Author responses to the Interactive discussion on "An explicit GPU-based material point method solver for elastoplastic problems (ep2-3De v1.0)" in the Geoscientific Model Development (GMD) Journal

Emmanuel Wyser, Yury Alkhimenkov, Michel Jaboyedoff, Yury Podladchikov

September 23, 2021

The referee comments appear in black, whereas our responses appear in blue and the changes made in the revised manuscript appear in red.

1 Reviewer #2

This paper demonstrates the numerical implementation of the explicit material point method using GPU for parallel computing. In Reviewer opinion, the paper is of interest for the Readers of Geoscientific Model Development. However, a number of issues need to be addressed before the paper can be accepted for publication.

We would like to acknowledge the reviewer for the time spent on the revision of our work.

Comment # 1 The paper is currently long and several part can be replaced by references. For example, the MPM algorithm is derived from forward-Euler scheme with update stress lass, GIMP basis functions in appendix A.

Reply # 1 Thank you for raising that point. We recognize that the paper can be long for the audience. However, we also think it is the bare minimum to have a grasp over the material point method. We strongly believe this avoids the reading of numerous references and facilitate the overall understanding of the method to a broader audience. We already attempted to shorten the section 2 by putting in Appendix a detailed description of the shape functions used in GIMPM.

Change # 1 -

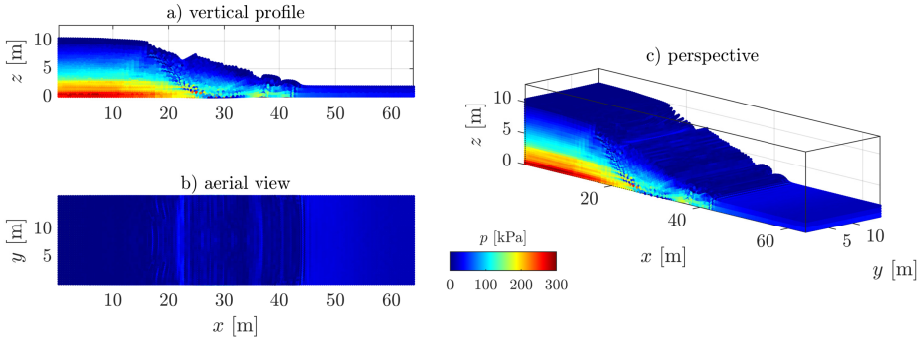
Comment # 2 There are several methods dealing with the volumetric locking in the literature. However, the author proposed the volumetric locking by averaging only the volumetric part of the stress tensor. The author is suggested to clarify the decision for that. Furthermore, volumetric locking can smooth out the value of the stress. It is better if the stress is plotted in the numerical examples to see the difference between simulations with and without volumetric locking.

Reply # 2 It is true that in the literature, the stress tensor $\boldsymbol{\sigma}$ is averaged (Mast et al., 2012; Cuomo et al., 2019; Lei et al., 2020). Volumetric locking procedures generally mitigates locking by a more appropriate formulation of the volumetric component of the strain. The B-bar method ($\bar{B}$, see Bisht et al. 2021) splits the strain tensor into deviatoric and spherical parts. Deviatoric strains are evaluated at the material point's coordinate whereas the spherical part is evaluated only at the element center's coordinate. This yield a stress tensor, for which the pressure is defined at the element's center.

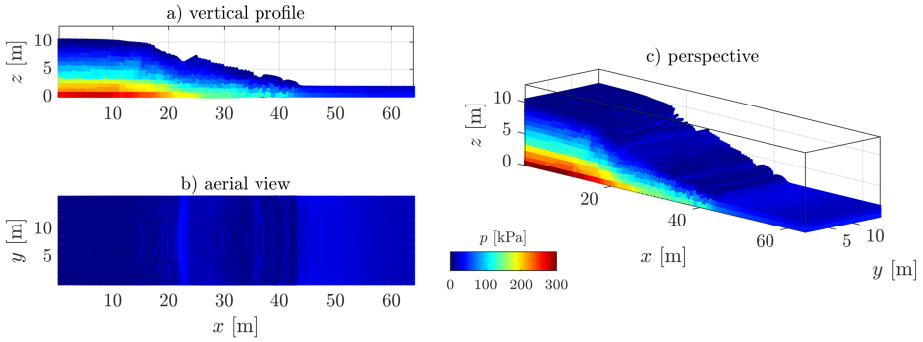
As such, we modified the element-based strategy (Mast et al., 2012; Cuomo et al., 2019; Lei et al., 2020), considering only the pressure term, which need to be averaged at the element's center. We believe this formulation is more consistent with the essence of the B-bar method, which splits the deviatoric part from the spherical part.

It is true that volumetric locking can smooth out the value of stresses. As suggested, we will add plots to show the difference between a locking-free and a locking-prone solution in the revised manuscript in the Appendix C: Volumetric locking and damping corrections.

We here show an example of slumping considering the geometry as in Model 2b. We selected an homogeneous initial cohesion field with values presented in the submitted manuscript. Figure 1 demonstrates that a significantly smoother pressure field is resolved with the proposed method. In addition, the pressure field is smoothed but it does not significantly differs from the original pressure field (in locations where locking is minimum). Volumetric locking is particularly highlighted within shear bands due to isochoring plastic flows, resulting in significant stress oscillations.



(a) Non-smooth pressure field due to volumetric locking



(b) Smooth pressure field when volumetric locking is mitigated with the proposed solution

Figure 1: Element-based reconstruction of the pressure field to mitigate volumetric locking issues, with a total number of material points $n_{mp} \approx 10^5$.

Change # 2 -

Comment # 3 Section 4 mentions that Model 1b demonstrates the influence of mesh resolution but I do not see it in Model1b. The author is suggested to perform convergence rate analysis in different mesh size in the plane strain to highlight the influence of the mesh resolution.

Reply # 3 We understand the concern of the reviewer. It is true that the mesh resolution is not clearly demonstrated in Model1b under plane strain conditions.

We here present additional plots (see Fig. 2) to show to the reviewer the influence of the mesh resolution over shear banding. All the parameters used are the same as described in the original manuscript, except that the number of element in the x-direction is increased, *i.e.,* $n_{el,x} = 40, 80, 160, 320$. We can clearly

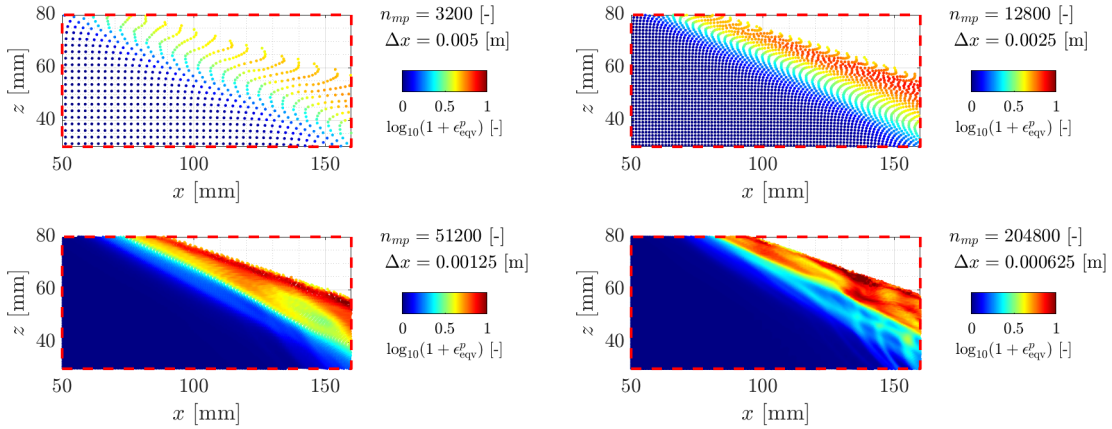


Figure 2: Equivalent plastic strain for a variety of mesh resolution. One can see the influence of the mesh resolution over shear banding.

observe that as the mesh resolution increases, the shear band resolution gets more accurate, *i.e.,* the finer the resolution the finer the shear band thickness. We also observe a more complex shear banding pattern at the bottom right, *e.g.,* $x > 140$ mm and $y < 50$ mm. Again, such shear banding arrangement is better resolved with a finer mesh resolution. We believe Fig. 2 clearly demonstrates the influence of the mesh resolution. Therefore, we do not think a convergence analysis is further needed.

Change # 3 -

Comment # 4 For Model 1b, the presented final geometry of the experiment is shorter to the one in Bui et al. (2008) experiment (see Figure 6) in my opinion. Please check. Therefore, it is not necessary to introduce the damping.

Reply # 4 Thank you for this remark. We checked and the reviewer is right. We performed additional simulations and still noticed that simulations without damping leaded to higher run-outs and softer response of the material. We will clarify this in the revised version of the manuscript.

Change # 4 -

Comment # 5 In Model 2, there is a boundary effect on the failure mechanism as the shear band can touch the bottom boundary. It would be better if there is a larger depth in the bottom direction. And, the Model 2 introduces local damping which in my opinion it is not necessary.

Reply # 5 We acknowledge the boundary effect on the failure mechanism. However, our concern was to show an example of large deformation. A larger depth would also significantly reduces the total amount of deformation. Model 2 shows that several elasto-plastic features are resolved, such as bulging or thrusting at the toe of the slope. Such features would not be as obvious when a larger depth is considered. Concerning the introduction of a local damping, it is mainly to prevent an excessive run-out of the material when considering free-slip boundary conditions at the bottom of the computational domain.

However, we show here an example of a similar setting while considering a larger depth, as suggested by the reviewer. We briefly describe parameters that needed to be changed regarding the increase of the depth.

Regarding the initial geometry, we increase by a factor of two the z-direction and the $y-$direction. To avoid a significant elastic compaction of the material, we selected a higher Young's modulus, i.e., $E = 10$ MPa. This reduces the amount of vertical elastic compaction of the material during the elastic loading stage. No-slip boundary conditions are enforced at the base of the material. As thought by the reviewer, the local damping can be reduced. But a small value is still needed to suppress elastic wave propagation. In this case, we selected a damping value $D = 0.025$. An isotropic Gaussian random field is still selected for the cohesion field, with the same parameters used for Model 2b in the submitted manuscript.

Figure 3 shows the total displacement field after plastic yielding. In comparison with results in Model 2b, the magnitude of deformation is smaller. For instance, the intense bulging reported in the original manuscript is less evident in this setting. However, we still report thrusting mechanisms at the toe of the slope. Heterogeneous displacements are still observed, more evidently at the toe.

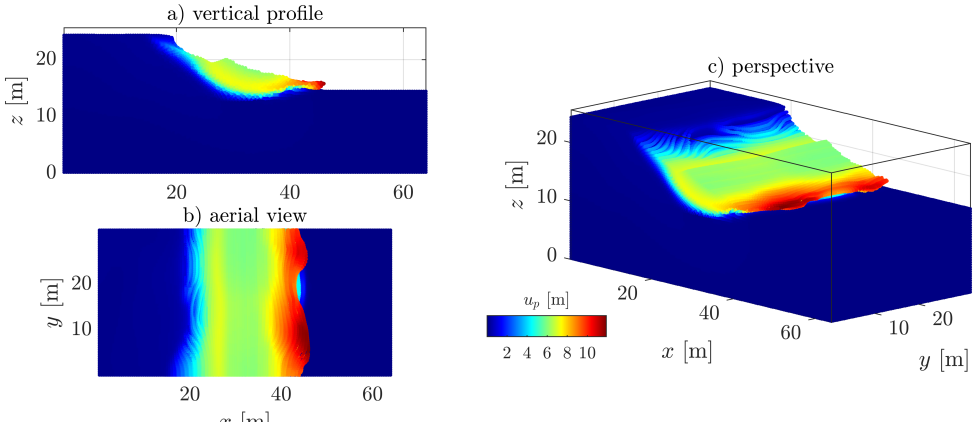


Figure 3: Total displacement after elasto-plastic loading.

We also report a principal shear band and a heterogeneous crown-like structure (see Fig. 4). Plastic strain localisation differs regarding what was reported in the original manuscript. However, other simulations revealed that multiple shear bands can initiate successively. In Fig. 5, we can observe both shallower and deeper shear bands.

However, the deeper shear band is more developed than the shallower one. Similarly, we also report a more significant heterogeneous displacement field and a deeper thrusting mechanism at the toe of the slope. This is because of the random generation of the cohesion field, which can results in weaker and/or stronger local cohesion values. This also demonstrates an important influence of the initial cohesion field over the elasto-plastic response of the material.

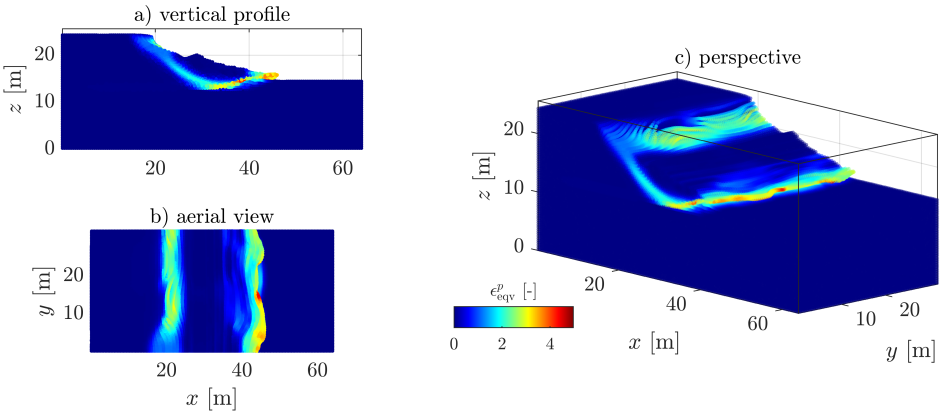
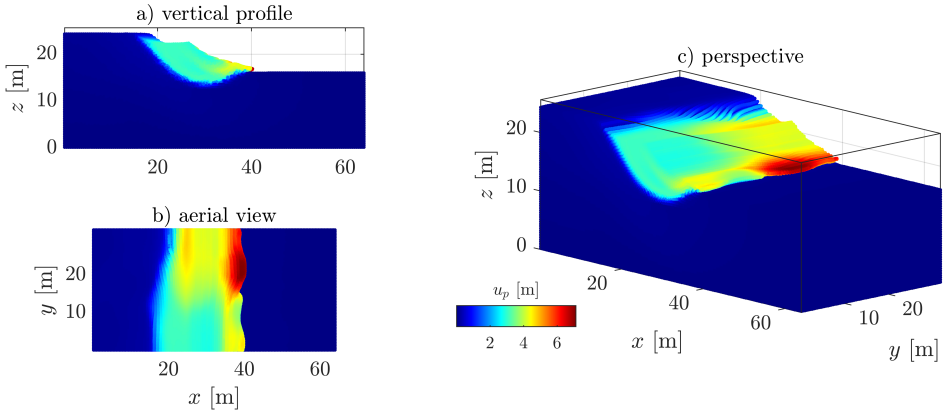


Figure 4: Equivalent plastic strain after elasto-plastic loading.

Still, we think that the original results presented in the manuscript better represent the different mechanisms during the elasto-plastic deformation of an unstable material. It is true that increasing the depth greatly reduce the influence over the shear band propagation. We will clarify and discuss these considerations in the revised manuscript. We will also notify the reader that the boundary influence the propagation of the shear band due to the shallower depth mentioned by the reviewer.



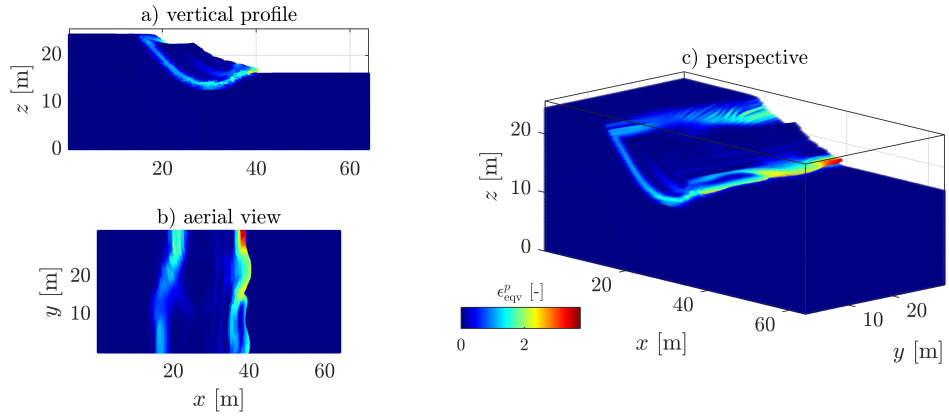


Figure 5: Total displacement (up) and equivalent plastic strain (down) after elasto-plastic loading.

Change # 5 -

Author comment # 6 We additionally extended the original single GPU code and implemented a multi-GPU version using the message passing interface standard. We provide the new section below, which then will be included (with some simplifications) into the manuscripts during the revision stage.

The Multi-GPU Code Implementation

Introduction

One of the major limitation of `ep2-3De v1.0` is the on-chip memory. We demonstrated that an implementation of the material point framework quickly reaches the hardware limit of GPUs, even on modern architectures. It is then essential to overcome this limit in order to resolve larger computational domain with a greater amount of material points.

Here, we address this concern by implementing a distributed memory parallelisation using the message passing interface (MPI) standard. However, we limit our implementation efforts by considering 1) a one-dimensional GPU topology, 2) no computation/communication overlaps, and 3) only mesh-related quantities are shared amongst GPUs, i.e., the material points are not transferred between GPUs during a simulation. We also selected a non-adaptative time step to avoid the collection of the material point's velocities located in different GPUs at the beginning of each calculation cycle.

Available computational resources

The multi-GPU simulations are performed on the supercomputer Octopus running on a CentOS with the latest CUDA version v11. The multi-GPU simulations are run on the two different systems. The first one is an Nvidia DGX-1 - like node hosting 8 Nvidia Tesla V100 Nvlink (32 GB) GPUs, 2 Intel Xeon Silver 4112 (2.6 GHz) CPUs. The second one is composed of 32 nodes, each featuring 4 Nvidia GeForce GTX Titan X Maxwell (12 GB) GPUs, 2 Intel XEON E5-2620V3 4112 (2.4 GHz) CPUs. To summarize the computational resources in use, Table 1 presents the main characteristics of the GPUs used in this study.

Table 1: List of the graphical processing units (GPUs) used for multi-GPU simulations.

GPU	Architecture	On-chip memory [GB]
8×V100	Volta	8×32
128×GTX Titan X	Maxwell	128×12

Model 2a

To avoid frequent material point's transfers amongst the GPUs, we consider an overlap of 8 elements between neighbouring meshes, i.e., 9 nodes. This results in a one-dimensional GPU topology, for which both material points and meshes are distributed along the $y-$direction of the global computational domain (see Figs. 6 & 7). Arranging GPUs along this direction allows to overcome the need to transfer material points amongst GPUs, provided that the material point's displacement is not greater than the buffer zone, i.e., the element overlap. The evaluation of the multi-GPU implementation is based on the Model 2a, with slight modifications, i.e., the number of element along the y-direction is largely increased. The size of the physical domain $l_z \times l_x \times l_y$ is, at most, 12 m × 64 m × (64×2048) m.

Model 2a: multi-GPU performances

We consider two distributed computing systems for parallel GPU computation, using up to 8 Tesla V100 (Volta architecture) or 128 Geforce GTX Titan X (Maxwell architecture). All numerical simulations are performed using a single-arithmetic precision (i.e., $n_\mathrm{p} = 4$ bytes). This allows to increase the maximum number of material points and mesh dimensions. In addition, our GPU implementation relies on the usage of the built-in function `atomicAdd()`. It does not support the double-precision floating-point format FP64 for GPUs with compute capabilities lower than 6.0, i.e., the Maxwell architecture amongst others.

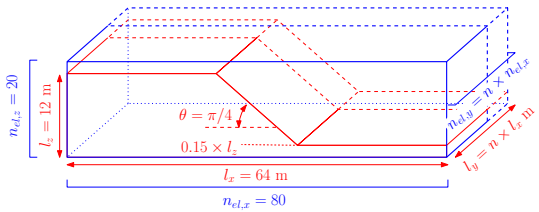


Figure 6: Geometry for the earth slump. For the multi-GPU implementation, the number of element along the $y-$direction can be largely increased, i.e., $n = 2048$.

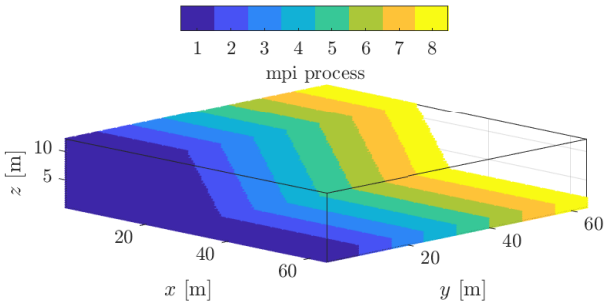


Figure 7: Domain partition of the material points amongst 8 GPUs. Combined with an overlap of 8 elements along the y-direction, material points can moderately move while still residing within the same GPU during the whole simulation.

Note that, unlike the Tesla V100, the Geforce GTX Titan X only delivers an effective memory throughput of $\text{MTP}_{\text{eff}} \approx 100$ GBs^{-1}. This corresponds to 38 % of its hardware limit. This was already reported by Räss et al., 2019; Alkhimenkov et al., 2021 and, it could be attributed to its older Maxwell architecture (Gao et al., 2018). This performance drop is even more severe, mainly due to the use of built-in functions like `atomicAdd()`.

Computing system: up to 8 Tesla V100

We first performed parallel simulations with a moderate number of GPUs, i.e., up to 8 Tesla V100 NVlink (32 GB). The respective wall-clock times are reported in Fig. 8. We report a wall-clock time of ≈ 110 s for $n_{mp} \approx 10^8$. For the same amount of material points, we report a roughly weak scaling between the number of GPUs and the wall-clock time. If n_{mp} is increased by a factor 2, 4 or 8, the wall-clock time is roughly similar to the baseline, i.e., $n_{\text{GPU}} = 1$.

Such weak scaling is more obvious when inspecting the MTP_{eff} measured (see Fig. 9), i.e., the total sum of MTP_{eff} across all the GPUs. Based on the memory throughput of 1 GPU, an estimation of a perfect weak scaling is possible. For 8 GPUs, it should correspond to $\text{MTP}_{\text{eff}} = 4824$ GBs^{-1}, whereas we report $\text{MTP}_{\text{eff}} = 4538$ GBs^{-1}. This gives a parallel efficiency of $\approx 94\%$ and, an effective speed-up of $7.5\times$. Similar observations are made for $n_{\text{GPU}} = 2$ and $n_{\text{GPU}} = 4$.

Computing system: up to 128 Geforce GTX Titan X

We investigate a parallel GPU computing using up to 128 Geforce GTX Titan X. This allows to address even larger geometries, as showed in Fig 10 where a geometry of nearly $n_{mp} \approx 8 \cdot 10^8$ is resolved in less than

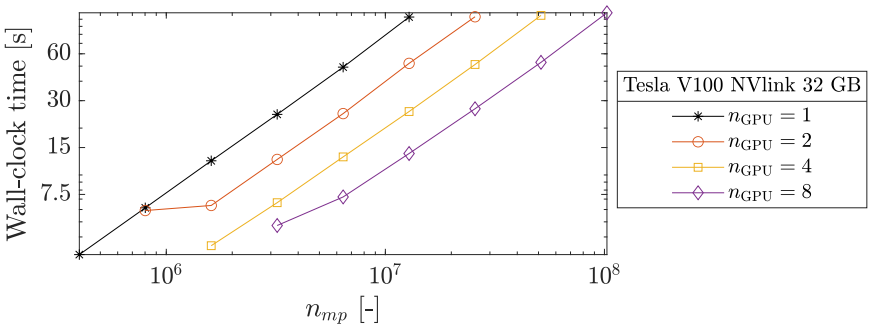


Figure 8: Wall-clock time for 1, 2, 4 and 8 Tesla V100 GPUs.

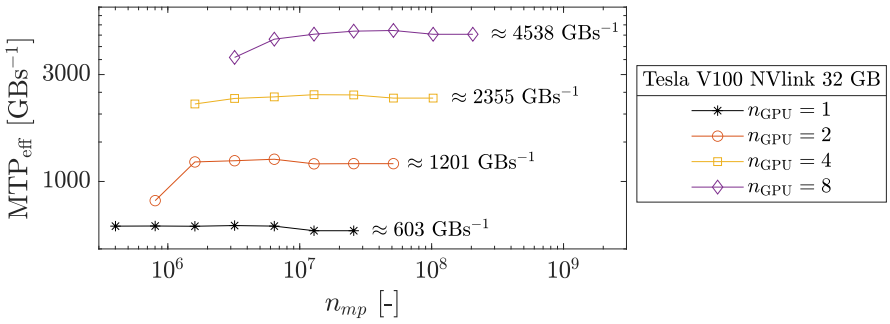


Figure 9: Sum across the GPUs involved of the MTP_{eff}. We roughly report a weak scaling between the number of GPUs and the overall effective memory throughput.

8 minutes. The first observation is that, for parallel computing up to 64 GPUs, the wall-clock time evolution is smooth. For 128 GPUs, the wall-clock time is chaotic for fewer material points whereas it stabilizes as the number of material points increases. We suspect the absence of computation/communication overlaps to be the main reason of this erratic behaviour. The communication between many GPUs requires careful synchronization between GPUs which can be hidden under computation/communication overlap. The total size of the overlap is constant, regardless of the $y-$dimension. As the number of material points increases, the time spent on computation becomes larger compared to the time spent on exchanges between GPUs and the wall-clock time stabilizes.

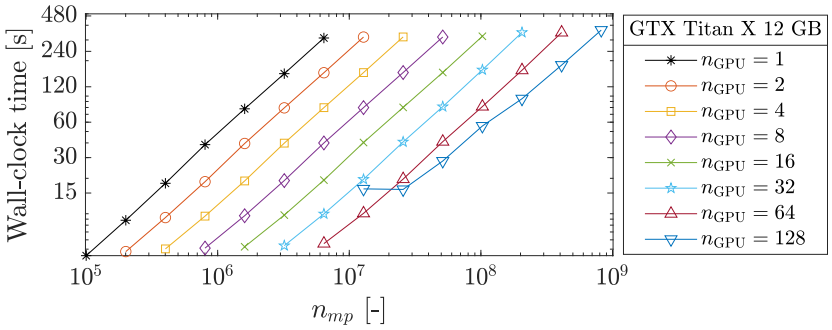


Figure 10: Wall-clock time reported for up to 128 Geforce GTX Titan X GPUs and up to $n_{mp} \approx 8 \cdot 10^8$.

Another observation is the effective memory throughput (see Fig. 11). When considering a perfect weak scaling, one should measure an effective memory throughput $\text{MTP}_{\text{eff}} = 12800$ GBs^{-1} for 128 GPUs whereas we report only $\text{MTP}_{\text{eff}} = 10953$ GBs^{-1}. This gives a parallel efficiency of $\approx 85\%$ and, an effective speed-up of $\approx 110\times$. When using less GPUs, the parallel efficiency is higher, i.e., 98 % for 8 GPUs.

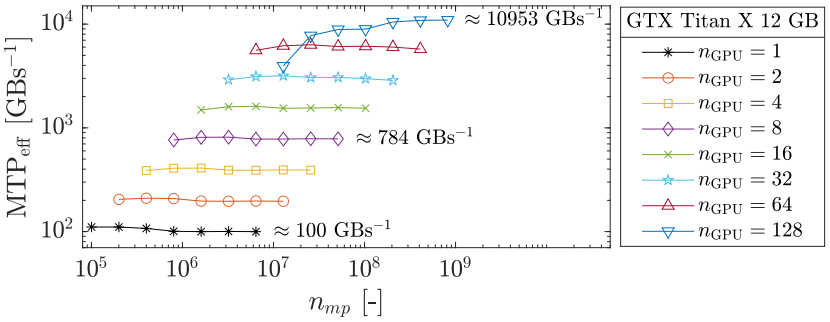


Figure 11: MTP_{eff} sum across the GPUs involved.

Discussion

Even tough the simplifications made alleviate the on-chip memory limitation, the type of problem, which can be addressed, is reduced. As an example, investigating high-resolution three-dimensional granular collapses is not possible under the assumptions made, because of small displacement required along the $y-$direction. This is incompatible with three-dimensional granular collapses. Hence, this motivates future deeper investigations toward a more versatile multi-GPU implementation. In addition, we report a slight drop of the parallel efficiency, as the number of GPUs increases. Future works should be directed toward a parallel strategy that hides communication latency, as proposed in Räss et al., 2019; Räss et al., 2020; Alkhimenkov et al., 2021.

However, such multi-GPU implementation is particularly well-suited to resolve highly-detailed three-dimensional shear-banding. We also reported decent wall-clock times (less than 8 minutes) for simulations with nearly a billion material points. One could argue that limiting the material point method to small displacement is a non-sense. Essentially, finite element codes are better suited for small strain analysis. However, this gives interesting insights on a multi-GPU implementation of the material point framework on a GPU supercomputer.

References

Alkhimenkov, Y., L. Räss, L. Khakimova, B. Quintal, and Y. Podladchikov (2021). "Resolving wave propagation in anisotropic poroelastic media using graphical processing units (GPUs)". In: *Journal of Geophysical Research: Solid Earth* n/a.n/a. e2020JB021175 2020JB021175, e2020JB021175. DOI: https://doi.org/10.1029/2020JB021175.

Bisht, V., R. Salgado, and M. Prezzi (2021). "Simulating penetration problems in incompressible materials using the material point method". In: *Computers and Geotechnics* 133, p. 103593. DOI: https://doi.org/10.1016/j.compgeo.2020.103593.

Cuomo, S., P. Ghasemi, M. Martinelli, and M. Calvello (2019). "Simulation of Liquefaction and Retrogressive Slope Failure in Loose Coarse-Grained Material". In: *International Journal of Geomechanics* 19.10, p. 04019116. DOI: 10.1061/(ASCE)GM.1943-5622.0001500.

Gao, M., X. Wang, K. Wu, A. Pradhana, E. Sifakis, C. Yuksel, and C. Jiang (Dec. 2018). "GPU Optimization of Material Point Methods". In: *ACM Trans. Graph.* 37.6. DOI: 10.1145/3272127.3275044.

Lei, X., S. He, and L. Wu (2020). "Stabilized generalized interpolation material point method for coupled hydro-mechanical problems". In: *Computational Particle Mechanics*. DOI: 10.1007/s40571-020-00365-y.

Mast, C. M., P. Mackenzie-Helnwein, P. Arduino, G. R. Miller, and W. Shin (2012). "Mitigating kinematic locking in the material point method". In: *Journal of Computational Physics* 231.16, pp. 5351–5373. DOI: https://doi.org/10.1016/j.jcp.2012.04.032.

Räss, L, T. Duretz, and Y. Podladchikov (2019). "Resolving hydromechanical coupling in two and three dimensions: spontaneous channelling of porous fluids owing to decompaction weakening". In: *Geophysical Journal International* 218.3, pp. 1591–1616.

Räss, L., A. Licul, F. Herman, Y. Y. Podladchikov, and J. Suckale (2020). "Modelling thermomechanical ice deformation using an implicit pseudo-transient method (FastICE v1. 0) based on graphical processing units (GPUs)". In: *Geoscientific Model Development* 13.3, pp. 955–976.