the laptop the same as the Apple M1 Pro?
Response: Yes, and we revised it to “Apple M1 Pro” to make it clear (line 386).

- line 284: why does LTRANS scale so poorly for large numbers of particles? This is very surprising for a Fortran code?

Response: to be honest, we don’t know the exact answer. Based on our experiments, LTRANS runs very fast with small numbers of particles but slows down a lot when tracking a large number of particles. It could be due to the algorithm structures, but it is beyond the scope of this paper. Here we hope the computation time may be one piece of information scientists can use when choosing a particle tracking package and designing an experiment.

Reviewer 2:

The authors present a Lagrangian particle tracking tool, Tracker, and compare it to several other existing particle tracking codes, as well as an Eulerian transport model run as an integrated part of ROMS. In general, I find comparisons like this to be very interesting and useful. In my personal opinion, the field of Lagrangian transport modelling would probably benefit from increased attention to details of implementation and comparison between codes. Hence, the topic should (in my opinion) be of interest to the readers of GMD. However, I find that the present manuscript is a bit lacking in some aspects, and my recommendation is that it should be reconsidered after revisions.

The abstract states that Tracker is introduced, and also compared to other models. However, there are also references to previous papers that use Tracker (Brasseale & MacCready, 2021 and Stone et al., 2022), even though these papers do not seem to explicitly use the name "Tracker" (from a quick search of the documents). If the current manuscript is the definitive introduction of Tracker, it should in my opinion contain additional details on the implementation to properly document the model.

For example, are the u and v components of the current assumed to be on separate grid points, and interpolated independently? How about the vertical current component? Is variable horizontal diffusivity supported? It is stated that vertical diffusion uses reflection at the boundaries, but what about edge cases where the particle is so far outside one boundary that reflection would take it outside the other boundary (can happen with non-zero probability, since the random walk uses Gaussian numbers).

Response: we used nearest neighbor for all u,v,w interpolation. We don't use horizontal diffusivity and the assumption here is that the important motions leading to horizontal diffusion are resolved in our coastal model (the horizontal grid resolution ranges from 500 m 3000 m). If the model is not eddy resolving such as a global model, the horizontal diffusion will be needed to include the effects of eddies.

For the edge cases, we enforce limits on vertical reflection using the numpy remainder function if the vertical advection moves particles more than the total water depth. Please see line 488-508 in https://github.com/parkermac/LO/blob/v1.1/tracker/trackfun.py

I would also say that a description of the implementation would be useful for a paper introducing a new model, even if one would hope that the actual performance in terms of accuracy is independent of implementation detail. What python libraries are used? How are the particles stored (simple arrays or custom objects?) Are particle positions recorded in lon-lat or in x-y coordinates in meters? If lon-lat, then how are displacements in meters converted to lon-lat? Are any assumptions made (and hard-coded?) about e.g. the radius of the Earth?

Response: thanks for the suggestion. We now describe more details on the implementation of Tracker in Methods (lines 113-153) and in Table 1.

Finally, I would be interested to see a bit more discussion of some of the choices that were made in the implementation. For example, why combine 4th-order Runge-Kutta with nearest-neighbour interpolation? 4th-order Runge-Kutta has requirements when it comes to continuous derivatives of the velocity field, which are not met by nearest-neighbour. Based on earlier work I have done, I would guess that you could get a better ratio of accuracy to computational effort with e.g. 2nd-order Runge-Kutta if you use nearest neigbour interpolation (see, e.g, https://doi.org/10.5194/gmd-13-5935-2020). This might not be very important in practice, though, so feel free to ignore.

Response: thanks for directing us to this study about the accuracy of different combinations of integrator and interpolation methods in calculating particle trajectories. For our case, we found that nearest neighbor gives very close results to bilinear interpolation but it scales better. It can speed up the computation in a big model grid, much faster than bilinear interpolation. We also tested the 2nd-order Runge-Kutta integration and found it performs poorly in regions with complex shoreline geometry, like the curving channels in Tacoma Narrow in the southern Salish Sea. We describe our choices of nearest neighbor and 4th-order Runge-Kutta in lines 140-144.

I would also like to see more discussion on the vertical diffusivity implementation. Why was a 3-point Hanning window chosen? And it says that this is applied in the calculation of the vertical derivative, but does that mean that the smoothing is not applied for the diffusivity values themselves? I think these values should be chosen consistently, otherwise you might have trouble with the well-mixed condition. Did you do any testing of this point? Note also that the well-mixed condition is only a neccessary condition, not a sufficient one. A perhaps more stringent test of the implementation would be to compare to a dedicated 1-D solver of the diffusion equation with variable diffusivity. The comparison to the dye in Fig. 5 is good, but since this is a 3D case with comparison only in the vertical it is a bit hard to reason about the cause of the discrepancy.

Response: we followed North et al. (2006) with the 3-point Hanning window to smooth the vertical eddy diffusivity profile and the smoothed vertical profile was then used to calculate the vertical derivative. We also tested 4-point and 8-point Hanning windows to see how well they can satisfy the vertical well mixed condition (WMC) test and found they gave similar results to the 3-point Hanning window method. In lines 132-134, we reworded the sentence to make it clear on the usage of Hanning window to smooth the vertical derivative.

The purpose of the vertical WMC tests for particles is to see if Tracker can reproduce the Eulerian solution to the 1-D vertical diffusion equation: $(\frac{\partial C}{\partial t} - \frac{\partial}{\partial z}\left(K\frac{\partial C}{\partial z}\right) = 0$, where K is eddy diffusivity, with an initial uniform concentration ($C(z) = C_0$) and no flux boundaries ($K\frac{\partial C}{\partial z} = 0$). In the setup of WMC test in Tracker, we uniformly released 4,000 particles in the water column and turned off both horizontal and vertical advection. Particles only move vertically, and the vertical location is controlled by $z_{n+1} = z_n + \frac{\partial AK_s}{\partial z}\cdot\Delta t + R\sqrt{2AK_s\cdot\Delta t}$. To satisfy the WMC test, the initially well-mixed particles are expected to remain uniform in a statistical sense regardless of the diffusivity profiles. Therefore, the WMC test applied here is like a 1-D solver of the diffusion equation. We add more details about the WMC tests in lines 157-186.

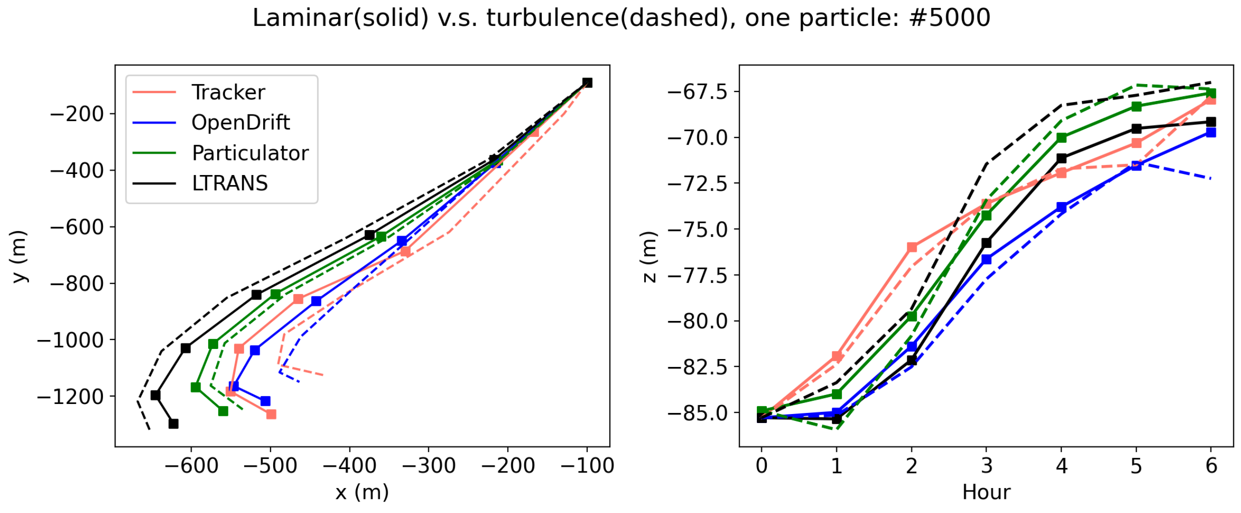
North, E. W., Hood, R. R., Chao, S. Y., Sanford, L. P.: Using a random displacement model to simulate turbulent particle motion in a baroclinic frontal zone: A new implementation scheme and model performance tests. J. Mar. Syst., 60(3-4), 365-380, https://doi.org/10.1016/j.jmarsys.2005.08.003, 2006.

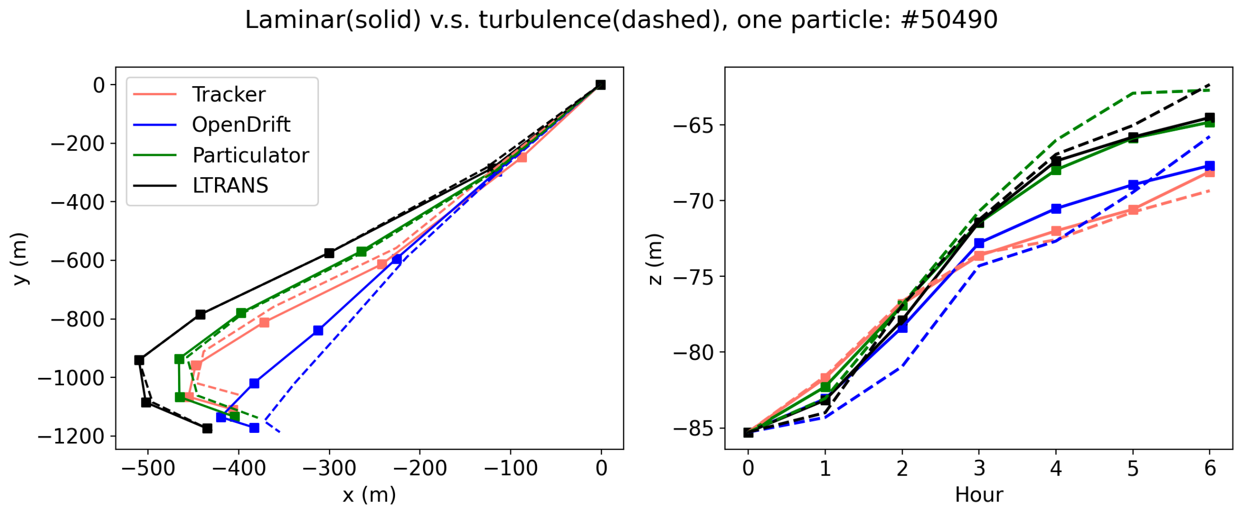
The second point of the paper is a comparison of different particle models and a dye study. Here, I would have liked to see a bit more discussion and investigation of details. For example, the paper states that "a very good inter-model match was achieved", with reference to Fig. 3. However, I would say that there is something odd in Fig. 3, as the trajectories of the centers of mass already from the start appear to separate very fast. This should be straightforward to investigate further, for example by running a single timestep with a single particle and no diffusivity, and checking if all the models move the particle the same distance and direction. Certainly those models that use the same interpolation and numerical integrator should have exactly identical behaviour in the case of no diffusivity.

Response: we thank the reviewer for the interesting comment and conducted the suggested experiments by turning off diffusivity for all four offline tracking codes.

In the figures below, we used 2nd-order Runge-Kutta for OpenDrift to make its advection and interpolation schemes the same as Particulator. The time step for particle tracking is 300 s and particle locations were saved every hour. Each dot in these figures below represents every hour. We expected OpenDrift and Particulator should give the same trajectory after turning off diffusion, but they sort of separate from each other in the first hour after 12 steps of tracking. Even though we selected the same interpolation and integration methods in different particle tracking codes, we are not sure if their interpolation/advection are implemented in exactly the same way. It is beyond the scope of the paper to evaluate all the numerical details. The roundoff error for different programming language might also contribute to the difference. Whatever the source of small differences, the differences generally accumulate over time as particles experience different advection and mixing, making the comparison ever less “direct” as time goes on. This decorrelation scale is an intrinsic problem of Lagrangian analysis.

Additionally, if two particle tracking codes applied exactly the same numerical methods and the same algorithm structures, we will expect them to give the same results, but if so, the results are not of any research interest. In this study, we are trying to look at several commonly used tracking codes that have many differences, but all of them perform similarly when tested in the same circulation model. The comparison of multiple tracking codes in the same circulation model output establishes confidence in all, and allows comparison of other factors such as their computational speed and ease of use.





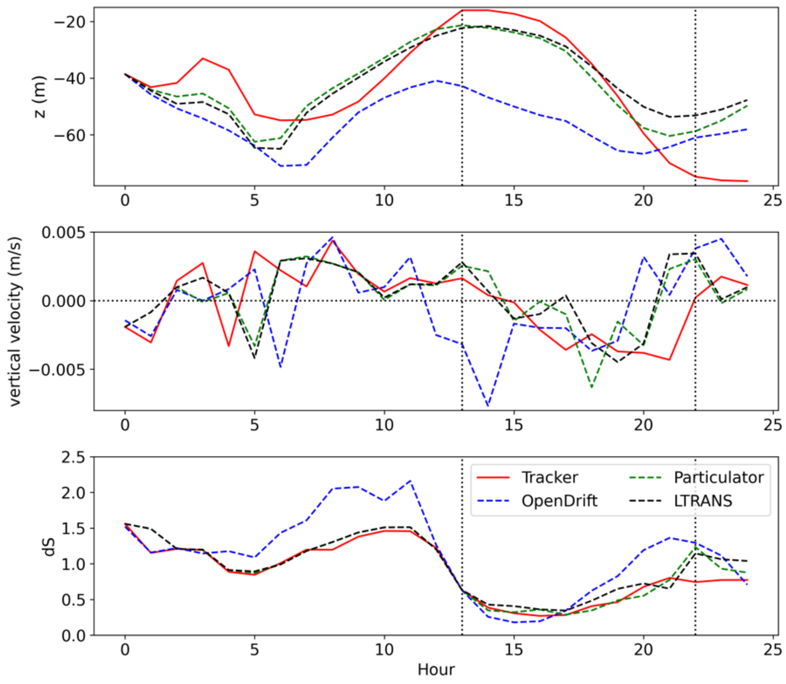
It is further stated that "vertical distributions of particles (Figure 5) exhibit similar evolutions among all tracking codes", whereas I would say that the distributions are quite different. In particular, Tracker shows quite a spikey distribution, and also the dye study has a lot of spikes in the distribution. This last point was particularly surprising to me, as I would have expected an Eulerian diffusion solver to produce a smoother concentration field than a particle based model. Of course, as this is 3D with advection, that makes it a bit harder to reason about, but this could be investigated. In any case, I would say that it is clear from the left panel of Fig. 5 that these models are _not_ equivalent, or at least not run with equivalent setups.

Response: we agree with the reviewer that there models are not run with equivalent setups because they used different interpolation and integration schemes. They also have different ways to process vertical diffusivity.

For the spikes in the vertical distribution of dye, the 3D advection-diffusion process could be one reason. Another reason could be due to the way we did the post-processing of the vertical position of dye (now described in greater details in lines 296-300). To obtain the histogram of vertical dye distribution, dye mass inside each model grid cell was first converted to an equivalent number of particles. Then the vertical coordinate in the center of the grid cell that contains dye was used to represent the vertical location of dye-converted particle number. This conversion might have led to the spiky vertical distribution in the early stage of dye transport as seen in Figure 5a.

Looking at Figs. 6 and 8, I am a bit surprised about the large vertical fluctuations in position, particularly in the mid-watercolumn release. Panel g in Fig 6 shows that with Tracker, the center of mass moves more than 60 meters downwards in less than 12 hours, which is a much larger displacement than with any of the other models. I'm also curious about what the mechanism behind this transport is. How large is the vertical current component? How stratified is the water column? Discussion on this point would be appreciated.

Response: we thank the reviewer for pointing this out. In the figure below, we plotted the vertical center of mass for particle trajectories calculated from each offline codes, the vertical velocity of the model grid cell that includes the center of particle mass, and the bottom and surface salinity difference (dS) of the water column. The large downward transport in Fig. 6g happened around hours 13-22 and the averaged vertical velocity at the center of particle mass is -1.8×e-3m/s (Tracker), -1.6e-3m/s (OpenDrfit), -3.6e-4m/s (Particulator), and -2.3e-4m/s (LTRANS). Negative values mean downward. The stratification is also weaker for particles tracked by Tracker, which may because the horizontal center of mass calculated from Tracker is closer to the coastline during this period (Fig. 6h). Therefore, large downward vertical velocity and weaker stratification could lead to the large vertical displacement of particles tracked by Tracker. Generally, if the horizontal advection (due to different interpolation and integration methods) leads particles to different places, they might experience more (or less) vertical advection (lines 348-353).



Finally, I think it would be good in the interest of reproducibility to provide the actual setups used for the different models, perhaps in a separate github repo for the paper. That would make it much easier for others who might want to look into the comparison.

Response: we thank the reviewer for the suggestion, yet it is very challenging to upload the test data (~200 GB in total) to an online repository. For reproducibility, it may be more relevant for the interested readers to set up their own particle tracking experiments using all these open-access tracking codes. In this study, as stated in lines 199-202, we only consider the advection of passive particles and vertical turbulence (without any particle behaviors), which are generally the most basic set up for a particle tracking model.