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 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.

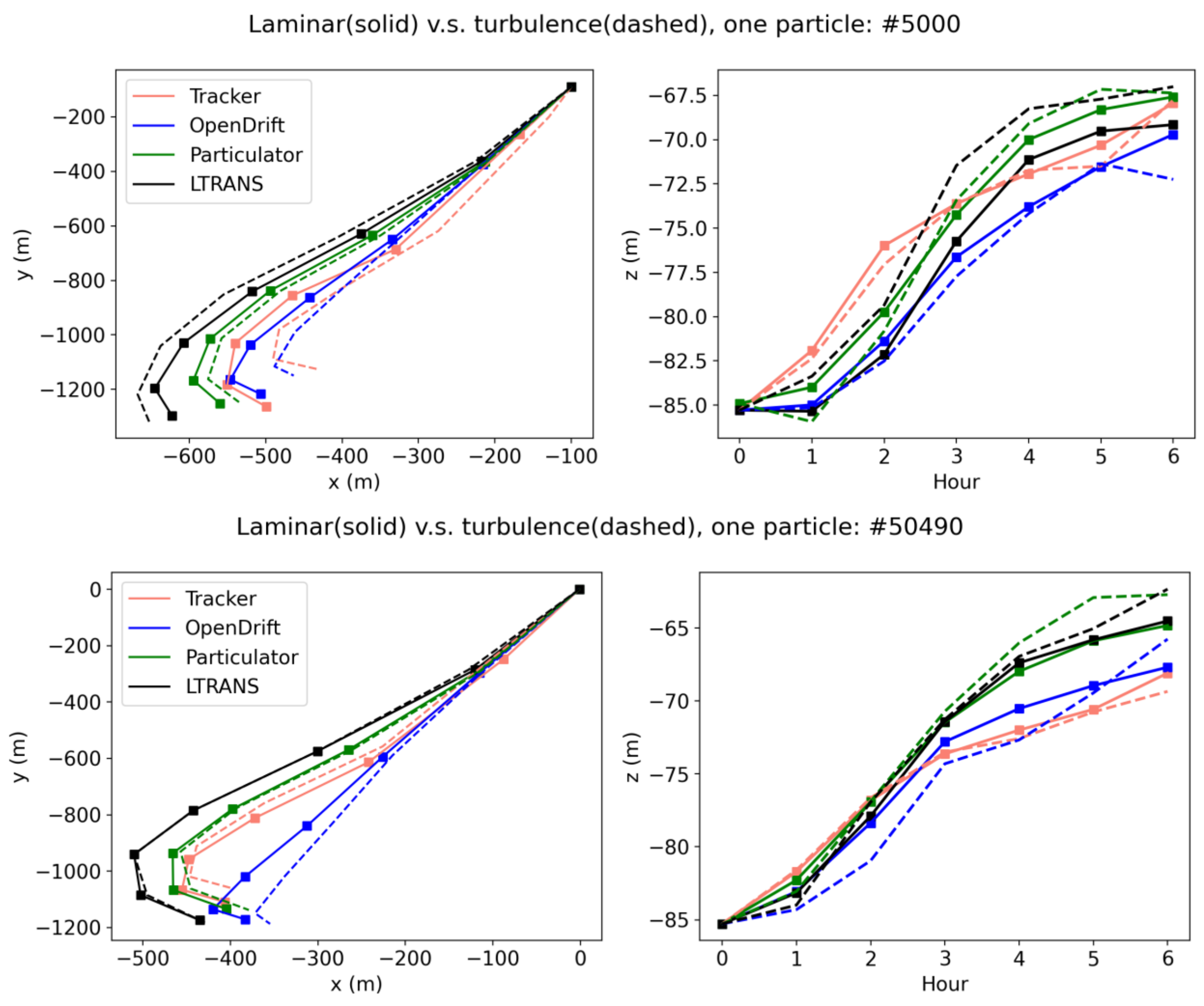
North, E. W., Hood, R. R., Chao, S. Y., Sanford, L. P.: Using a random displacement model to simulate turbulent particle motion in a baroclinic frontal zone: A new implementation scheme and model performance tests. J. Mar. Syst., 60(3-4), 365-380, https://doi.org/10.1016/j.jmarsys.2005.08.003, 2006.

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.



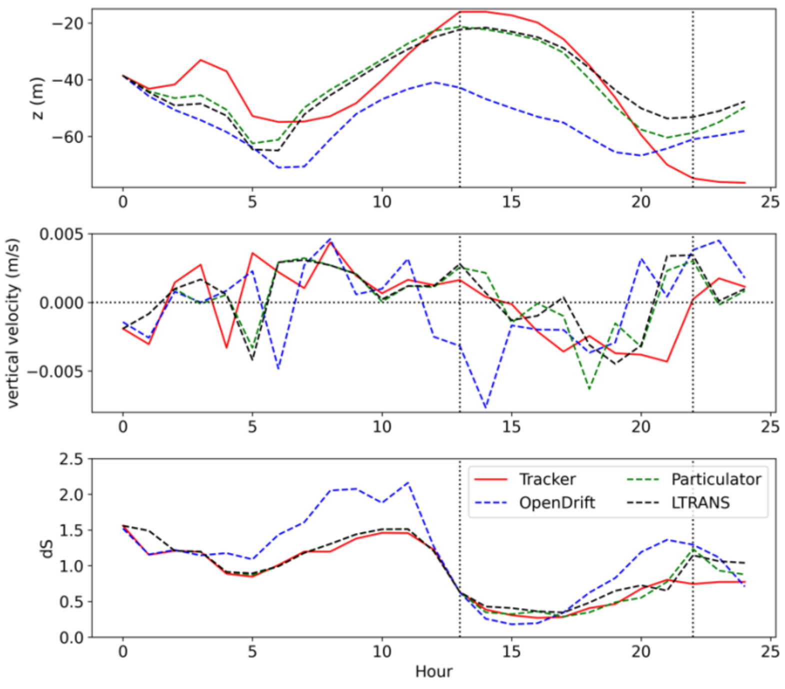
It is further stated that "vertical distributions of particles (Figure 5) exhibit similar evolutions among all tracking codes", whereas I would say that the distributions are quite different. In particular, Tracker shows quite a spikey distribution, and also the dye study has a lot of spikes in the distribution. 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.