03/05/2024 13:39:00“Modelling chemical advection during magma ascent” by Dominguez et al. is an interesting manuscript that compares four different numerical schemes for phase field advection, emphasizing their suitability within the context of two-phase flow models. I believe the manuscript is valuable for the earth sciences modeling community, as it assesses the behaviour of these advection schemes in terms of accuracy -here mass conservation- and computational cost.

The upwind and SL schemes are quickly disregarded as they introduce excessive numerical diffusion and have a low accuracy. The authors find WENO as the most suitable advection scheme as it conserves mass at a decent computational cost, arguing that MIC does not conserve mass efficiently well in one of their two numerical experiments and is far too expensive in terms of performance and memory consumption.

Below are some general comments regarding the manuscript, followed by detailed line by line comments.

Albert de Montserrat

General comments:

(1) The authors should provide more details about the interpolants used in both the Semi-Lagrangian and MIC schemes. They also mention that complex boundary conditions may require some extra care in the WENO scheme. It is not clear to me what these complex boundary conditions are. Do commonly used boundary conditions such as free-slip or no-slip require some special treatment? Since the periodic boundary conditions used in the examples given in the manuscript are not the only common ones, a more detailed discussion would be helpful.

Response: Bicubic B-spline was used for the semi-Lagrangian scheme and a weighted-distance averaging bilinear interpolant was used for MIC.

→ Details about them were added in the manuscript, in particular about MIC as it is a less conventional interpolant.

We agree that the mention of “complex” boundary conditions is confusing and misleading. Common boundary conditions, such as Neumann and Dirichlet boundaries can be applied with WENO schemes, commonly done by using ghost points. The “complexity” mentioned refers to the fact that, as it is a 5 points stencil, 2 ghost points are required on each side of the model for boundaries other than periodic.

→ This has been clarified in the text and the confusing notion of complexity has been removed.

(2) The rotating circle benchmark shows that WENO and MIC produce similar results in terms of mass conservation. However, WENO performs better in conserving mass in the two-phase flow example. It's important to note that in the first benchmark, WENO SL, and MIC are run with the same constant time step. In the other example, the Courant number for MIC was more than twice that of WENO's. To ensure a fair comparison of how well these schemes conserve mass, I suggest running the two-phase flow example with the same time step for

WENO, MIC and SL. This will make the comparison of mass conservation fairer. If the mass conservation of MIC and SL improves by using the same time step as WENO, it would be helpful to run both MIC and SL with increasing time steps to determine when their accuracy starts to deteriorate.

Response: → Following this suggestion, both models with SL and MIC were performed using a Courant number of 0.7 and 1.5. In addition, 2 implementation mistakes were found concerning the QMSL and MIC algorithms. This improved the general mass conservation of the algorithms. This will be added in the manuscript. However, the main conclusions of the paper remain unchanged.

(3) The manuscript concludes that, besides better mass conservation, the WENO scheme is the best candidate because of the poor performance of MIC and its excessive memory consumption. However, the comment stating that the MIC does not run at high resolution because it does not fit in 128 GB of RAM is surprising and unclear. Let's assume that one allows a maximum of 30 particles per cell, then in the higher resolution case you have an upper bound of 15e6 particles (500 x 1000 x 30). Further assuming that you are using FP64, this means that only 120MBs (in Julia: sizeof(Float64)*15e6 / 1e6) are needed to store a particle field. With this in mind, there is plenty of RAM available to store multiple fields in the particles, and even for a much larger number of particles. Last, all the necessary arrays can be pre-allocated so that all the MIC kernels (advection, interpolation, injection….) can be performed in-place without requiring additional memory

Inspecting your MIC algorithm with packages such as Cthulhu.jl and JET.jl will reveal type instabilities in the code, which result in dynamic dispatch and potentially generate a large number of unnecessary allocations. Available profiling tools can be used to track down where the remaining allocations come from.

The main point of this paper is not addressing the optimization of the implementation of any of the advection schemes here present. However, since performance is one of the criterions chosen by the authors to select the best candidate scheme, I would suggest that the authors consider resolving the main performance issues in their codes. By doing so, the MIC's performance is also likely to improve, enabling it to run the two-phase flow example at higher resolutions. This would allow for a better comparison between MIC and WENO, and potentially turn MIC into a more viable candidate.

Response: → following this comment, the main problem of memory in MIC was tackled and multithreading on CPU was implemented on all algorithms to further speed up the computation and allowing to perform all models at high resolution. The main slow down on MIC is now the injection and removal of markers that occur at almost every timestep in the two-phase flow case due to the highly divergent melt velocity field. The authors did not manage to fully parallelise this part as it requires to change the size of current vectors holding marker properties at run time (either to remove or to add new markers) as the high number of marker accumulating doesn't allow to preallocate them in a heuristic way.

(4) Attached is a PDF with a series of typo, grammar and style corrections and suggestions.

Line by line comments

L5: “weighted essentially non-oscillatory”. Throughout the manuscript WENO is spelled out with either lower or upper case initials (e.g. line 125). For consistency, I recommend to use either one or the the other throughout the entire manuscript

Response: → This has been corrected.

L87 “It consists at” -> “It consists of”

Response: → This has been corrected.

L88 “to interpolate” -> “interpolating”

Response: → This has been corrected.

L88 “mesh grid”. I would call it either mesh or grid.

Response: → The word “grid” is now used consistently.

L89 “to be” -> “being”

Response: → This has been corrected.

L133 “element” what is the element here?

Response: It is here the chemical element.

→ It has been clarified in the manuscript.

L147 You could consider using the machine precision epsilon (function eps() in Julia) instead of fixing epsilon=1e-6

Response: → This will be added in the manuscript.

L153 “Careful consideration must then be given to boundary conditions for complex problems.” What kind of considerations have to be made for the boundary conditions? It would be helpful to elaborate on this. I don’t think it is obvious.

Response: → This has been reformulated, as described in response to (1).

L166, Isn’t the time integration third order in space and first order in time? If I’m reading correctly eqs. 11-13, C is evaluated at the same time step in all the integration stages.

Response: A time integration will only impact the order in time. You can see C^2 and C^1 as being intermediate timesteps with the coefficients in front of them and hence being third order in time. It should be seen as a modified 3rd order Runge Kutta scheme. For more detail, Gottlieb et al., 2001 provide the original derivation (eq 4.2 for this scheme in the original paper).

L176 “rectilinear grid” perhaps “uniform” or “regular” are more appropriate

Response: → modified to “regular grid”.

L176 ”this reduces the complexity of the implementation and the numerical cost of the interpolation function.” Could you briefly describe why this is so (finding the parent cell, cheap mapping to the reference cell, etc…) ? It may not be obvious for a reader who never tried to implement any of these advection schemes.

Response: → An explanation will be added.

L177 “From a particle point of view, the goal is to find the starting points at the previous timestep for each grid point.” Perhaps this could benefit from some rephrasing, what you want to do is to find where a particle that is at a grid point (i,j) at time t^n+1 was at the time t^n

Response: → This has been rephrased in the revised manuscript.

L183 What about higher order methods such as Runge-Kutta 4 ? would they work better to back track the particles?

Response: To our knowledge, mainly higher order linear multistep methods have been explored. For example, recently by Filbet and Prouveur, 2016 for backwards SL. They showed that using Adams-Moulton or Adams-Bashforth schemes improves the mass conservation and allow to take larger time steps when the resolution is big enough. This approach has the downside of also requiring values for the velocity at t^n-1. Runge-Kutta methods are not popular for this problem and the authors believe it is essentially for historical reasons, as SL schemes were originally developed for modelling the atmosphere (see Pursue and Leslie 1995 for more details on the advantages of multistep methods for this purpose). This could be interesting to explore as it may impact the conservation of mass of the scheme but this it is beyond the scope of this paper.

→ A more concise version of this answer was added in the discussion of the manuscript.

L190 How is vf interpolated to t^n+½? is it just the average between t^n and t^n+1? I guess this requires storing the velocity at two time-steps, this should be mentioned somewhere in the manuscript.

Response: It can be either the average of t^n and t^n+1 or extrapolated from t^n-1 and t^n if the value at t^n+1 is not accessible. In this study, the mean is used as the momentum equations are solved prior to the advection equation.

→ A mention of this work has been added to the manuscript.

L197 Are bi/tri-linear interpolations not good enough?

Response: Using linear interpolation is equivalent to using an upwind scheme. Actually, for a Courant number lower than 1 and for a constant velocity field, it can be shown than upwind and linear SL schemes are equivalent (see Brasseur and Jacob, 2017 p.331 for instance).

L205 “clipping”, I think “clamping” is a better word for this.

Response: Clipping is the common term used in the literature for this scheme. It is also consistent with the idea of limiting the maximum value.

L205-210 The manuscript lacks the definition of C^H. It would be helpful to include this information.

Response: As mentioned on line 205, C^H is a high order interpolant. The definition is left as general as possible so it can be a cubic spline, or a fifth order hermite interpolation for example.

→ It has been clarified that it corresponds to a cubic spline in this study.

L214 “interpolate their values on a fixed grid.” Interpolation often goes in both directions (e.g. temperature), not only from the markers onto the grid.

Response: → This is now mentioned in the manuscript.

L220 Are the initial positions of the markers randomized? If they are, it should be mentioned later on the benchmarks descriptions.

Response: They are not.

L220 “The initial value of each marker can then be obtained by linear interpolation from the initial conditions of the Eulerian grid.” Most of the time, a field can be initialized directly at the particles, which provides a more accurate starting point

Response: → Correct, this will be clarified in the manuscript.

L223 “a **non-conservative** strategy”

Response: → This has been added.

L246-147 “This is more complex than for SL schemes because the markers are not uniformly distributed for a non-trivial velocity field.” Another disadvantage is that the parallelizing this interpolation is susceptible to race conditions in shared memory systems, requiring the use of atomic operations.

Response: → This comment was added in the manuscript.

L252-253 Could you add eq. of the linear interpolant?

Response: → This will be added.

L263 How is the computational time of one time step measured? Is it the time of the first step? If so, does it include compilation time? Or is it the average step time throughout the whole model?

Response: The computation time is measured by running each scheme with the initial conditions 10000 times. The reported time is the minimum elapsed time measured (using @btime from Benchmark.jl) so it does not include the compilation time.

→ This procedure has been clarified in the manuscript.

As the number of particles varies throughout the model run time, it may be better to average the time of each time step (excluding the first one to avoid measuring compilation time, or doing a warm up run) and add a confidence interval.

Response: There is no reseeding or removal in the test as the number of particles doesn't go below the threshold of 25% of the original number of markers per cell and don't accumulate. We think that repeating the MIC with the initial conditions 10000 times is sufficient to capture the representative computational time of the MIC.

Would be helpful to clarify what is actually done and measured in the MIC case. I believe that you are interpolating from particles to grid (plus reseeding?) in each time step. This is technically not needed in this case, since you are merely advecting passive markers. It may be helpful to include in the manuscript a breakdown of the percentage of time spent on all the MIC stages done in the benchmark.

Response: The measured time include the interpolation from particles to grid.

Table 2.

- Here you call it "SL QM", however, it's written as "QM SL" in some other places. Please choose one naming convention for the whole manuscript.

  Response: → This has been corrected.

- I get very different timings for SL and MIC (Julia 1.10, Windows 11, CPU: 24 × AMD Ryzen 9 5900X 12-Core Processor). Different Julia versions are known to have different performances, it would be helpful to add to the caption what Julia version was used. Later on you mention what CPU you used, but I think it would be nice to have this information also in this caption.

  Response: The timings were corrected, with single and multithreaded cases added. → The Julia version (1.10.2) and the CPU details were also added to the caption.

Section 4.2 Perhaps it is better to define the scaling lengths in a small table instead of spelling them out line by line.

Response: → The scaling variables has been added to Table 3 and removed from the text.

Section 4.3 The time step of QMSL and MIC is roughly ~x2 with respect to WENO and Upwind. It may be beneficial to run QMSL and MIC with the same Courant number as in the other two cases and include the mass conservation results to Fig 9.. This could potentially reduce the reseeding in the MIC case and improve the mass conservation.

Response: → Following this concern, both QMSL and MIC were also performed with a Courant number of 0.7. Results will be included on Fig 10 and 11.

L349 “This method” -> “This package”

Response: → This will be corrected.

L354 As a curiosity, what kind of automatic differentiation is being used?

Response: Forward automatic differentiation using ForwardDiff.jl.

→ This was added to the manuscript.

L360 “assert” means to confirm something. Perhaps “examine”, “test”, or similar, work better here.

Response: → It will be replaced by assess

L365 It is stated later that all the runs are running serially in a single core, perhaps it’s useful to mention that fact here instead/as well.

Response: → The models are now performed using multithreading. This will be mentioned there.

L399 “that is linked on how often the two-phase flow solver is called” perhaps you could rephrase it so that it is more clear that being able to increase the time step results in fewer time steps being required.

Response: → This has been clarified.

L400 “...QMSL shows the best scaling…” What do you mean by scaling here? Scaling is usually referred to as how well one algorithm scales with the number of cores/CPUs. However, all the models were run serially in a single core.

Response: The scaling referred here to the increase in resolution as QMSL was the fastest algorithm at high resolution due to the high Courant number used.

→ The word scaling was removed to clarify the meaning.

L423 “If complex boundary conditions are required” As in my previous comment, I think it is not obvious what these complex boundary conditions are.

Response: → This has been reformulated, as described in response to (1).

References:

Gottlieb, S., Shu, C. W., & Tadmor, E. (2001). Strong stability-preserving high-order time discretization methods. *SIAM review*, *43*(1), 89-112.

Filbet, F., & Prouveur, C. (2016). High order time discretization for backward semi-Lagrangian methods. *Journal of Computational and Applied Mathematics*, *303*, 171-188.

Brasseur, G. P., & Jacob, D. J. (2017). *Modeling of atmospheric chemistry*. Cambridge University Press.

Purser R. J. & Leslie L. M. (1995). Accuracy and conservation in semi-Lagrangian time-integration. *ECMWF*