Geosci. Model Dev. Discuss.,
https://doi.org/10.5194/gmd-2019-249-RC2, 2019

Geoscientific
Model Development
Discussions

Interactive comment on "Modelling thermomechanical ice deformation using a GPU-based implicit pseudo-transient method (FastICE v1.0)" *by* Ludovic Räss et al.

Anonymous Referee #2

Received and published: 24 October 2019

This paper presents a new GPU-based thermomechanical land-ice dynamical core, termed FastICE. This dynamical core relies on a numerical framework in which pseudo-transient iterations solve the implicit thermomechanical coupling equations between the ice velocities (governed by the full Stokes equations with nonlinear viscosity given by Glen's flow law) and ice temperature. The spatial discretization for the governing equations is the finite difference method on a staggered Cartesian grid. The algorithm requires no preconditioned linear solves and no global reductions. Strong as well as weak scalability, including a 93% weak scaling parallel efficiency, is demonstrated for the GPU-accelerated code.

There is currently a lot of ongoing R&D in the area of performance portability of land ice (and more broadly climate) models to GPUs and other advanced architectures, but I have not seen any papers prior to this one that present and demonstrate a full end-to-end land-ice model that runs correctly and efficiently on GPUs. Hence, this paper has a lot of archival value, and I anticipate it will be very much of interest to glaciologists and computer scientists interested in performance portability of land ice models to GPUs. The proposed thermomechanical pseudotransient time-stepping formulation is nice in that it does not require a linear solve (the development of portable preconditioners/linear solvers for land-ice is an ongoing research area that is holding up certain land ice models from being fully ported). Additionally, the non-dimensionalization of the thermo-mechanically coupled full Stokes equations is a worthwhile contribution of the paper, even as something that stands alone from the GPU implementation (although the authors derive this primarily to allow them to study the effect of reduced precision arithmetic in their land ice computations). It has been argued by some researchers that running land-ice models non-dimensionally may reduce ill-conditioning and improve performance, as it often does in CFD. It is nice to have a non-dimensionalization documented in an archival publication such as this one.

Overall, this is a well-written and interesting paper that makes a good contribution to the field of land-ice modeling. I do have a few comments/questions/concerns that I would like the authors to address in a revised manuscript prior to publication. These are summarized below.

1. The authors are correct that there has been little work in performance portability of existing land-ice dycores. One reference that is worth mentioning in this area is the following recent work involving the portability of the Albany Land-Ice first Order Stokes model of (Tezaur et al. 2015) to GPUs and other next-generation architectures using the Kokkos library and programming model:

J. Watkins, I. Tezaur, I. Demeshko. "A study on the performance portability of the finite element assembly process within the Albany land ice solver", E. van Brummelen, A.

Corsini, S. Perotto, G. Rozza, eds. Numerical Methods for Flows: FEF 2017 Selected Contributions, Elsevier, 2019.

This paper does not present a full end-to-end workflow that is portable to GPUs, however; it focuses on the performance portability of only the finite element assembly time, not the linear solve. It is nonetheless worth adding this reference to the bibliography and literature overview.

2. The discretization utilized in FastICE is a finite difference one on a staggered Cartesian grid. In recent years, many production land-ice models have moved to finite element or finite volume discretizations, as these allow you to use unstructured regionally and/or adaptively refined meshes to reduce the total number of dofs in the computation and allow the concentration of computational power where it is needed, which is not possible with structured uniform Cartesian grids. Moreover, w/ structured uniform Cartesian meshes, one ends up with very crude representations of the ice extent and grounding line. I realize that your reason for choosing finite differences was to utilize stencil-based techniques for approximating spatial derivatives in a way that is amenable to the GPU hardware. Is there any hope of extending the scheme to unstructured grids, perhaps using something like DG?

3. When starting your code, did you consider libraries such as Kokkos and RAJA for performance portability over straight-up CUDA? These libraries select the optimal data layout for the hardware used at compile time, thereby making a code portable to multiple architectures, including NVIDIA GPUs. Your current implementation relies on CUDA, which may be problematic if one wishes to run the code on GPUs not from NVIDIA (e.g. AMD GPUs). This may be important in the near future, as there are some planned open science machines coming out soon that are expected not to have NVIDIA GPUs.

4. Pseudo-transient Jacobian-free methods similar in flavor to those proposed here have shown promise for solving the Navier-Stokes equations on GPUs. These methods

work very well until the problem gets too stiff. In this stiff regime, one typically needs to cut the time step substantially, and a preconditioner/matrix is needed, which can be expensive on GPUs. Realistic land ice problems are in general very stiff, and one has a hard time developing good preconditioners even if one has the Jacobian matrix. The numerical examples described in the test case are very simple verification problems. I worry about how the method will perform on realistic problems. It would be good to see one such example in the paper to alleviate this concern. Of particular interest would be a test case with floating ice (e.g. Antarctica simulation), which can pose a lot of challenges for the solver (see R. Tuminaro, M. Perego, I. Tezaur, A. Salinger, S. Price. "A matrix dependent/algebraic multigrid approach for extruded meshes with applications to ice sheet modeling", SIAM J. Sci. Comput. 38(5) (2016) C504-C532). Something simpler to try before doing Antarctica would be a test case with floating ice, e.g. confined shelf, circular shelf.

5. Is CUDA unified virtual memory (UVM) utilized in the implementation, or the memory is managed manually? I assume the latter, but it would be good to state this in the paper. A lot of implementation rely on CUDA UVM, and I think one should move away from that to get the best performance – your paper may make a case for that.

6. The authors introduce the non-dimensionalization of the governing equations as something that is needed for studying the effect of single vs. double precision on the computations (which makes a lot of sense). The study of single vs. double precision arithmetic seems not that rigorous to me, however. Most of the cases were run with double precision, with a couple run single precision, and the authors don't really seem to draw any meaningful conclusions from these results. The effect of reduced/mixed precision arithmetic in continental scale land ice (and more broadly climate) applications is a very interesting research area, which can be formulated as a sensitivity problem and could merit its own publication. I suggest the authors either streamline the single vs. double precision arithmetic discussion, or cut it from this paper, saving it for a later follow on publication where it can be given the proper attention.

7. I am confused about the different resolutions of grids b/w the Elmer/ICE and FastICE computations (e.g. experiments 1 and 2). The codes are quite different as are the techniques therein (e.g. different disrcretizations – PSPG stabilized FEM for Elmer/ICE vs. staggered finite difference for FastICE) so it's hard to say which mesh resolution in Elmer/ICE will be "comparable" to one in FastICE. You must have had some reason for selecting the relative resolutions you considered – can you please explain this here and in the paper? It is difficult to convince the reader that the verification is rigorous w/o explaining discrepancies such as this one.

8. Along the lines of the previous comment, I do not like the discrepancies b/w Elmer/ICE and FastICE for experiment 2. Your theory about the pinning seems plausible, but you should really get to the bottom of this prior to publishing this manuscript.

9. Note that Elmer/ICE uses PSPG stabilization for the full Stokes equations rather than using inf-sup stable velocity-pressure finite elements. This may be worth keeping in mind when making comparisons to Elmer/ICE results.

10. I would be interested to see still more rigorous verification of FastICE, for example, convergence analyses with grid refinement. One can do this on a method of manufactured solutions problem (see

W. Leng, L. Ju, M. Gunzburger, S. Price. "Manufactured solutions and the verification of three-dimensional Stokes ice-sheet models", The Cryosphere 7 19-29, 2013.

for some MMS tests for the full Stokes equations) or by performing a convergence study w.r.t. a reference solution on a fine mesh on a canonical test case: ISMIP-HOM, Dome, Circular Shelf, Confined Shelf, etc. This is important for creating a culture of verification within the climate modeling community, and also to provide evidence that your results are trusted.

11. In my opinion, including the MATLAB and Elmer/ICE results in the computational performance section of the paper is somewhat misleading/confusing, given that the

runs are only on a single core CPU and not representative of CPU hardware capabilities. I am not sure one can make a conclusion from the results that the CPU algorithms are "bad" and the GPU ones are "good". To do a fair comparison you would have to, for instance, take 1 node of a machine with CPUs, max it out, and run Elmer/ICE, then repeat the same procedure for 1 node + GPUs, and look at the relative CPU times. Are you able to perform a study like this? I strongly suggest that you do this and modify the results to have a fair comparison and to avoid misleading the reader.

12. Ultimately, when you get to "real" ice sheet calculations, you will need a thickness solver, to determine how your geometry will change in time. This would need to be coupled with your temperature and velocity equations. Is adding the thickness solver the next step? Please sketch out how that will fit in with your algorithm and maintain performance on GPUs.

13. On p. 29: you state that you "established that a relatively high spatial numerical resolution is necessary to resolve the non-linear and spontaneous localisation of thermomechanically coupled ice flow, including more than 100 grid-points in the vertical direction". Can you please expand on this? It doesn't seem like you really studied the effect of vertical resolution in the problems presented, and this study would be more meaningful on more realistic land ice geometries than those considered. 100 grid points in the vertical dimension would be a lot more than is currently used in practice (most land ice models use on the order of 10 finite elements in the vertical dimension regardless of the horizontal spatial resolution although there is some evidence that more layers may be needed for finer resolution problems in (Tezaur et al. 2015)).

Please address also the following minor comments/typos:

- On p. 1, line 19: you imply that the models in parentheses (Bueler and Brown, 2009; Bassis, 2010; ….) are all shallow ice models, which is not true. For instance, the (Perego et al 2012) and (Tezaur et al. 2015) references are based on the first order Stokes equations, which are derived using a hydrostatic approximation together with the assumption that the ice sheet is thin. The (Bueler and Brown, 2009) reference focuses on the shallow shelf approximation, not the shallow ice approximation. A simple fix would be to change “such as shallow ice models” to “such as first-order Stokes (refs), shallow shelf (ref) and shallow ice (ref) models”.

- P. 2, line 43: since you define CPU, you should also define GPU.
- Title of Section 3 should be “Leveraging”.
- Title of Section 5.4: should be “Experiment 4” instead of “Experiment 3”.
- P. 29, like 554: “lever” should be “leverage”.

Interactive comment on Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2019-249, 2019.