Reply to Anonymous Referee #1

Referee's comment 1

The manuscript is well-written, and the governing equations are clearly presented. Overall the manuscript was enjoyable to read, and I learned a lot. The tests were also convincing – as convincing as visual comparisons of results can be. In general, the structure of the manuscript is traditional: Introduce new model, explain the basic principles and the implementation, and test the model against other models. This is fine, and it provides a convenient reference for later work. However, as a reader I would have liked to see a demonstration of what the new model can really do – just a sneak peek into the suite of problems that the authors hope to address with this new model. There are so many ice models being presented, but it is unfortunately surprisingly rare that we see ice-sheet models applied in ways that make us wiser. So, if possible, I encourage the authors to include a demonstration of the model toward the end of the manuscript – something that is visually, and intellectually, more appealing than the benchmark tests.

Author's reply 1

Thank you for your encouraging feedback. We agree that there is an increasing number of ice models and that it is not always clear what the specific contribution of these models to ice dynamics is. The motivation behind developing this model is to develop a process-based model that affords the necessary 3D resolution to capture englacial strain localisation. This process may be of critical importance in the boundaries of fast flow like the basal interface, grounding zones and shear margins. It is also a subtle component of the overall ice dynamics and requires a careful assessment of when and why it becomes relevant and which locations on our ice sheets might serve as test sites for the model predictions. We are currently working on two follow-up manuscripts applying this code to the flow-to-sliding transition and to shear margin stability. As you mention, developing sophisticated models and advancing our understanding of ice dynamics are two distinct challenges. The first is a necessary but not a sufficient condition for the latter. To do justice to both, we prefer to focus on the numerical methods, benchmarking and performance evaluation for this manuscript and leverage this code for advancing our understanding of ice dynamics in a separate manuscript that we will submit to a glaciological journal. While we agree that an actual application case is more appealing and interesting than benchmarks, we believe that this code can help us make progress on important, fundamental questions in glaciology and we prefer to develop this potential fully in our separate contributions. We are happy to make preliminary results available to you to demonstrate the value of the code for these problem. For this manuscript, we have included a more detailed motivation for this kind of code and more extensive reference to the problems for which it is relevant.

Referee's comment 2

Line 36: The GPU-acceleration is very interesting and, as far as I know, rather new in ice-sheet models. However, a quick search leads to Brædstrup et al. (2014) "Ice- sheet modelling accelerated by graphic cards" in Computers & Geosciences 72, 210- 220. This paper is not cited here, although it covers some of the same challenges and principles of GPU-acceleration.

Author's reply 2

Thank you for pointing this out. We indeed overlooked the citation of the work from Brædstrup et al. (2014).

Changes in the manuscript 2

We added reference to this work in the revised manuscript at line 61: "We tailor our numerical method to optimally exploit the massive parallelism of GPU hardware, taking inspiration from recent successful GPU-based implementations of viscous and coupled flow problems (Brædstrup et al., 2014; Omlin, 2017; Räss et al., 2018; Duretz et al., 2019; Räss et al., 2019a)."

Referee's comment 3

line 42: Also, regarding GPU-acceleration, it would be good to see reference to other flow problems that have successfully been GPU accelerated. What problems and models have inspired the authors?

Author's reply 3

We rephrased in a more explicit way the source of inspiration of the GPU-based FastICE implementation (line 61).

Changes in the manuscript 3

Line 61: "We tailor our numerical method to optimally exploit the massive parallelism of GPU hardware, taking inspiration from recent successful GPU-based implementations of viscous and coupled flow problems (Brædstrup et al., 2014; Omlin, 2017; Räss et al., 2018; Duretz et al., 2019; Räss et al., 2019a)."

Referee's comment 4

line 122: The comment on single-precision calculations leaves me confused. Are the GPU-calculations single precision? Or does it depend on the specific GPU architecture? Please clarify.

Author's reply 4

The benchmarks and calculations in this study are performed using double precision arithmetic if not specified otherwise. We reported single precision efficiency to show the potential performance gain from reducing the arithmetic precision of the calculations. Until recently, it was commonly admitted and implicitly assumed that scientific calculations are (and should be) performed using double precision floating point arithmetic. This choice goes back a couple of decades ago when hardware was computation-bounded; double precision would provide enhanced convergence, thus more efficient calculations, since less floating operations were needed. However, we nowadays observe a shift towards memory-bounded hardware and software where transferring memory (numbers) is more limiting compared to performing arithmetic operations. Thus single or half precision calculation may become interesting as the numbers take twice or four time less amount of memory - which results in factor 2 or 4 performance increase. Alternatively, similar performance can be observed for a two or four-times increase in the numerical grid resolution. Future work may address whether performing calculations using lower arithmetic precision but increased numerical grid resolutions can outperform well-established double precision calculations. A detailed assessment of the issue may deserve separate publication.

Changes in the manuscript 4

Line 257: "The computations in CUDA C shown in the remainder of the paper were performed using double-precision arithmetic, if not specified otherwise."

Referee's comment 5

line 163: Braedstrup et al has a nice description of staggered grids and GPU acceleration – must be cited here.

Author's reply 5

Although we do not question the accurate description of the staggered grid from Braestrup et al., they use a Gauss-Seidel solver in their study, which shows some limitations in terms of parallel implementation. The solve they use requires information from neighbouring cells at each iteration which may, when executed in parallel, lead to read/write conflicts. Our PT solver relies on a fully parallel iteration strategy, which inherently takes care of updating the entire field of old values with updated ones thus circumventing the neighbouring cell read/write issues and avoiding to rely on a "red-black" type of scheme. We are now citing the suggested work, just not with specific reference to the staggered grid setup.

Changes in the manuscript 5

—

Referee's comment 6

line 175: Even up-wind advection schemes are going to suffer from numerical diffusion – and high numerical resolution is just making it worse. Please discuss this here.

Author's reply 6

True, upwind scheme also suffer from numerical diffusion. To ensure that our numerical results are not confounded by numerical diffusion, we set the numerical resolution such that the Grid Peclet number is smaller than the physical Peclet number, i.e. $\mathrm{n}_x > \mathrm{L}_x \mathrm{V}_x / 2$. Limiting numerical diffusion is one motivation for using high numerical resolution in our computations.

Changes in the manuscript 6

We have added the following clarification to the paragraph on line 183: "To ensure that our numerical results are not confounded by numerical diffusion, the Grid Peclet number must be smaller than the physical Peclet number. Limiting numerical diffusion is one motivation for using high numerical resolution in our computations."

Referee's comment 7

line 182: The matrix-free solver using pseudo-time is nicely explained. However, it would be good to see exactly how the residuals propagate in the grid. Many similar matrix-free relaxation schemes use multi-grid setups to make the residuals decay faster – these could be discussed.

Author's reply 7

An excellent point, thanks for bringing it up. We have included an additional figure in section 5.5 displaying the decay of the residual as function of the damping parameter.
Multi-Grid configuration are an alternative solution improving residual decay. However, MG methods may generate quite some overhead by the addition of multiple grid levels and may hinder performance by restriction and prolongation operators. Also, coarser grid may not saturate the GPU and result in a drop of efficiency.

Changes in the manuscript 7

Line 247: "The iteration count increases with the numerical problem size for second-order PT solvers scales close to ideal multi-grid implementations. However, the main advantage of the PT approach is its conciseness and the fact that only one additional read/write operation needs to be included - keeping additional memory transfers to the strict minimum."

Referee's comment 8

Eqn. 15: I believe that theta < 1 is often referred to as under-relaxation.

Author's reply 8

The variable θ is a scalar we use to select the fraction of a given nonlinear quantity to be updated each iteration. When θ=0, we would always use the initial guess, while θ=1, we would take 100% of the current nonlinear quantity. We usually define theta to be in the range of 1e-2 - 1e-1 in order to account for some time to fully relax the nonlinear quantities as the nonlinear problem may not be sufficiently converged at the beginning of the iterations. This approach is in a way similar to an under-relaxation scheme.

Changes in the manuscript 8

Line 204-209: "We use the scalar […] to select the fraction of a given nonlinear quantity, here the effective viscosity […], to be updated each iteration. When θ=0, we would always use the initial guess, while θ=1, we would take 100% of the current nonlinear quantity. We usually define theta to be in the range of […] in order to account for some time to fully relax the nonlinear viscosity as the nonlinear problem may not be sufficiently converged at the beginning of the iterations. This approach is in a way similar to an under-relaxation scheme and was successfully implemented in the ice sheet model development by Tezaur (2015), for example."

Referee's comment 9

Eqn. 19: Again, I miss information on how the residuals decay in the grid – particularly when using this stabilizing scheme. Also, I could not find previous reference to alpha, but I might have missed it.

Author's reply 9

Thank you for pointing out the missing α definition. We no longer use α in the manuscript, replacing it explicitly for enhanced clarity in Eqn. 19. We have also added a Figure 16 in the new Section 5.5 displaying a) the residuals' convergence history for a 2-D simulation and b) the impact of the "stabilising" scheme as function of the damping parameter ν in terms of the total number of iteration count to reach convergence threshold.

Changes in the manuscript 9

Line 453-458 and Figure 16.

Referee's comment 10

line 299: I can see how the non-dimensional equation makes the implementation simpler, but is it necessary to present results in the non-dimensional form? It just makes the output harder to understand.

Author's reply 10

As you point out, presenting results in a non-dimensional form has advantages and drawbacks. Dimensional results are more intuitive and easier to compare to observations, but non-dimensional results are more general and can be scaled back easily using the scales provided in Eqns 7 and 9 to various configurations without having to re-run the model. Here, we prefer the generality of non-dimensionality since we are looking at generic benchmark cases instead of applying our model to a particular field site or comparing against specific field measurements.

Changes in the manuscript 10

—

Referee's comment 11

Section 5: There is some repetition of captions in the text. "In Figure 4, we plot. . ."; "Figure 5 shows. . ."; "Figure 6 shows. . ." etc. This could be skipped to make the text smoother.

Author's reply 11

Thank you for pointing this out.

Changes in the manuscript 11

We re-phrased Section 5.1 and 5.2 avoiding the figure caption repetition for better clarity. Please refer to the revised text in Section 5 for updates.

Referee's comment 12

line 308: Why are the benchmark tests performed at different resolutions? Does the GPU-model require order-of-magnitude more DOFs to yield the same accuracy as the FEM model? The comparisons give leave me with that impression, and then what is the advantage of the PT setup?

Author's reply 12

Thank you for raising this important point. The benchmark tests where originally run at higher resolutions with the FastICE GPU code since we can afford it. The Elmer/Ice results are obtained on the largest available single-core/ direct solver resolution (or robust iterative solver for the 3D case). The latest results for the benchmark of experiment 2 show the good agreement among FastICE and Elmer/Ice at comparable resolutions. However, discrepancy between low and high numerical grid resolutions suggest that although the two different solution strategies match, they both may not fully capture the physics with accuracy at low resolutions in some cases, such as the 3D benchmark of Experiment 2. We report this issue in a new Figure 15 in the Section 5.5, showing the convergence of the numerical implementation among grid refinement.

Changes in the manuscript 12

Lines 441-458: We added a new Section "5.5: Validation of the FastICE numerical implementation" to discuss this topic and a related Figure 15.

Referee's comment 13

line 314: "numerical resolution grid resolution"

Author's reply 13

Thank you for pointing this out. We corrected the sentence. Which now reads:

Changes in the manuscript 13

Line 329: "…and used a numerical grid resolution…"

Referee's comment 14

line 333: The authors are right to address the discrepancies between the model results – but why not follow up on the idea to pin nodes in the FEM mesh?

Author's reply 14

We re-evaluated the benchmark test case using a comparable numerical grid resolution for our FastICE GPU solver and for Elmer/Ice. The result now agree for a particular numerical grid resolution. However, discrepancy with previous results suggest that the numerical resolution used to compare the two software may not be sufficient to resolve the physical process. To address this second limitation, we provide one additional figure showing the convergence of our method with and increase in numerical grid resolution and comparing the results to a high-resolution "reference" simulation.

Changes in the manuscript 14

We updated the Figure 7 with the latest benchmark test results at similar numerical grid resolutions between FastICE and Elmer/Ice and adapted the text from Section 5.2.

Lines 441-458: We added a new Section "5.5: Validation of the FastICE numerical implementation" to discuss this topic and a related new Figure 15.

Referee's comment 15

Fig. 15: The performance diagrams are very convincing – however, the use of widely different DOFs for the FEM and PT models in the benchmark tests makes we wonder if the speedup is real?

Author's reply 15

The purpose of these graphs is not to report speed-up versus single-core Matlab or Elmer/ice, but to inform the reader about the potential and the scaling of the iterative and matrix-free PT approach to handle large number of grid points representative of high-resolutions simulations. In terms of high-performance "desktop" computing - what certainly majority of the researcher still rely on - it is fair to compare the range of affordable DOF for the FEM and PT implementations. Finally, high resolution calculations affordable with the PT approach may become necessary when

resolving internal deformation localising into self-consistent formation of boundary layers prone to a sliding-like behaviour.

Changes in the manuscript 15

—

Sincerely yours,

Ludovic Räss, on behalf of the authors.

Reply to Anonymous Referee #2

Referee's comment 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.

Author's reply 1

Thank you for suggesting this reference on related topics. We have included it into our manuscript.

Changes in the manuscript 1

Line 63: "Our work contributes to the few land-ice dynamical cores targeting many-cores architectures such as GPUs (Brædstrup et al., 2014; Watkins et al., 2019)"

Referee's comment 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 discretisations, 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?

Author's reply 2

Indeed, many large-scale ice models have moved to finite elements to conform to complex basal topography and other geometric complexities arising in the grounding zone or on ice shelfs. The motivation behind FastICE is develop a complementary tool to existing approaches that enables us to better model and understand englacial instabilities such as thermo-mechanical localisation at the scale of individual field sites. Thermo-mechanical localisation arise in a self-consistent way in shear margins, at the grounding zone or in the vicinity of the basal sliding interface, but the degree and location of localisation is not known apriori. A body-fitted mesh is hence less valuable for our purposes than for problems with fixed geometry. Grid adaptivity could be beneficial and we have used it in previous problems that were dominated by singularities (e.g., Suckale et al., 2014). Recent work, however, suggests that singularities are blunted dynamically and that the flow field exhibits significant 3D variability throughout the entire boundary layer. The goal of FastICE is to better understand the physical processes governing this small-scale variability by quantifying the observational signature of different processes and comparing these model predictions against observational data at the field-site, rather than the regional, scale. You are of course correct in pointing out that Cartesian uniform meshes combined with the Finite-difference method enable the numerical application to run in parallel on GPUs close to hardware limit, but amenability of our grid setup to the GPU hardware is only one reason for opting for a Cartesian grid. The more important difference is that FastICE is targeting other scientific problems than many existing land-ice models. We added it t the discussion.

Changes in the manuscript 2

Line 539-543: "To address these limitations, we have developed FastICE, a new parallel GPU-based numerical model. The goal of FastICEis to better understand the physical processes that govern englacial instabilities such as thermomechanical localisation at the field-site, rather than the regional, scale. It hence targets other scientific problems than many existing land-ice models and complements these previous models."

Referee's comment 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.

Author's reply 3

Code portability is an important point, thank you for raising it. FastICE development aligns within a general effort to spread high-performance, parallel and super computing to Earth sciences. Usually performance and portability are rather opposite as a general and portable implementation may trade off performance, and vice-versa. However, the vectorised CUDA indexes could be replaced by explicit loops that can be parallelised using a shared memory approach (such e.g. openMP). Regarding various GPU designs, there are active development efforts by the broader community of wrappers to enable porting CUDA-based code to AMD or Intel GPUs.

Changes in the manuscript 3

—

Referee's comment 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.

Author's reply 4

An important point, thank you for raising it. Stiffness is indeed a concern in ice-sheet modelling, but it is a challenge not only for numerical reasons. Rather, it is a reflection of changing physical processes that govern ice flow at different scales and also at different locations along outlet glaciers and ice streams. One approach to tackling that challenge is to focus on numerical techniques suited specifically for stiff problems. Another is to focus on understanding the physical processes that lead to stiff behaviour in the first place and adjust the governing equations in suitable ways to represent these. The philosophy behind FastICE is the latter approach. We argue that specific locations on ice sheets like shear margins, grounding zones and the basal sliding interface require a multi-physics approach that could be built into FastICE. You mention the example of ice shelfs, which is of course at the heart of the current debate about sea-level-rise projections. There are many challenges in better understanding the coupling between ice shelfs, the ocean, and land ice including the ice-cliff instability (which requires a brittle rheology and failure model), the vulnerability of ice shelfs to meltwater ponding at the surface (which requires an englacial hydrology model), and the dynamics of the grounding zone (which requires a free-boundary model). Needless to say, ultimately we need both, better numerical techniques for stiff problems and a better physical understanding. Since we focus primarily on the field-site rather than the regional or ice-sheet scale, some of the large-scale numerical issues like stiffness are less of a problem for the applications that we are interested in. We clarified the motivations behind FastICE and how our model complements existing approaches rather than attempting to replace them.

Changes in the manuscript 4

Line 245-256: "Many large-scale ice models have moved to finite elements to conform to complex basal topography and other geometric complexities arising in the grounding zone or on ice shelves. The motivation behind FastICE is develop a complementary tool to existing approaches that enables us to better model and understand englacial instabilities such as thermomechanical localisation at the scale of individual field sites. Thermomechanical localisation arises in a self-consistent way in shear margins, at the grounding zone or in the vicinity of the basal sliding interface, but the degree and location of localisation is not known apriori. A body-fitted mesh is hence less valuable for our purposes than for problems with fixed geometry. Grid adaptivity could be beneficial and we have used it in previous problems that were dominated by singularities […]. Recent work, however, suggests that singularities are blunted dynamically and that the flow field exhibits significant 3-D variability throughout the entire boundary layer. The goal of FastICE is to better understand the physical processes governing this small-scale variability by quantifying the observational signature of different processes and comparing these model predictions

against observational data at the field-site, rather than the regional, scale. FastICE is targeting other scientific problems than many existing land-ice models."

Referee's comment 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.

Author's reply 5

Thank you for pointing out the need to clarify memory management. Our implementation does indeed not rely on the UVM features from CUDA, because at the time we initiated the work and later on assessed the UVM performance (early 2018), UVM was showing about one order of magnitude lower performance. We suspect the internal memory handling to be responsible of constantly synchronising host and device memory, which is not needed in our case. We clarified this by adding a statement in the Section 3.1.

Changes in the manuscript 5

Line 273: "Our implementation does not rely on the CUDA unified virtual memory (UVM) features. UVM avoids to explicitly define data transfer between the host (CPU) and device (GPU) arrays but results in about one order of magnitude lower performance. We suspect the internal memory handling to be responsible of continuously synchronising host and device memory, which is not needed in our case."

Referee's comment 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.

Author's reply 6

The choice of arithmetic precision is an important topic and merits an in-depth assessment resulting its own publication (see also response 4 to review #1). Our current study does not aim at investigating the effects, benefits and drawbacks of various arithmetic precision implementations. Although not in the current spotlight, we still wish to highlight the ability of our model to perform using single precision floating point arithmetics. Together with the non-dimensional for of the governing equations, the features pave the path for future studies addressing these important issues related to lower precision arithmetic and their benefits in light of memory bounded applications.

Changes in the manuscript 6

Line 257: "The computations in CUDA C shown in the remainder of the paper were performed using double-precision arithmetic, if not specified otherwise."

Referee's comment 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.

Author's reply 7

You are correct pointing out it is hard to say what are the optimal mesh resolutions in order to compare various discretisation and numerical methods. For the benchmark, we decided to employ as large as possible numerical resolutions that would still deliver results in "reasonable" (day-scale) wall-times while running on desktop-type of computer hardware (single CPU - single GPU). For optimal comparison, we selected rectangular mesh elements within the Elmer/Ice FEM framework; we are confident about our choice to be a reasonable comparison involving similar regular spatial discretisation. The two solving approaches should deliver similar results independently of the numerical implementations. We addressed this in the result section.

Changes in the manuscript 7

Line 329-334: “We use higher numerical grid resolution within FastICE as we can afford it. Varying the numerical resolution also permits to test both the agreement between to different numerical approaches and convergence. The fact that we obtain matching results when increasing grid resolution significantly suggests that we resolve the relevant physical processes sufficiently, even at lower resolutions. We report an exception to this trend in the 3-D case of Experiment 2.”

Referee’s comment 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.

Author’s reply 8

We addresses the issue regarding the discrepancy between FastICE and Elmer/Ice in the 3D configuration of experiment 2. We repeated the benchmark using similar gird resolution in FastICE than Elmer/Ice and the results agree. We are thus confident FastICE reproduces the benchmark tests with similar accuracy than Elmer does. However, our original results suggests that the spatial resolution at which the benchmark is performed may not be sufficient in order to achieve convergence of the numerical results. We investigated this issue by performing an additional test refining the numerical grid resolution from coarse to a reference numerical solution on a fine grid. We show convergence of the method among grid refinement.

Changes in the manuscript 8

Lines 441-458: We added a new Section “5.5: Validation of the FastICE numerical implementation” to discuss this topic and a related new Figure 15.

Referee’s comment 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.

Author’s reply 9

Yes, thank you for pointing this out.

Changes in the manuscript 9

—

Referee’s comment 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.

Author’s reply 10

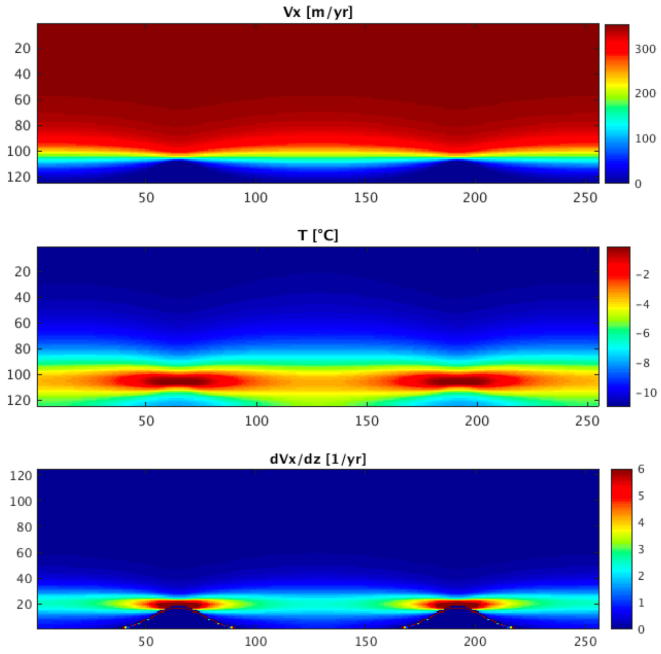
We agree and support the importance of a culture of verification within the climate modelling community (and beyond). We thus provided an additional figure reporting the convergence of our method for a given configuration among increase of the numerical grid resolution. We report that our method is first order accurate (expected from the finite-difference approximation) with regards to high-resolution reference results in both 2-D and 3-D.

Changes in the manuscript 10

Lines 441-458: We added a new Section “5.5: Validation of the FastICE numerical implementation” to discuss this topic and a related new Figure 16.

Referee’s comment 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.

Author's reply 11

We support your comment and agree one should not jump to conclusions about an algorithm being "bad" or "good" based on those single-core CPU results displayed besides GPU-based results. However, those are just facts and we want to show what value to expect in our metric for a single-core CPU process. Due to the infinite number of possible node configurations, I do not think that one could ever make a relevant comparison. This motivated our choice to report the following results. We compared non MPI Elmer/Ice runtime on a desktop machine versus a non MPI FastICE runtime on a single desktop GPU, with the drawback that CPU utilisation is not maximised by construction while GPU utilisation is. Finally, we are mostly interested to report the scaling of the fastICE runtime with increase in problem size rather than to perform and extensive comparison among FastICE and Elmer/Ice as performance cannot be fairly compared given the different approaches.

Changes in the manuscript 11

—

Referee's comment 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.

Author's reply 12

Indeed, including a thickness solver could be one way forward. That being said, our primary goal with FastICE is an improved process-based understanding of the boundaries of fast flow including shear margins, grounding zones and the basal sliding interface instead of focusing on "real" ice-sheet calculations for which several models already exist. Recent studies (e.g., Elsworth and Suckale, 2016) have shown that shear margin locations can shift almost discontinuously over as little as a few months if their location is governed by subglacial hydrology. These rapid adjustments of the sliding interface are an important contributor to the uncertainty in near-term sea-level-rise projections and are currently our primary focus. In most locations, with the possible exception of Thwaites Glacier, ice thickness will change very little on the monthly to annual time scale. With that scope in mind, a thickness solver is less important than integrating multi-physics behavior such as englacial and subglacial hydrology. There is no general answer on how these multi-physics components will alter GPU performance and we agree that a careful implementation is necessary to maintain scalability. That being said, the pseudo-transient algorithm behind FastICE lends itself to the integration of other components and can be tailored to the need of future specific studies.

Changes in the manuscript 12

—

Referee's comment 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)).

Author's reply 13

High vertical (and horizontal) resolution will be needed to resolve local stress and pressure gradient arising from interaction with non-flat topography or to dynamically capture the localisation of strain and heat in the formation of shear-zones such as internal sliding layers (see attached figure). Those results are in consideration for publication in a separate study.

Changes in the manuscript 13

—

Referee's comment 14

Please address also the following minor comments/typos:

- 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".

Author's reply 14

- Thank you for your suggestions. We rephrased that portion of the introduction following your guideline.
- GPU is defined 6 lines previous to the definition of CPU.
- To lever (verb), to lever + age (noun), So the verb is to lever and not to leverage (see this link https://this.isfluent.com/blog/2010/are-you-stupid-enough-to-use-leverage-as-a-verb for further details - apologies for the somewhat inappropriate language).
- Experiment 4 is a variation of Experiment 3. We thus renamed them Experiments 3a and 3b for enhanced readability.

Changes in the manuscript 14

See previous lines.

Sincerely yours,

Ludovic Räss, on behalf of the authors.

Modelling thermomechanical ice deformation using a GPU-based implicit pseudo-transient method (FastICE v1.0)

Ludovic Räss[1], Aleksandar Licul[2,3], Frédéric Herman[2,3], Yury Y. Podladchikov[3,4], and Jenny Suckale[1]

[1]Stanford University, Geophysics Department, 397 Panama Mall, Stanford CA 94305, USA.
[2]Institute of Earth Surface Dynamics, University of Lausanne, 1015 Lausanne, Switzerland.
[3]Swiss Geocomputing Centre, University of Lausanne, 1015 Lausanne, Switzerland.
[4]Institute of Earth Sciences, University of Lausanne, 1015 Lausanne, Switzerland.

Correspondence: Ludovic Räss (ludovic.rass@gmail.com)

Abstract. ~~Accurate predictions of future sea level rise require numerical models that capture the complex thermomechanical feedbacks in rapidly deforming ice. Shear~~ Ice sheets loose the majority of their mass through outlet glaciers or ice streams, corridors of fast ice moving multiple orders of magnitude more rapidly than the surrounding ice. The future stability of these corridors of fast moving ice depends sensitively on the behaviour of their boundaries, namely shear margins, grounding zones and the basal sliding interface~~are locations of particular interest~~ , where the stress-field is complex and fundamentally three-dimensional. These ~~transition zones~~ boundaries are prone to thermomechanical localisation, which can be captured numerically only with high temporal and spatial resolution. Thus, better understanding the coupled physical processes that govern ~~these boundaries of localised strain~~ the response of these boundaries to climate change necessitates a non-linear, full Stokes model that affords high resolution and scales well in three dimensions. This paper's goal is to contribute to the growing toolbox for modelling thermomechanical deformation in ice by levering GPU accelerators' parallel scalability. We propose FastICE, a numerical model that relies on pseudo-transient iterations to solve the implicit thermomechanical coupling between ice motion and temperature involving shear-heating and a temperature-dependant ice viscosity. ~~Our method~~ FastICE is based on the finite-difference discretisation, and we implement the pseudo-time integration in a matrix-free way. We benchmark the mechanical Stokes solver against the finite-element code Elmer/Ice and report good agreement among the results. We showcase a parallel version of ~~the solver~~ FastICE to run on GPU-accelerated distributed memory machines, reaching a parallel efficiency of ~~93~~99%. We show that our model is particularly useful for improving our process-based understanding of flow localisation in the complex transition zones bounding rapidly moving ice.

1 Introduction

The fourth IPCC report (Solomon et al., 2007) ~~revealed~~ concludes that existing ice sheet flow models do not accurately describe polar ice sheet discharge (e.g., Gagliardini et al., 2013; Pattyn et al., 2008) owing to their inability to simultaneously model slow and fast ice ~~flow~~ motion (Gagliardini et al., 2013; Bueler and Brown, 2009). This issue results from the fact that many ice flow models are based on simplified approximations of non-linear Stokes equations, such as ~~shallow ice models (Bueler and Brown, 2009; Bassis, 2010; Schoof and Hindmarsh, 2010; Goldberg, 2011; Egholm et al., 2011; Pollard and DeConto, 2012;~~

first-order Stokes (Perego et al., 2012; Tezaur et al., 2015), shallow shelf (Bueler and Brown, 2009) and shallow ice (Bassis, 2010; Schoof a
models. Shallow ice models are computationally more tractable and describe the motion of large homogeneous portions of ice as a function of the basal friction. However, this category of models fails to capture the coupled multi-scale processes that govern the behaviour of the boundaries of streaming ice, including shear margins, grounding zones and the basal interface. These boundaries dictate the stability of the current main drainage routes from Antarctica and Greenland, and predicting their future evolution is critical for understanding polar ice sheet discharge.

Full Stokes models (Gagliardini and Zwinger, 2008; Gagliardini et al., 2013; Jarosch, 2008; Jouvet et al., 2008; Larour et al., 2012; Leng et al., 2012, 2014; Brinkerhoff and Johnson, 2013; Isaac et al., 2015) provide a complete mechanical description of deformation by capturing the entire stress-rate and strain-rate tensor. In three dimensions (3-D), full Stokes calculations set a high demand on computational resources that requires a parallel and high-performance computing approach to achieve reasonable times to solution. An added challenge in full Stokes models is ~~ice's~~ the strongly non-linear thermomechanics ~~. Ice's~~ of ice. Ice viscosity significantly depends on both temperature and strain-rate (Robin, 1955; Hutter, 1983; Morland, 1984), which can lead to spontaneous localisation of shear (e.g., Duretz et al., 2019; Räss et al., 2019a). Particularly challenging is the scale separation associated with localisation, which leads to micro-scale physical interaction generating meso-scale features such as thermally-activated shear zones or preferential flow paths in macro-scale ice domains. Thus, both high spatial and temporal resolutions are important for numerical models to capture and resolve spontaneous localisation.

~~This paper's main contribution~~ The main contribution of this paper is to lever the unprecedented parallel performance of modern graphical processing units (GPUs) to accelerate the time-to-solution for thermomechanically coupled full Stokes models in 3-D utilising a pseudo-transient (PT) iterative scheme – FastICE (Räss et al., 2019b). ~~We argue that our numerical model is particularly useful for advancing our~~ FastICE is a process-based ~~understanding of the boundaries of streaming flow including~~ model that focuses specifically on improving our ability to better model and understand spontaneous englacial instabilities such as thermomechanical localisation at the scale of individual field sites. Thermomechanical localisation arise in a self-consistent way in shear margins, ~~grounding zones and the~~ at the grounding zone and in the vicinity of the basal sliding interface~~. We demonstrate our thermomechanical Stokes models' ability to resolve the spontaneous ice flow localisation in both 2-D and 3-D and on (multiple) GPUs~~, making our model particularly well suited for assessing the complex physical feedbacks in the boundaries of fast moving ice. FastICE is a complement to existing models by providing a multi-physics platform for studying the transition between fast and slow ice motion rather than addressing the large-scale evolution of the entire ice sheet.

Recent trends in the computing industry show a shift from single-core to many-core architectures as an effective way to increase computational performance. This trend is common to both central processing unit (CPU) and GPU hardware architectures (Cook, 2012). GPUs are compact, affordable and relatively programmable devices that offer high performance throughput (close to TB/s peak memory throughput) and a good price to performance ratio. GPUs offer an attractive alternative to conventional CPUs owing to their massively parallel architecture featuring thousands of cores. The programming model behind GPUs is based on a parallel principle called Single Instruction Multiple Data (SIMD). This principle entails that every single instruction is executed on different data. The same instructions block is executed by every thread. GPUs' massive parallelism and the related high performance is achieved by executing thousands of threads concurrently using multi-threading

in order to effectively hide latency. Numerical stencil-based techniques such as the finite-difference method allow one to take advantage of GPU hardware, since spatial derivatives are approximated by differences between two (or more) adjacent grid-points. This results in minimal, local and regular memory access patterns. The operations performed on each stencil are identical for each grid-point throughout the entire computational domain. Combined with a matrix-free discretisation of the equations and iterative PT updates, the finite-difference stencil evaluation is well suited for the SIMD programming philosophy of GPUs. Each operation on the GPU assigns one thread to compute the update of a given grid-point. Since on the GPU device, one core can simultaneously execute several threads, the operation set is executed on the entire computational domain almost concurrently.

We tailor our numerical method to optimally exploit the massive parallelism of GPU hardware~~(Omlin, 2017; Räss et al., 2018; Duretz et a~~ ~~. Our~~ , taking inspiration from recent successful GPU-based implementations of viscous and coupled flow problems (Omlin, 2017; Räss et a , Our work contributes to the few land-ice dynamical cores targeting many-cores architectures such as GPUs (Brædstrup et al., 2014; Watkin . Our numerical implementation relies on an iterative and matrix-free method to solve the mechanical and thermal problems using a finite-difference discretisation on a Cartesian staggered grid. We ensure optimal performance, minimising the memory footprint bottleneck while ensuring optimal data alignment in computer memory. Our accelerated PT algorithm (Frankel, 1950; Cundall et al., 1993; Poliakov et al., 1993; Kelley and Keyes, 1998; Kelley and Liao, 2013) utilises an analogy of transient physics to converge to the steady-state problem at every time step. One advantage of this approach is that the iterative stability criterion is physically motivated and intuitive to adjust and to generalise. Using transient physics for numerical purpose allows us to define local CFL-like criteria in each computational cell to be used to minimise residuals. This approach enables maximal convergence rate simultaneously in the entire domain and avoids costly global reduction operations from becoming a bottleneck in parallel computing.

We verify the numerical implementation of our mechanical Stokes solver against available benchmark studies including EISMINT (Huybrechts and Payne, 1996) and ISMIP (Pattyn et al., 2008). There is only one model inter-comparison that investigates the coupled thermomechanical dynamics, EISMINT 2 (Payne et al., 2000). Unfortunately, experiments in EISMINT 2 are usually performed using a coupled thermomechanical first-order shallow ice model (Payne and Baldwin, 2000; Saito et al., 2006; Hindmarsh, 2006; Bueler et al., 2007; Hindmarsh, 2009; Brinkerhoff and Johnson, 2015) making the comparison to our full Stokes implementation less immediate. Although thermomechanically coupled Stokes models exist (Zwinger et al., 2007; Leng et al., 2014; Schäfer et al., 2014; Gilbert et al., 2014; Zhang et al., 2015; Gong et al., 2018), very few studies have investigated key aspects of the implemented model, such as convergence among grid refinement and impacts of one-way vs. two-way couplings, with few exceptions (e.g. Duretz et al., 2019).

We start by providing an overview over the mathematical model, describing ice dynamics and its numerical implementation. We then discuss GPUs capabilities and explain our GPU implementation. We further report model comparison against a selection of benchmark studies, followed by sharing the results and performance measurements. Finally, we discuss pros and cons of the method, and highlight glaciological contexts in which our model could prove useful. The codes examples based on the PT method in both MATLAB and CUDA C programming language are available for download from Bitbucket at ~~and from~~ https://bitbucket.org/lraess/fastice/ and from http://wp.unil.ch/geocomputing/software/.

2 The model

2.1 The mathematical model

We capture the flow of an incompressible, non-linear, viscous fluid – including a temperature-dependent rheology. Since ice is approximately incompressible, the equation for conservation of mass reduces to:

$$\frac{\partial v_i}{\partial x_i} = 0 \,, \tag{1}$$

where v_i is the velocity component in the spatial direction x_i.

Neglecting inertial forces, ice's flow is driven by gravity and is resisted by internal deformation and basal stress:

$$\frac{\partial \tau_{ij}}{\partial x_j} - \frac{\partial P}{\partial x_i} + F_i = 0 \,, \tag{2}$$

where $F_i = \rho g \sin(\alpha)[1, 0, -\cot(\alpha)]$ is the external force. Ice density is denoted by ρ, g is the gravitational acceleration, and α is the characteristic bed slope. P is the isotropic pressure and τ_{ij} is the deviatoric stress tensor. The deviatoric stress tensor τ_{ij} is obtained by decomposing the Cauchy stress tensor σ_{ij} in terms of deviatoric stress τ_{ij} and isotropic pressure P.

In the absence of phase transitions, the temporal evolution of temperature in deforming, incompressible ice is governed by advection, diffusion and shear-heating:

$$\rho c \left(\frac{\partial T}{\partial t} + v_i \frac{\partial T}{\partial x_i} \right) = \frac{\partial}{\partial x_i} \left(k \frac{\partial T}{\partial x_i} \right) + \tau_{ij} \dot{\epsilon}_{ij} \,, \tag{3}$$

where T represents the temperature deviation from the initial temperature T_0, c is the specific heat capacity, k is the spatially-varying thermal conductivity and $\dot{\epsilon}_{ij}$ is the strain-rate tensor. The term $\tau_{ij}\dot{\epsilon}_{ij}$ represents the shear-heating, a source term that emerges from the mechanical model.

Shear-heating could locally raise the temperature in the ice to the pressure melting point. Once ice has reached melting point, any additional heating is converted to latent heat, which prevents further temperature increase. Thus, we impose a temperature cap at the pressure melting point, following Suckale et al. (2014), by describing the melt production using a heavy-side function ~~$\theta(T - T_\mathrm{m})$:~~ $\chi(T - T_\mathrm{m})$:

$$\rho c \left(\frac{\partial T}{\partial t} + v_i \frac{\partial T}{\partial x_i} \right) = \frac{\partial}{\partial x_i} \left(k \frac{\partial T}{\partial x_i} \right) + [1 - \chi(T - T_\mathrm{m})] \tau_{ij} \dot{\epsilon}_{ij} \,, \tag{4}$$

where T_m stands for the ice melting temperature. We balance the heat produced by shear-heating with a sink term in regions where the melting temperature is reached. The volume of produced meltwater can be calculated in a similar way as proposed by Suckale et al. (2014).

We approximate the rheology of ice through Glen's flow law (Glen, 1952; Nye, 1953):

$$\begin{aligned}\dot{\epsilon}_{ij} &= \frac{1}{2}\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i}\right) \\ &= a_0 {\tau_{\mathrm{II}}}^{n-1} \exp\left(-\frac{Q}{R(T+T_0)}\right)\tau_{ij} \ ,\end{aligned} \tag{5}$$

where a_0 is the pre-exponential factor, R is the universal gas constant, Q is the activation energy, n is the stress exponent, and τ_{II} is the second invariant of the stress tensor defined by $\tau_{\mathrm{II}} = \sqrt{1/2\tau_{ij}\tau_{ij}}$. Glen's flow law posits an exponent of $n = 3$.

At the ice top surface $\Gamma_{\mathrm{t}}(t)$, we impose the upper surface boundary condition $\sigma_{ij}n_j = -P_{\mathrm{atm}}n_j$, where n_j denotes the normal unit vector at the ice surface boundary, and P_{atm} the atmospheric pressure. Because atmospheric pressure is negligible relative to pressure within ice column, we can also use a standard stress-free simplification of the upper surface boundary condition $\sigma_{ij}n_j = 0$. On the bottom ice-bedrock interface, we can impose two different boundary conditions. For the parts of the ice-bedrock interface $\Gamma_0(t)$ where the ice is frozen to the ground, we impose a zero velocity $v_i = 0$ and thus no sliding boundary condition. On the parts of ice-bedrock interface $\Gamma_{\mathrm{s}}(t)$ where the ice is at the melting point, we impose a Rayleigh friction boundary condition – the so-called linear sliding law – given by:

$$\begin{aligned} v_i n_i &= 0 \ , \\ n_i \sigma_{ij} t_j &= -\beta^2 v_j t_j \ ,\end{aligned} \tag{6}$$

where the parameter β^2 denotes a given sliding coefficient, n_i denotes the normal unit vector at the ice-bedrock interface, and t_j denotes any unit vector tangential to the bottom surface. On the side or lateral boundaries, we impose either Dirichlet boundary conditions if the velocities are known, or periodic boundary conditions, mimicking an infinitely extended domain.

2.2 Non-dimensionalisation

For numerical purposes and for ease of generalisation, it is often preferable to use non-dimensional variables. This allows one to limit truncation errors (especially relevant for single-precision calculations) and to scale the results to various different initial configurations. Here, we use two different scale sets, depending on whether we solve the purely mechanical part of the model or the thermomechanically coupled system of equations.

In the case of an isothermal model, we use ice thickness, H, and gravitational driving stress to non-dimensionalise the governing equations:

$$\begin{aligned} \overline{L} &= H \ , \\ \overline{\tau} &= \rho g \overline{L} \sin(\alpha), \ , \\ \overline{v} &= 2^n A_0 \overline{L} \overline{\tau}^n \ ,\end{aligned} \tag{7}$$

where A_0 is the isothermal deformation rate factor and α is the mean bed slope. We can then rewrite the governing equations in their non-dimensional form as follows:

$$\begin{aligned}
&\frac{\partial v_i'}{\partial x_i'} = 0 \,, \\
&\frac{\partial \tau_{ij}'}{\partial x_j'} - \frac{\partial P'}{\partial x_i'} + F_i' = 0 \,, \\
&\dot{\epsilon}'_{ij} = \frac{1}{2}\left(\frac{\partial v_i'}{\partial x_j'} + \frac{\partial v_j'}{\partial x_i'}\right) = 2^{-n} {\tau'_{\mathrm{II}}}^{n-1} \tau_{ij}' \,,
\end{aligned} \tag{8}$$

where F_i' is now defined as $F_i' = [1, 0, -\cot(\alpha)]$. The model parameters are the mean bed slope α and domain size in each horizontal direction, i.e. L_x' and L_y'.

Reducing the thermomechanically coupled equations to a non-dimensional form requires not only length and stress, but also temperature and time. We choose the characteristic scales such that the coefficients in front of the diffusion and shear-heating terms in the temperature evolution Eq. (3) reduce to one:

$$\begin{aligned}
&\overline{T} = \frac{n R {T_0}^2}{Q} \,, \\
&\overline{\tau} = \rho c_p \overline{T} \,, \\
&\bar{t} = 2^{-n} a_0^{-1} \overline{\tau}^{-n} \exp\left(\frac{Q}{R T_0}\right) \,, \\
&\overline{L} = \sqrt{\frac{k}{\rho c_p} \bar{t}} \,.
\end{aligned} \tag{9}$$

These choices entail that the velocity scale in the thermomechanical model is $\overline{v} = \overline{L}/\bar{t}$. We obtain the non-dimensional (primed-variables) by using the characteristic scales given in Eq. (9), which leads to:

$$\begin{aligned}
&\frac{\partial v_i'}{\partial x_i'} = 0 \,, \\
&\frac{\partial \tau_{ij}'}{\partial x_j'} - \frac{\partial P'}{\partial x_i'} + F_i' = 0 \,, \\
&\frac{\partial T'}{\partial t'} + v_i' \frac{\partial T'}{\partial x_i'} = \frac{\partial^2 T'}{\partial {x_i'}^2} + \tau_{ij}' \dot{\epsilon}'_{ij} \,, \\
&\dot{\epsilon}'_{ij} = \frac{1}{2}\left(\frac{\partial v_i'}{\partial x_j'} + \frac{\partial v_j'}{\partial x_i'}\right) \\
&\qquad = 2^{-n} {\tau'_{\mathrm{II}}}^{n-1} \exp\left(\frac{n T'}{1 + \frac{T'}{T_0'}}\right) \tau_{ij}' \,,
\end{aligned} \tag{10}$$

where F_i' is now defined as $F_i' = \overline{F}\,[1, 0, -\cot(\alpha)]$ and $\overline{F} = \rho g \sin(\alpha)\overline{L}/\overline{\tau}$. The model parameters are the non-dimensional initial temperature T_0', the stress exponent n, the non-dimensional force $\overline{F}$, the mean bed slope α, non-dimensional domain height L_z', and the horizontal domain size L_x' and L_y' (Figure 3). We motivate the chosen characteristic scales by their usage in other studies of thermomechanical strain localisation (Duretz et al., 2019; Kiss et al., 2019). In the interest of a simple notation, we will omit the prime symbols on all non-dimensional variables in the remainder of the paper.

2.3 A simplified 1-D semi-analytical solution

We consider a specific 1-D mathematical case where all horizontal derivatives vanish ($\partial/\partial x = \partial/\partial y = 0$). The only remaining shear stress component τ_{xz} and pressure P are determined by analytical integration and are constant in time considering a fixed domain (Figure 3). We assume that stresses vanish at the surface and we set both horizontal and vertical basal velocity components to 0. We then integrate the 1-D mechanical equation in the vertical direction and substitute it into the temperature equation, which leads to:

$$\begin{aligned}
&\frac{\partial T(z,t)}{\partial t} = \frac{\partial^2 T(z,t)}{\partial z^2} + 2^{(1-n)} \left(\overline{F}\mathrm{L}_z\right)^{(n+1)} \\
&\left(1 - \frac{z}{\mathrm{L}_z}\right)^{(n+1)} \exp\left(\frac{nT(z,t)}{1 + \frac{T(z,t)}{T_0}}\right), \\
&v_x(z,t) = 2^{(1-n)} \left(\overline{F}\mathrm{L}_z\right)^n \int_0^z \left(1 - \frac{z}{\mathrm{L}_z}\right)^n \\
&\exp\left(\frac{nT(z,t)}{1 + \frac{T(z,t)}{T_0}}\right) dz\,.
\end{aligned} \tag{11}$$

Notably, the velocity and shear-heating terms (Eq. 11) are now a function only of temperature and, thus, of depth and time. To obtain a solution of the coupled system, one only needs to numerically solve for the temperature evolution profile, while the velocity can then be obtained diagnostically by a simple numerical integration.

2.4 The numerical implementation

We discretise the coupled thermomechanical Stokes equations (Eq. 10) using the finite-difference method on a staggered Cartesian grid. Among many numerical methods currently used to solve partial differential equations, the finite-difference method is commonly used and has been successfully applied in solving a similar equations' set relating to geophysical problems in geodynamics (Harlow and Welch, 1965; Ogawa et al., 1991; Gerya, 2009). The staggering of the grid provides second-order accuracy of the method (Virieux, 1986; Patankar, 1980; Gerya and Yuen, 2003; McKee et al., 2008), avoids oscillatory pressure modes (Shin and Strikwerda, 1997), and produces simple yet highly compact stencils. The different physical variables are located at different locations on the staggered grid. Pressure nodes and normal components of the strain-rate tensor are located at the cell centres. Velocity components are located at the cell mid-faces (Figure 1), while shear stress components are located at the cell vertices in 2-D (e.g., Harlow and Welch, 1965). The resulting algorithms are well suited for taking advantage of

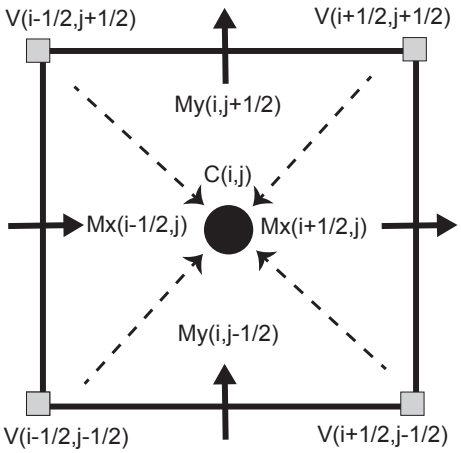


Figure 1. Setup of the staggered grid in 2-D. Variable C is located at the cell centre, V depicts variables located at cell vertices and Mx and My represents variables located at cell mid-faces in x or y direction.

modern many-core parallel accelerators, such as graphical processing units (GPUs) (Omlin, 2017; Räss et al., 2018; Duretz et al., 2019; Räss et al., 2019a). Efficient parallel solvers utilising modern hardware provide a viable solution to resolve the computationally challenging coupled thermomechanical full Stokes calculations in 3-D. The power law viscous ice rheology (Eq. 5) exhibits a non-linear dependence on both the temperature and the strain-rate:

$$\eta = \dot{\epsilon_{\mathrm{II}}}^{\frac{1-n}{n}} \exp\left(-\frac{T}{1+\frac{T}{T_0}}\right) , \tag{12}$$

where $\dot{\epsilon}_{\mathrm{II}}$ is the square root of the second invariant of the strain-rate tensor $\dot{\epsilon}_{\mathrm{II}} = \sqrt{1/2\dot{\epsilon}_{ij}\dot{\epsilon}_{ij}}$. We regularise the strain-rate and temperature dependant viscosity η to prevent non-physical values for negligible strain-rates, $\eta_{\mathrm{reg}} = 1/(\eta^{-1} + \eta_0^{-1})$. We use a harmonic mean to obtain a naturally smooth transition to background viscosity values at negligible strain-rate η_0.

We define temperature on the cell centres within our staggered grid. We discretise the temperature equation's advection term using a first-order upwind scheme, doing the physical time integration using either an implicit backward Euler or a Crank-Nicolson (Crank and Nicolson, 1947) scheme. To ensure that our numerical results are not confounded by numerical diffusion, the Grid Peclet number must be smaller than the physical Peclet number. Limiting numerical diffusion is one motivation for using high numerical resolution in our computations.

We rely on a pseudo-transient (PT) continuation or relaxation method to solve the system of coupled non-linear partial differential equations (10) in an iterative and matrix-free way (Frankel, 1950; Cundall et al., 1993; Poliakov et al., 1993; Kelley

and Keyes, 1998; Kelley and Liao, 2013). To this end, we reformulate the thermomechanical Eq. (10) in a residual form:

$$
\begin{aligned}
&-\frac{\partial v_i}{\partial x_i} = f_{\mathrm{p}} \,, \\
&\frac{\partial \tau_{ij}}{\partial x_j} - \frac{\partial P}{\partial x_i} + F_i = f_{\mathrm{v}_i} \,, \\
&-\frac{\partial T}{\partial t} - v_i \frac{\partial T}{\partial x_i} + \frac{\partial^2 T}{\partial {x_i}^2} + \tau_{ij}\dot{\epsilon}_{ij} = f_{\mathrm{T}} \,,
\end{aligned}
\tag{13}
$$

The right-hand-side terms $(f_{\mathrm{p}}, f_{\mathrm{v}_i}, f_{\mathrm{T}})$ are the non-linear continuity, momentum and temperature residuals, respectively, and quantify the magnitude of the imbalance of the corresponding equations.

We augment the steady-state equations with PT terms using the analogy of physical transient processes such as the bulk compressibility or the inertial terms within the momentum equations (Duretz et al., 2019). This formulation enables us to integrate the equation forward in pseudo-time τ until we reach the steady-state (i.e. the pseudo-time derivatives vanish). Relying on transient physics within the iterative process provides well-defined (maximal) iterative time step limiters. We reformulate Eq. (10):

$$
\begin{aligned}
&-\frac{\partial v_i}{\partial x_i} = \frac{\partial P}{\partial \tau_{\mathrm{p}}} \,, \\
&\frac{\partial \tau_{ij}}{\partial x_j} - \frac{\partial P}{\partial x_i} + F_i = \frac{\partial v_i}{\partial \tau_{\mathrm{v}_i}} \,, \\
&-\frac{\partial T}{\partial t} - v_i \frac{\partial T}{\partial x_i} + \frac{\partial^2 T}{\partial {x_i}^2} + \tau_{ij}\dot{\epsilon}_{ij} = \frac{\partial T}{\partial \tau_{\mathrm{T}}} \,,
\end{aligned}
\tag{14}
$$

where we introduced the pseudo-time derivatives $\partial/\partial\tau$ for the continuity $(\partial P/\partial\tau_{\mathrm{p}})$, the momentum $(\partial v_i/\partial\tau_{\mathrm{v}_i})$, and the temperature $(\partial T/\partial\tau_{\mathrm{T}})$ equation.

For every non-linear iteration k, we update the effective viscosity ${\eta_{\mathrm{eff}}}^{[k]}$ in the logarithmic space by taking a fraction θ_η of the actual physical viscosity $\eta^{[k]}$ using the current strain-rate and temperature solutions fields and a fraction $(1-\theta_\eta)$ of the effective viscosity calculated in the previous iteration ${\eta_{\mathrm{eff}}}^{[k-1]}$~~.~~:

$$
{\eta_{\mathrm{eff}}}^{[k]} = \exp\left[\theta_\eta \ln\left(\eta^{[k]}\right) + (1-\theta_\eta)\ln\left({\eta_{\mathrm{eff}}}^{[k-1]}\right)\right] \,. \tag{15}
$$

~~where~~ We use the scalar θ_η $(0 \leq \theta_\eta \leq 1)$ ~~is a viscosity relaxation factor. This relaxation of the non-linearity allows the effective viscosity to iteratively approach its physical value within the pseudo-transient iterations. A similar non-linear viscosity relaxation approach~~ to select the fraction of a given nonlinear quantity, here the effective viscosity η_{eff}, to be updated each iteration. When $\theta_\eta = 0$, we would always use the initial guess, while $\theta_\eta = 1$, we would take 100% of the current nonlinear quantity. We usually define theta to be in the range of $10^{-2} - 10^{-1}$ in order to account for some time to fully relax the nonlinear viscosity as the nonlinear problem may not be sufficiently converged at the beginning of the iterations. This approach is in a way similar to an under-relaxation scheme and was successfully implemented in the ice sheet model development by Tezaur et al. (2015)~~.~~, for example.

The pseudo-time integration of Eq. (14) leads to the definition of pseudo-time steps $\Delta\tau_\mathrm{p}, \Delta\tau_{\mathrm{v}_i}$ and $\Delta\tau_\mathrm{T}$, for the continuity, momentum and temperature equations, respectively. Transient physical processes such as compressibility (continuity equation) or acceleration (momentum equation) dictate the maximal allowed explicit pseudo-time step to be utilised in the transient process. Using the largest stable steps allows one to minimise the iteration count required to reach the steady-state:

$$\begin{aligned}
\Delta\tau_\mathrm{p} &= \frac{2.1 n_\mathrm{dim} \eta_\mathrm{eff}^k (1+\eta_\mathrm{b})}{\max(n_i)} \,, \\
\Delta\tau_{\mathrm{v}_i} &= \frac{\min(\Delta x_i)^2}{2.1 n_\mathrm{dim} \eta_\mathrm{eff}^k (1+\eta_\mathrm{b})} \,, \\
\Delta\tau_\mathrm{T} &= \left(\frac{2.1 n_\mathrm{dim}}{\min(\Delta x_i)^2} + \frac{1}{\Delta t} \right)^{-1} \,,
\end{aligned} \tag{16}$$

where n_dim is the number of dimensions, Δx_i and n_i are the grid spacing and the number of grid-points in the i direction ($i = x$ in 1-D, x, z in 2-D and x, y, z in 3-D), respectively. The physical time step, Δt, advances the temperature in time. The pseudo-time step $\Delta\tau_\mathrm{T}$ is an explicit Courant-Friedrich-Lewy (CFL) time step that combines temperature advection and diffusion. Similarly, $\Delta\tau_{\mathrm{v}_i}$ is the explicit CFL time step for viscous flow, representing the diffusion of strain-rates with viscosity as the diffusion coefficient. It is modified to account for the numerical equivalent of a bulk viscosity η_b. We choose $\Delta\tau_\mathrm{p}$ to be the inverse of $\Delta\tau_{\mathrm{v}_i}$ to ensure that the pressure update is proportional to the effective viscosity, while the velocity update is sensitive to the inverse of the viscosity. This interdependence reduces the iterative method's sensitivity to the variations in the ice's viscosity.

During the iterative procedure, we allow for finite compressibility in the ice, $\partial P / \partial \tau_\mathrm{p}$, while assuring that the PT iterations eventually reach the incompressible solution. The relaxation of the incompressibility constraint is analogous to the penalisation of pressure pioneered by Chorin (1967, 1968), and built on extensively subsequently. Compared to projection-type methods, it has the advantage that no pressure boundary condition is necessary that will lead to numerical boundary layers (Weinan and Liu, 1995). We use the parameter η_b to balance the divergence-free formulation of strain-rates in the normal stress component evaluation, where it is multiplied with the pressure residual f_p. Thus, normal stress is given by $\tau_{ii} = 2\eta(\dot{\epsilon}_{ii} + \eta_\mathrm{b} f_\mathrm{p})$. With convergence of the method, the pressure residual f_p vanishes and the incompressible form of the normal stresses is recovered.

Combining the residual notation introduced in Eq. (13), with the pseudo-time derivatives in Eq. (14) leading to the update rules:

$$\begin{aligned}
P^{[k]} &= P^{[k-1]} + \Delta P^{[k]} \,, \\
{v_i}^{[k]} &= {v_i}^{[k-1]} + \Delta {v_i}^{[k]} \,, \\
T^{[k]} &= T^{[k-1]} + \Delta T^{[k]} \,,
\end{aligned} \tag{17}$$

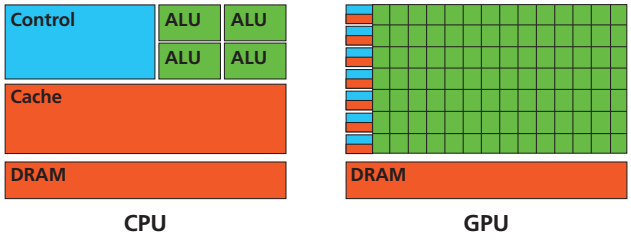


Figure 2. Schematic chip representation for both the central processing unit (CPU) and graphical processing unit (GPU) architecture. The GPU architecture consist of thousands of arithmetic and logical units (ALU). On the CPU, most of the on-chip space is devoted to controlling units and cache memory, while the number of ALUs is significantly reduced.

where the pressure, velocity and temperature iterative increments represent the current residual $[k]$ multiplied by the pseudo-time step:

$$\Delta P^{[k]} = \Delta\tau_{\mathrm{p}} f_{\mathrm{p}}{}^{[k]} ,$$
$$\Delta v_i{}^{[k]} = \Delta\tau_{\mathrm{v}_i} f_{\mathrm{v}_i}{}^{[k]} , \tag{18}$$
$$\Delta T^{[k]} = \Delta\tau_{\mathrm{T}} f_{\mathrm{T}}{}^{[k]} .$$

The straight-forward update rule (Eq. 17) is based on a first-order scheme $(\partial/\partial\tau)$. In 1-D, it implies that one needs N^2 iterations to converge to the stationary solution, where N stands for the total number of grid-points. This behaviour arises because the time step limiter $\Delta\tau_{v_\mathrm{i}}$ implies a second-order dependence on the spatial derivatives for the strain-rates. In contrast, a second-order scheme (Frankel, 1950), ~~$(\partial^2/\partial\tau^2 + \partial/\partial\tau)$~~ $(\psi\partial^2/\partial\tau^2 + \partial/\partial\tau)$ invokes a wave-like transient physical process for the iterations. The main advantage is the scaling of the limiter as Δx instead of Δx^2 in the explicit pseudo-transient time step definition. We can reformulate the velocity update as:

$$\Delta v_i{}^{[k]} = \Delta\tau_{v_i} f_{\mathrm{v}_i}{}^{[k]} + \left(1 - \frac{\nu}{n_i}\right) \Delta v_i{}^{[k-1]} \tag{19}$$

where ~~α~~ ψ can be expanded to $(1 - \nu/n_i)$ and acts like a damping term on the momentum residual. A similar damping approach is used for elastic rheology in the FLAC (Cundall et al., 1993) geotechnical software in order to significantly reduce the number of iterations needed for the algorithm to converge. The optimal value of the introduced parameter ν is found to be in a range $(1 \leq \nu \leq 10)$, and it is usually problem-dependent. This approach was successfully implemented in recent PT developments by Räss et al. (2018, 2019a) and Duretz et al. (2019). The iteration count increases with the numerical problem size for second-order PT solvers scales close to ideal multi-grid implementations. However, the main advantage of the PT approach is its conciseness and the fact that only one additional read/write operation needs to be included - keeping additional memory transfers to the strict minimum.

Notably, the PT solution procedure leads to a two-way numerical coupling between temperature and deformation (mechanics), which enables us to recover an implicit solution of the entire system of non-linear partial differential equations. Besides

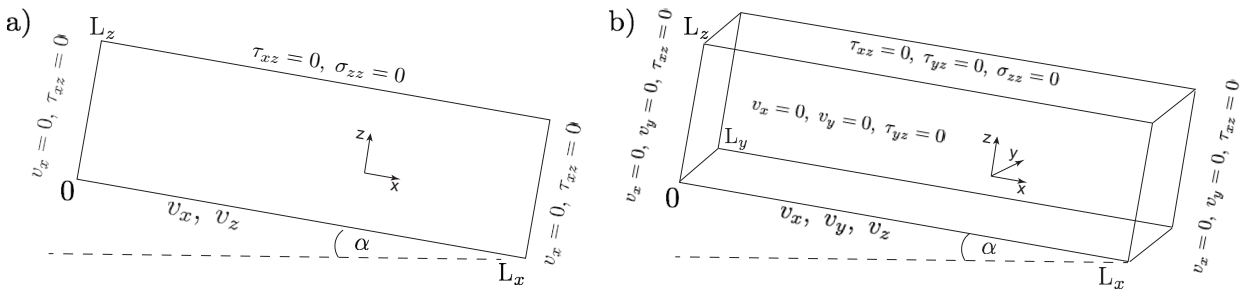


Figure 3. Model configuration for the numerical experiments: a) 2-D model and b) 3-D model. Both surface and bed topography are flat but inclined at a constant angle of α. We show both the model coordinate axes and the prescribed boundary conditions.

the coupling terms, rheology is also treated implicitly, i.e. the shear viscosity η is always evaluated using the current physical temperature, T, and strain-rate, $\dot{\epsilon}_{\mathrm{II}}$. Our method is fully local. At no point during the iterative procedure does one need to perform a global reduction, nor to access values that are not directly collocated. These considerations are crucial when designing a solution strategy that targets parallel hardware such as many-core GPU accelerators. We implemented the PT method in the MATLAB and CUDA C programming languages. Computations in CUDA C can be performed in both double and single precision arithmetic. The computations in CUDA C shown in the remainder of the paper were performed using double-precision arithmetic, if not specified otherwise.

3 Levering hardware accelerators

3.1 Implementation on graphical processing units

Our GPU algorithm development effort is motivated by the aim to resolve the coupled thermomechanical system of equations (Eq. 12-13) with high spatial and temporal accuracy in 3-D. To this end, we exploit the low-level intrinsic parallelism of shared memory devices, targeting particularly GPUs. A GPU is a massively parallel device originally devoted to render the colour values for pixels on a screen independently from one another where the latency can be masked by high throughput (i.e. compute as many jobs as possible in a reasonable time). A schematic representation (Figure 2) highlights the conceptual discrepancy between GPU and CPU. On the GPU chip, most of the area is devoted to the arithmetic units, while on the CPU, a large area of the chip hosts scheduling and control microsystems.

The development of GPU-based solvers requires that one ~~devote~~ devotes time to the design of new algorithms that lever the massively parallel potential of the current GPU architectures. Considerations such as limiting the memory transfers to the mandatory minimum, avoiding complex data layouts, preferring matrix-free solvers with low memory footprint, and optimal parallel scalability instead of classical Direct-Iterative solver types (Räss et al., 2019a) are key in order to achieve optimal performance.

Experiment	L_x	L_y	α	n	β_0	L_x^D	L_y^D	L_z^D
Exp. 1 2-D	10	–	10	3	–	2 km	–	200 m
Exp. 1 3-D	10	4	10	3	–	2 km	800 m	200 m
Exp. 2 2-D	10	–	0.1	3	0.1942	10 km	–	1 km
Exp. 2 3-D	10	10	0.1	3	0.1942	10 km	10 km	1 km

Table 1. Experiments 1 and 2: Non-dimensional model parameters and the dimensional values $\left(^{D}\right)$ for comparison.

Our implementation does not rely on the CUDA unified virtual memory (UVM) features. UVM avoids to explicitly define data transfer between the host (CPU) and device (GPU) arrays but results in about one order of magnitude lower performance. We suspect the internal memory handling to be responsible of continuously synchronising host and device memory, which is not needed in our case.

3.2 Multi-GPU implementation

We rely on a distributed memory parallelisation using the message passing interface (MPI) library to overcome the on-device memory limitation inherent to modern GPUs and exploit supercomputers' computing power. Access to a large number of parallel processes enables us to tackle larger computational domains or to refine grid resolution. We rely on domain decomposition to split our global computational domain into local domains, each executing on a single GPU handled by an MPI process. Each local process has its boundary conditions defined by a) physics if on the global boundary or b) exchanged information from the neighbouring process in case of internal boundaries. We use CUDA-aware non-blocking MPI messages to exchange the internal boundaries among neighbouring processes. CUDA-awareness allows us to bypass explicit buffer copies on the host memory by directly exchanging GPU pointers resulting in an enhanced workflow pipe-lining. Our algorithm implementation and solver requires no global reduction. Thus, there is no need for global MPI communication, eliminating an important potential scaling bottleneck. Although the proposed iterative and matrix-free solver features a high locality and should scale by construction, the growing number of MPI processes may deprecate the parallel runtime performance by about 20% owing to the increasing number of messages and overall machine occupancy (Räss et al., 2019c). We address this limitation by overlapping MPI communication and the computation of the inner points of the local domains using streams, a native CUDA feature. CUDA streams allow one to assign asynchronous kernel execution and thus enable the overlap between communication and computation, resulting in optimal parallel efficiency.

4 The model configuration

To verify the numerical implementation of the developed ~~PT~~ FastICE solver, we consider three numerical experiments based on a box inclined at a mean slope angle of α. We perform these numerical experiments on both 2-D and 3-D computational domains (Figure 3a and 3b, respectively). The non-dimensional computational domains are $\Omega_{2D} = [0 \quad L_x] \times [0 \quad L_z]$ and

Experiment	L_x	L_y	L_z	α	n	$\overline{F}$	T_0	L_x^D	L_y^D	L_z^D	T_0^D
Exp. 3 1-D	–	–	3×10^5	10	3	2.8×10^{-8}	9.15	–	–	300 m	-10 °C
Exp. 3 2-D	$10L_z$	–	3×10^5	10	3	2.8×10^{-8}	9.15	3 km	–	300 m	-10 °C
Exp. 3 3-D	$10L_z$	$4L_z$	3×10^5	10	3	2.8×10^{-8}	9.15	3 km	1.2 km	300 m	-10 °C

Table 2. Experiment 3: Non-dimensional model parameters and the dimensional values $\left(^{\mathrm{D}}\right)$ for comparison

$\Omega_{3\mathrm{D}} = [0 \quad \mathrm{L}_x] \times [0 \quad \mathrm{L}_y] \times [0 \quad \mathrm{L}_z]$ for 2-D and 3-D domains, respectively. The difference between the 2-D and the 3-D configurations lies in the boundary conditions imposed at the base and at the lateral sides. At the surface, the zero stress $\sigma_{ij} n_j = 0$ boundary condition is prescribed in all experiments. Experiment 2's model configuration corresponds to the ISMIP benchmark (Pattyn et al., 2008), where experiment C relates to the 3-D case and experiment D relates to the 2-D case.

Experiments 1 and 2 seek to first verify the implementation of the mechanical part of the Stokes solver, which is the computationally most expensive part (Eq. 8). For these experiments, we assume that the ice is isothermal and neglect temperature. We compare our numerical solutions to the solutions obtained by the commonly used finite-element Stokes solver Elmer/Ice (Gagliardini et al., 2013), which has been thoroughly tested (Pattyn et al., 2008; Gagliardini and Zwinger, 2008). Experiment 3 is a thermomechanically coupled case. The model parameters are the stress exponent n, the mean bed slope α and the two horizontal distances L_x and L_y in their respective dimensions (x, y), and appear in Table 1. If a linear basal sliding law (Eq. 6) is prescribed, the respective 2-D and 3-D sliding coefficients are:

$$\begin{aligned} \beta^2(x) &= \beta_0 \left[1 + \sin\left(\frac{2\pi x}{\mathrm{L}_x}\right)\right] , \\ \beta^2(x,y) &= \beta_0 \left[1 + \sin\left(\frac{2\pi x}{\mathrm{L}_x}\right) \sin\left(\frac{2\pi y}{\mathrm{L}_y}\right)\right] , \end{aligned} \tag{20}$$

where β_0 is a chosen non-dimensional constant. Differences may arise depending on the prescribed values for the parameters α, L_x, L_y and β_0. Experiment 2 represents the ISMIP experiments C and D for $\mathrm{L} = 10$ km (Pattyn et al., 2008), but in our case using non-dimensional variables.

The mechanical part of Experiment 3 is analogous to Experiment 2. The boundary conditions are periodic in x and y directions unless specified otherwise. The thermal problem requires additional boundary conditions in terms of temperature or fluxes. We set the surface temperature T_0 to 0. At the bottom, we set the vertical flux q_z to 0 and, on the sides, we impose periodic boundary conditions. The model parameters used in Experiment 3 are compiled in Table 2. We employ the semi-analytical 1-D model (Section 2.3) as an independent benchmark for the Experiment 3 calculations.

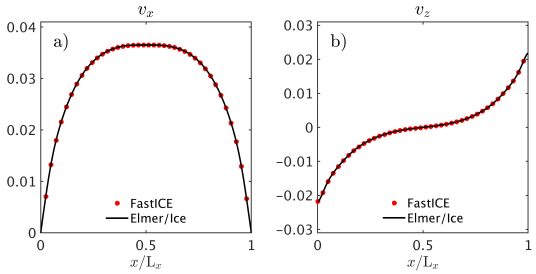


Figure 4. Comparison of the non-dimensional simulation results for the 2-D configuration of Experiment 1. We show a) the horizontal component of the surface velocity, v_x, and b) the vertical component of surface velocity, v_z, across the ice slab for both our FastICE model and Elmer/Ice. For context, the maximum horizontal velocity ($v_x \approx 0.0365$) corresponds to ≈ 174 m/yr. The horizontal distance is 2 km, while the ice thickness is 200 m. The box is inclined at 10°.

5 Results and performance

5.1 Experiment 1: Stokes flow without basal sliding

We compare our numerical solutions obtained with the GPU-based PT method using a CUDA C implementation (FastICE) to the reference Elmer/Ice model. We report all the values in their non-dimensional form, and the horizontal axes are scaled with their aspect ratio. ~~In Figure 4, we plot both horizontal v_x and vertical v_z~~ We impose a no-slip boundary condition on all velocity components at the ~~top surface for Experiment 1 in 2-D. Since the~~ base and prescribe free-slip boundary conditions on all lateral domain sides. We prescribe a stress-free upper boundary in the vertical direction.

In the 2-D configuration (Figure 4), the horizontal velocity component vanishes at the left and right boundary, $v_x = 0$, thus the maximum velocity values in the horizontal direction are located in the middle of the slab. ~~We impose a no-slip boundary condition on all velocity components at the base and prescribe free-slip boundary conditions on all lateral domain sides. We prescribe a stress-free upper boundary in the vertical direction.~~ On the left side ($x/L_x = 0$), the ice is pushed down (compression); thus, the vertical velocity values were negative. On the right side ($x/L_x = 1$), the ice is pulled up (extension), and the vertical velocity values were positive. Our ~~PT GPU-based~~ FastICE results agree well with the numerical solutions produced by Elmer/Ice. The numerical resolution of the Elmer/Ice model is 1001×275 grid-points in x and z directions ($\approx 8.25 \times 10^5$ degrees of freedom (DOF)), while we employed 2047×511 grid-points ($\approx 3.13 \times 10^6$ DOF) within our PT method. We use higher numerical grid resolution within FastICE to jointly verify agreement with Elmer/Ice and convergence. The fact that we obtain matching results when increasing grid resolution significantly suggests that we resolve the relevant physical processes sufficiently, even at relatively low resolution. We report an exception to this trend in the 3-D case of Experiment 2. The PT method's efficiency enables considering the large number of grid-points without affecting the runtime. The DOF represent three variables in 2-D (v_x, v_z, P) and four variables in 3-D (v_x, v_y, v_z, P) multiplied by the number of grid-points involved.

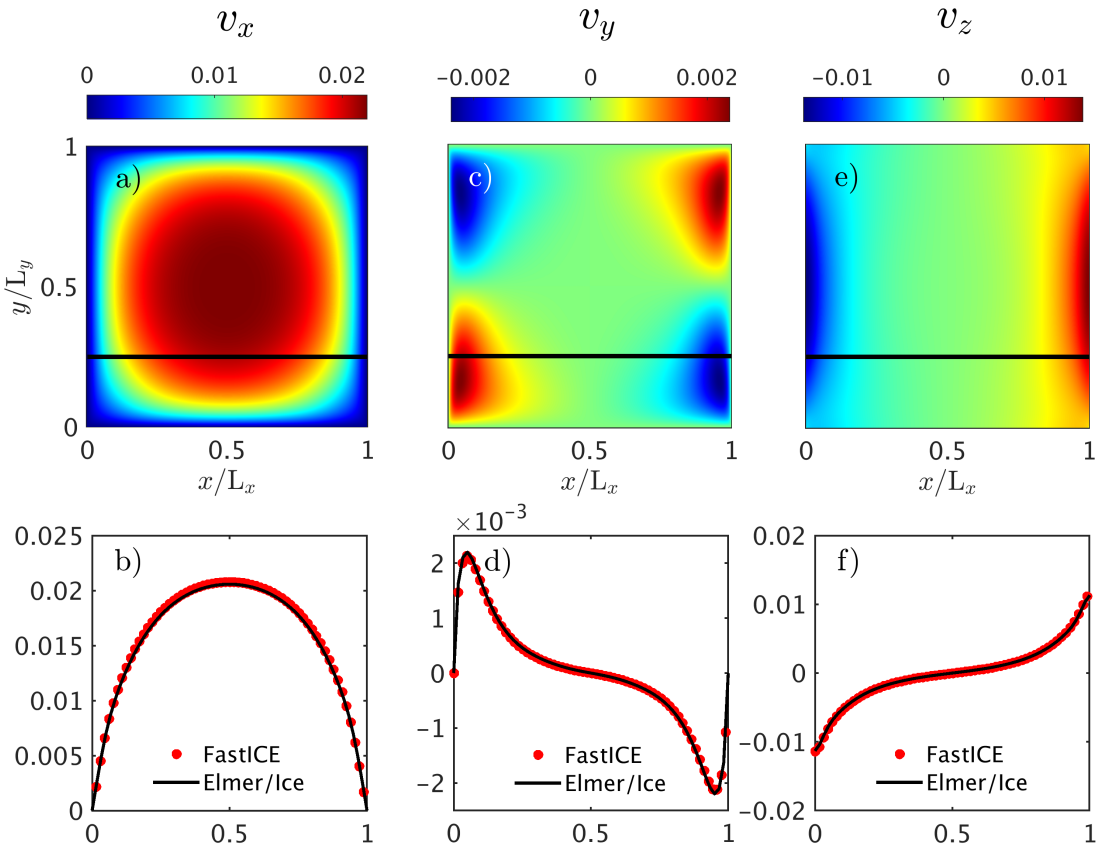


Figure 5. Non-dimensional simulation results for the 3-D configuration of Experiment 1. We report a) the horizontal surface velocity component v_x, c) the horizontal surface velocity component v_y, and e) the vertical surface velocity component v_z. The black solid line depicts the position where $y = \mathrm{L}_y/4$. Panels b) d) and f) show the surface velocity components v_x, v_y and v_z, respectively, at $y = \mathrm{L}_y/4$ and compare them against the results from the Elmer/Ice model.

~~Figure 5 shows the results for the 3-D configuration of Experiment 1. It plots our computed horizontal v_x, v_y and vertical v_z velocity components at the top surface (Figure 5a,c,e) and compares them to the reference solution from Elmer/Ice at $y \approx \mathrm{L}_y/4$ (Figure 5b,d,f).~~ We find good agreement between the two model solutions in the 3-D configuration as well (Figure 5). We employed a numerical ~~resolution~~ grid resolution of $319 \times 159 \times 119$ grid-points in x, y and z directions ($\approx 2.41 \times 10^7$ DOF), and used a numerical grid resolution of $61 \times 61 \times 21$ ($\approx 3.1 \times 10^5$ DOF) in Elmer/Ice. Scaling our result to dimensional values (Table 1) results in maximal horizontal velocity (v_x) of ≈ 105 m/yr. The horizontal distance is 2 km in the x-direction and 800 m in the y-direction, and the ice thickness is 200 m. The box is inclined of 10°.

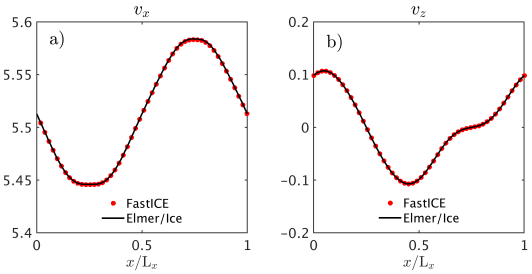


Figure 6. Non-dimensional simulation results for the 2-D configuration of Experiment 2. We plot a) the horizontal surface velocity component v_x and b) the vertical surface velocity component v_z across the slab for both our FastICE model and Elmer/Ice. In dimensional terms, the maximum horizontal velocity ($v_x \approx 5.58$) corresponds to ≈ 16.9 m/yr. The horizontal distance is 10 km, while the ice thickness is 1 km. The box is inclined at 0.1°.

5.2 Experiment 2: Stokes flow with basal sliding

We ~~now~~ then consider the case where ice is sliding at the base (ISMIP experiments C and D). We prescribe periodic boundary conditions at the lateral boundaries and apply a linear sliding law at the base. The top boundary remains stress-free in the vertical direction. ~~Figure 6 shows the results of the~~

We performed the 2-D simulation of Experiment 2 ~~, where we employed~~ (Figure 6) using a numerical grid resolution of 511×127 grid-points ($\approx 1.95 \times 10^5$ DOF) for the ~~PT GPU-based~~ FastICE solver and computed the Elmer/Ice solution using a numerical grid resolution of 241×120 ($\approx 8.7 \times 10^4$ DOF). We show both v_x and v_z velocity components at the slab's surface. The two models' results agree well.

~~The~~ We performed the 3-D simulation ~~results for~~ of Experiment 2 ~~appear in Figure 7. The upper panels~~ (Figure 7~~a,c,e) show the spatial pattern in the three surface velocity components v_x, v_y and v_z computed with our PT GPU-based solver. The lower panels (Figure 7b,d,f) compare the three surface velocity components at $y \approx L_y/4$ computed by our PT GPU-based solver to Elmer/Ice. We employed~~) using a numerical grid resolution of ~~$256 \times 256 \times 64$ ($\approx 1.67 \times 10^7$~~ $63 \times 63 \times 21$ ($\approx 3.33 \times 10^5$ DOF) for our ~~PT GPU-based~~ FastICE solver and a numerical grid resolution of $61 \times 61 \times 21$ ($\approx 3.12 \times 10^5$ DOF) in the Elmer/Ice model. In dimensional units, the maximum horizontal velocity (v_x) corresponds to ≈ 16.4 m/yr. The horizontal distance is 10 km in the x-direction 10 km in the y-direction, and the ice thickness is 1 km. The box is inclined at 0.1°.

We find good agreement between the two numerical implementations~~, despite some discrepancies in the horizontal velocity component v_y. A potential explanation for the minor mismatch is the fact that the finite-element grid does not exactly coincide with the location $y = L_y/4$ in Elmer/Ice, which may be resolved by specifically pinning nodes of the finite-element mesh.~~ . Since the flow is mainly oriented in the x direction, the v_y velocity component is more than two orders of magnitude smaller than the v_x velocity component. Numerical errors in v_y are more apparent than in the leading velocity component v_x. We report a one-order magnitude increase in the time-to-solution in Experiment 2 compared to the Experiment 1 configuration owing to the periodicity on the lateral boundaries.

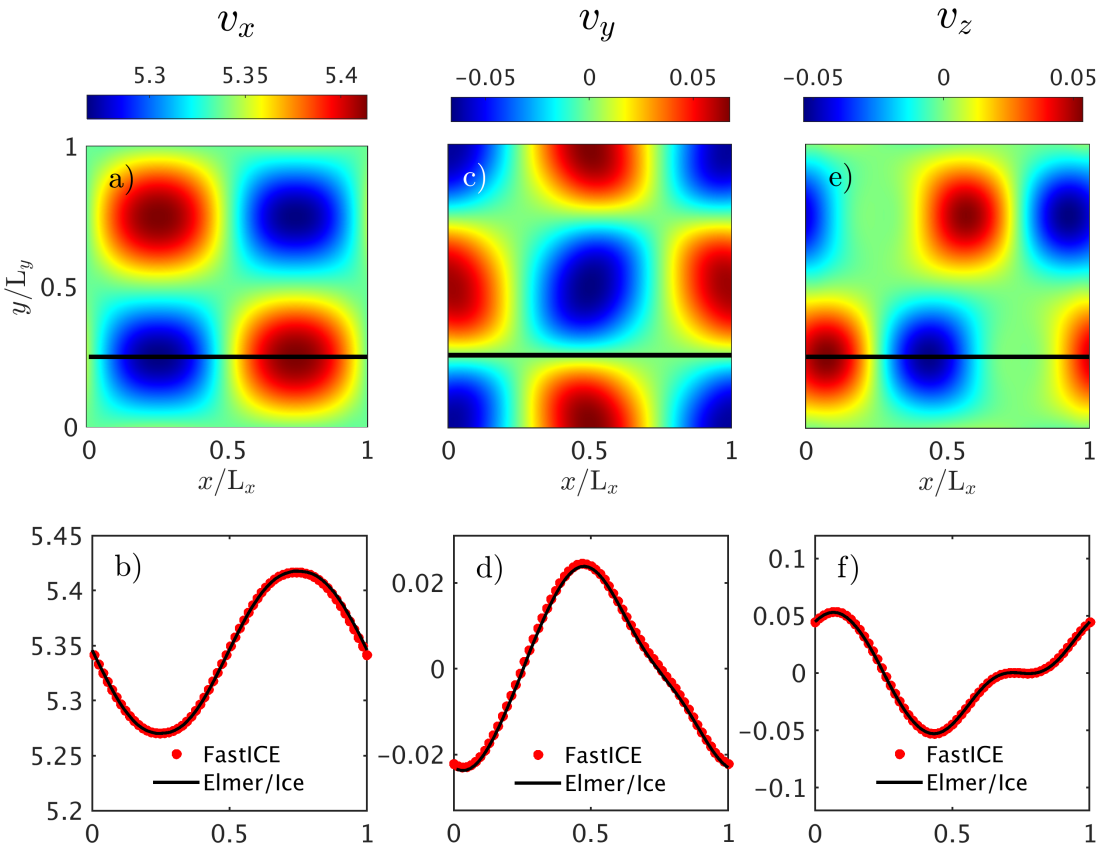


Figure 7. Non-dimensional simulation results for the 3-D configuration of Experiment 2. We report a) the horizontal surface velocity component v_x, c) the horizontal surface velocity component v_y and e) the vertical surface velocity component v_z. The black solid line depicts the position where $y = L_y/4$. Panels b) d) and f) show the surface velocity components v_x, v_y and v_z, respectively, at $y = L_y/4$ and compare them against the results from the Elmer/Ice model.

We employ a matching numerical resolution between FastICE and Elmer/Ice in this particular benchmark case. Using higher resolution for FastICE results in minor discrepancy between the two solutions, suggesting that the resolution in Figure 7 is insufficient to capture small-scale physical processes. We discuss this issue more in Section 5.5 where we test the convergence of the FastICE numerical implementation upon grid refinement.

5.3 Experiment 33a: Thermomechanically coupled Stokes flow without basal sliding

We first verify that both the 1-D, 2-D and 3-D model configurations from Experiment 3 produce identical results assuming periodic boundary conditions on all lateral sides. In this case, all the variations in the x or y directions vanish ($\partial/\partial x$ and $\partial/\partial y$);

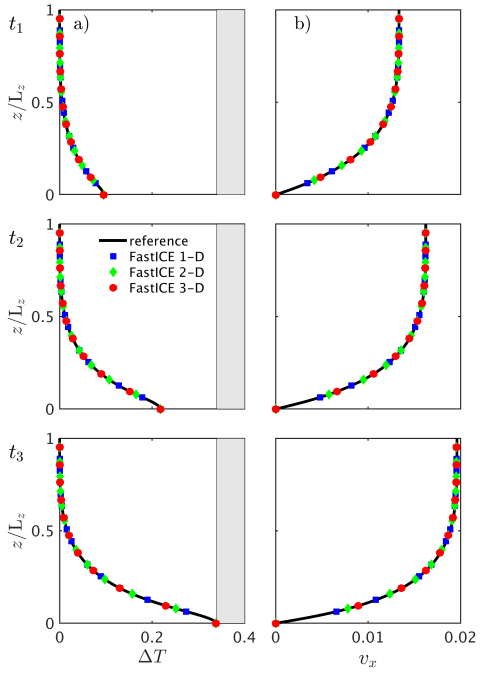


Figure 8. Non-dimensional simulation results for a) the temperature deviation T and b) the horizontal velocity component v_x for the 1-D, 2-D and 3-D FastICE models at three different non-dimensional times 0.7×10^8, 1.4×10^8 and 1.9×10^8 and compare them to the 1-D reference model results. We employ a vertical grid resolution n_z of $31, 95$ and 201 grid-points. We sample the 1-D profiles at location $x = \mathrm{L}_x/2$ in 2-D and at $x = \mathrm{L}_x/2$ and $y = \mathrm{L}_y/2$ in 3-D. The shaded areas correspond to the part of the solution that is above the melting temperature, since we do not account for phase transitions in this case.

thus, both the 2-D and 3-D models reduce to the 1-D problem. We employ a numerical grid resolution of $127 \times 127 \times 127$ grid-points in x, y and z direction, 127×127 grid-points in x and z directions and 127 grid-points in the z direction for the 3-D, 2-D and 1-D problems, respectively.

We ensure that all results collapse onto the semi-analytical 1-D model solution (Section 2.3), which we obtained by analytically integrating the velocity field and solving the decoupled thermal problem separately (Eq. 11). From a computational perspective, we numerically solve Eq. 11 using a high spatial and temporal accuracy and therefore minimise the occurrence of numerical errors. We establish the 1-D reference solution for both the temperature and the velocity profile, solving Eq. 11 on a regular grid, reducing the physical time steps until we converge to a stable reference solution. Our reference simulation involves 4000 grid-points and a non-dimensional time step of 5×10^5 (using a backward Euler time integration). We reach the total simulation time of 2.9×10^8 within 580 physical time steps.

We report overall good agreement of all model solutions (1-D, 2-D, 3-D and 1-D reference) at the three reported stages for this scenario (Figure 8). As expected from the 1-D model solution, temperature varies only as a function of time and depth with

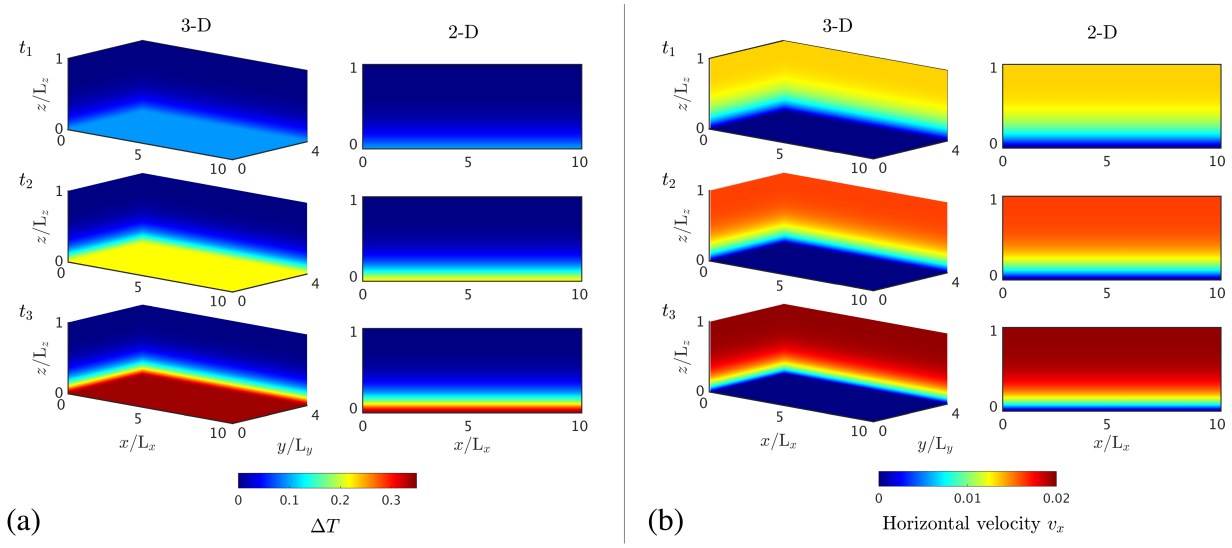


Figure 9. Spatial distribution of a) the temperature deviation from the initial temperature T and b) the horizontal velocity component v_x for the 3-D (left column) and the 2-D (right column) in non-dimensional units. We scale the domain extend with L_z. We compare the numerical solutions at non-dimensional times 0.7×10^8, 1.4×10^8 and 1.9×10^8.

the highest value obtained close to the base and for longer simulation times. Similarly, the velocity profile is equivalent to the 1-D profile and the largest velocity value is located at the surface. We only report the horizontal velocity component v_x for the 2-D and the 3-D models, since v_y and v_z feature negligible magnitudes. Thus, we only observe spatial variation in the vertical z direction. We report the non-dimensional temperature T (Figure 9a) and horizontal velocity v_x (Figure 9b) fields for both the 3-D and the 2-D configurations compared at non-dimensional time 0.7×10^8, 1.4×10^8 and 1.9×10^8. The dimensional results from Experiment 3 correspond to a 300 m thick ice slab inclined at 10° angle with an initial surface temperature of -10°C. The maximum initial velocity for the isothermal ice slab corresponds to $\approx 486\ \mathrm{m/yr}$, while the maximum velocity just before the melting point is reached corresponds to 830 $\mathrm{m/yr}$. The comparison snapshot times are 1.6, 3.2 and 4.4 years.

The semi-analytical 1-D solution enables us to evaluate the influence of the numerical coupling method and time integration and to quantify when and why high spatial resolution is required in thermomechanical ice flow simulations. We compare the 1-D semi-analytical reference solution (Eq. 11) to the results obtained with the 1-D ~~PT-based~~ FastICE solver for three spatial numerical resolutions (n_z =31, 95 and 201 grid-points) at three non-dimensional times 1×10^8, 2×10^8 and 2.9×10^8 (Figure 10). The grey area in Figure 10 highlights where the melting temperature is exceeded. Since our semi-analytical reference solution does not include phase transitions, we also neglect this component in the numerical results. During the early stages of the simulation, the thermomechanical coupling is still minor and solutions at all resolution levels are in good agreement with one another and with the reference. The low resolution solution starts to deviate from the reference (Figure 10b) when the coupling become more pronounced close to the thermal runaway point (Clarke et al., 1977). The high spatial resolution

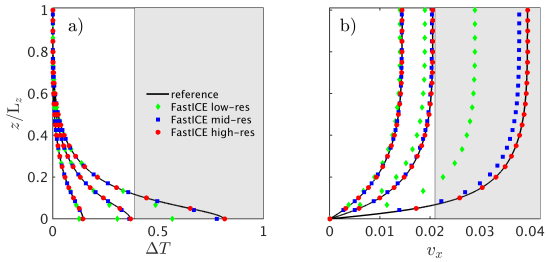


Figure 10. Non-dimensional simulation results for a) the temperature deviation T and b) the horizontal velocity component v_x to test solver performance at three resolutions. The vertical resolutions are $\mathrm{LR} = 31$, $\mathrm{MR} = 95$ and $\mathrm{HR} = 201$ grid-points for low-, mid- and high-resolution runs, respectively. We compare the results for non-dimensional time 1×10^8, 2×10^8 and 2.9×10^8. The shaded areas correspond to the part of the solution that is above the melting temperature, since we do not account for phase transitions in this benchmark.

solution is satisfactory at all stages. We conclude that high spatial resolutions is required to accurately capture the non-linear coupled behaviour in regimes close to the thermal runaway, which is seldom the case in the models reported in the literature.

Thermomechanical strain localisation may significantly impact on the long-term evolution of a coupled system. A recent study by Duretz et al. (2019) suggested that partial coupling may result in under-estimating the thermomechanical localisation compared to the fully coupled approach, as reported in their Figure 8. We compare three coupling methods (Figure 11): (1) A fully coupled implicit PT method, as described in the numerical section, where the viscosity and the shear-heating term are implicitly determined by using the current guess. (2) An implicit numerically uncoupled mechanical and thermal model. (3) An explicit numerically uncoupled mechanical and thermal model. The numerical time integration in physical time is performed using an implicit backward Euler method for (1) and (2) and a forward Euler explicit time integration method for (3). We utilise the identical non-dimensional time step for both the explicit and the implicit numerical time integration. We perform 580 time steps, reaching a simulation time of 2.9×10^8. We employ a vertical grid resolution of $n_z = 201$ grid-points for all models. The chosen time step for the explicit integration of the heat diffusion equation is below the CFL stability condition given by $\Delta z^2/2.1$ in 1-D, where Δz represent the grid spacing in a vertical direction.

Physically, the viscosity and shear-heating terms are coupled and are a function of temperature and strain-rates, but we update the viscosity and the shear-heating term based on temperature values from the previous physical time step. Thus, the shear-heating term can be considered as a constant source term in the temperature evolution equation during the time step, leading to a semi-explicit rheology. We show the 1-D numerical solutions of (blue) the fully coupled method with a backward Euler (implicit) time integration and the two uncoupled methods with either (green) backward (implicit) or (red) forward (explicit) Euler time integration (Figure 11) and compare them to the 1-D reference model solution. Surprisingly, and in contrast to Duretz et al. (2019), we observe a good agreement between all methods, suggesting that the different coupling strategies capture the coupled flow physics with sufficient accuracy given high enough spatial and temporal resolution. However, for a longer-term evolution, the uncoupled approaches may predict lower temperature and velocity values than the fully coupled approach.

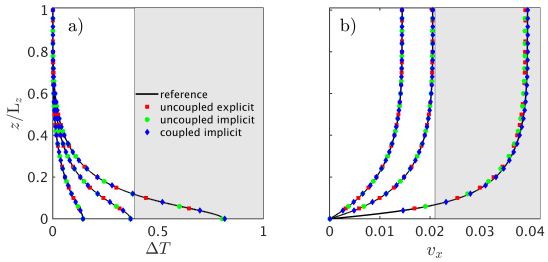


Figure 11. Non-dimensional simulation results for a) the temperature deviation T and b) the horizontal velocity component v_x to evaluate different numerical time integration schemes. We consider three non-dimensional time 1×10^8, 2×10^8 and 2.9×10^8 and compare our numerical estimates to the reference model. As before, the shaded areas correspond to the part of the solution that is above the melting temperature, since we neglect phase transitions in this comparison.

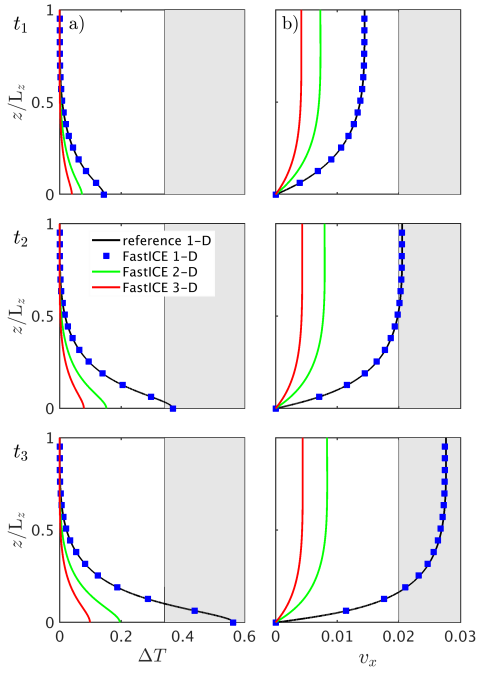


Figure 12. Non-dimensional simulation results for a) the temperature deviation T and b) the horizontal velocity component v_x for the 1-D, 2-D and 3-D FastICE models at three non-dimensional times 1×10^8, 2×10^8 and 2.5×10^8 compared to our analytical solution. We sample the 1-D profiles at location $x = \mathrm{L}_x/2$ in 2-D and at $x = \mathrm{L}_x/2$ and $y = \mathrm{L}_y/2$ in 3-D. The shaded area corresponds to the part of the solution that is above the melting temperature, approximately 0.35 of the temperature deviation.

5.4 Experiment 33b: Thermomechanically coupled Stokes flow in a finite domain

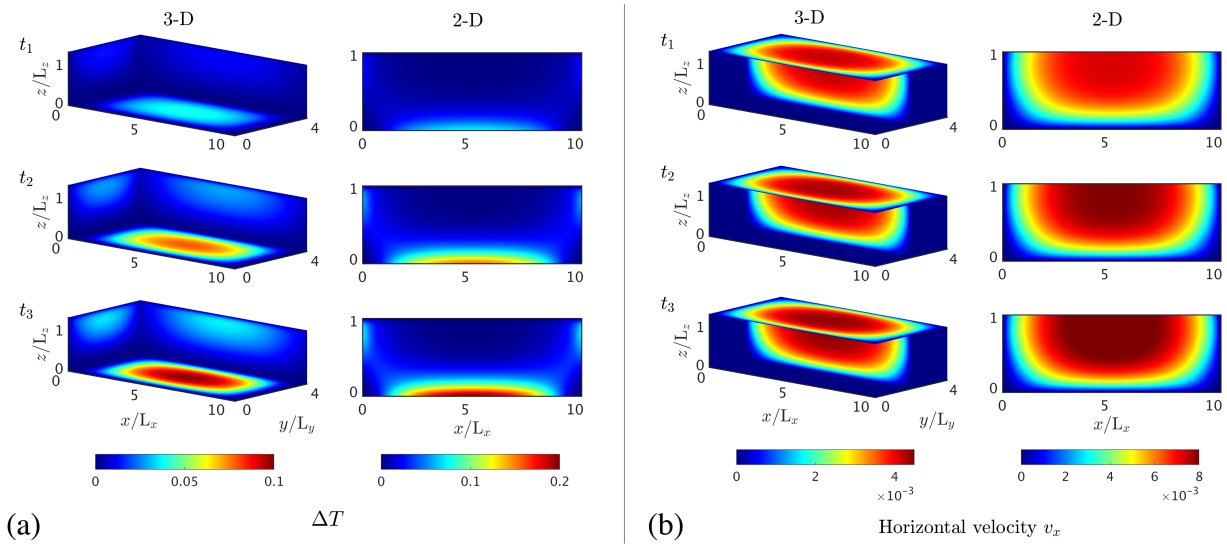


Figure 13. Non-dimensional simulation results of a) the temperature deviation from the initial temperature T and b) the horizontal velocity component v_x for Experiment 3 at three non-dimensional times 1×10^8, 2×10^8 and 2.5×10^8 for both the 2-D and 3-D configurations.

Boundary conditions corresponding to immobile regions in the computational domain may induce localisation of deformation and flow observed in locations such as shear margins, grounding zones or bedrock interactions. Dimensionality plays a key role in such configurations, causing the stress distribution to be variable among the considered directions.

We used the configuration in Experiment 3 to investigate the spatial variations in temperature and velocity distributions by defining no-slip conditions on the lateral boundaries for the mechanical problem and hindering any heat flux through those boundaries. We employ a numerical grid resolution of $511 \times 255 \times 127$ grid-points, 511×127 grid-points and 201 grid-points for the 3-D, 2-D and 1-D case, respectively. We prescribe a non-dimensional time step of 5×10^5. We perform 500 numerical time steps and reach a total non-dimensional simulation time of 2.5×10^8. We then compare the temperature T and horizontal velocity component v_x at three times obtained with the 1-D, 2-D and 3-D ~~PT GPU-based~~ FastICE solver to the reference solution (Figure 12). We use 1-D profiles for comparison, taken at location $x = \mathrm{L}_x/2$ in the 2-D model and at location $x = \mathrm{L}_x/2$ and $y = \mathrm{L}_y/2$ in the 3-D model. We also report the temperature variation ΔT (Figure 13a) and the horizontal velocity component v_x (Figure 13b) for both the 2-D and 3-D simulations. The melting temperature approximately corresponds to 0.35 of the temperature deviation. The reported results correspond to a $2.3-$, $4.6-$ and $5.8-$ year evolution.

All three models start with identical initial conditions for the thermal problem, i.e. $\Delta T = 0$ throughout the entire ice slab. The difference between the models arises owing to different stress distributions in 1-D, 2-D or 3-D. For instance, the additional stress components inherent in 2-D and 3-D are in the same order of magnitude as the 1-D shear stress for the considered aspect ratio, reducing the horizontal velocity v_x in the 2-D and 3-D models. This also impacts on the shear-heating term, reducing the source term in the temperature evolution equation. In the 1-D configuration, the unique shear stress tensor component is a

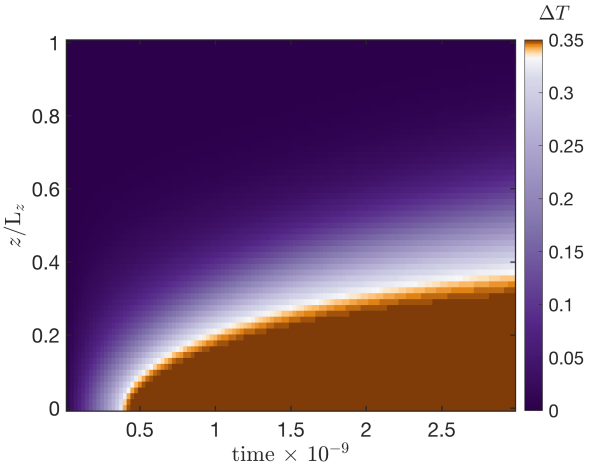


Figure 14. Experiment 3 includes a phase transition owing to melting. We report the evolution in time of non-dimensional temperature variation ΔT along a vertical profile picked at location $x = \mathrm{L}_x/2$ within a 2-D run from Experiment 3. For this purpose, we run the 2-D FastICE models from Experiment 3 for a duration of 2.9×10^9.

function only of depth. On the other end-member, the 3-D configurations allow for a spatially more distributed stress state. They lower strain-rates in this scenario and reduce the magnitude of shear-heating in higher dimensions. The spatially heterogeneous temperature and strain-rate fields in all directions require the utilisation of sufficiently high spatial numerical resolution in all directions in order to accurately resolve spontaneous localisation.

We did not consider phase transition in the previous experiments for the sake of model comparison and because the analytical solution excluded this process. The existence of a phase transition caps the temperature at the pressure melting point in regions with pronounced shear-heating, as illustrated in 2-D in Figure 14. The simulation represents the thermomechanically coupled Experiment 3 with no-sliding and heat impermeable walls (similar to Figure 13). Meltwater production consumes excess heat generated by shear-heating. Thus, melting provides a physical mechanism that avoids thermal runaway in shear-heating dominated zones in the ice. The experiment duration in dimensional units is 70 years, and the maximal temperature increase is 10°C upon reaching the melting point.

5.5 Validation of the FastICE numerical implementation

In order to confirm the accuracy of the FastICE numerical implementation, we report truncation errors (L2-norms) upon numerical grid refinement. We consider both the 2-D and 3-D configurations of Experiment 2 for this convergence test. We vary the numerical grid resolution keeping the relative grid step $\Delta x, \Delta y$ (and Δz in 3-D) ratio. We utilise a high-resolution numerical simulation as reference and perform three additional simulations where we keep dividing the number of grid points

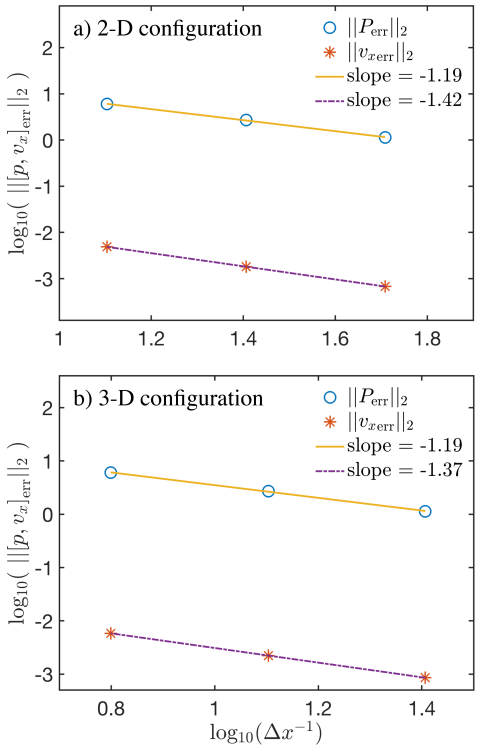


Figure 15. Evolution of velocity and pressure truncation errors (L2-norm) upon grid refinement for a) the 2-D configuration and b) the 3-D configuration of the Experiment 2.

in both x, y (and z in 3-D) direction by a factor 2. We report the L2-norms:

$$||P_{\mathrm{err}}||_2 = ||P_{\mathrm{ref}} - P_{\mathrm{coarse}}||_2 \,, \qquad ||v_{x\mathrm{err}}||_2 = ||v_{x\mathrm{ref}} - v_{x\mathrm{coarse}}||_2 \,, \tag{21}$$

for both the pressure P and the horizontal down slope v_x velocity component on a logarithmic plot for both the 2-D (Figure 15a) and 3-D configurations (Figure 15b). The FastICE numerical implementation converges with increasing numerical resolution and we report linear fitting slopes of -1.19 for pressure and of about -1.4 for horizontal velocity component.

We additionally report the behaviour of the residuals' converge as function of the nonlinear iterations $n_{\mathrm{iter}}^{\mathrm{nonlin}}$ for the FastICE GPU-based implementation (Figure 16a). The reported convergence history stands for a 2-D configuration of the Experiment 3 and a numerical grid resolution of 511×127 grid points. The optimal damping parameter used in this case is $\nu = 2$ (Eq. 19). We further report the sensitivity of the accelerated PT scheme on the damping parameter ν (Figure 16b). We show that selecting the optimal damping parameter (in the reported case $\nu = 2$) ensures a relative low number of iterations to converge both the linear and nonlinear thermomechanical problem.

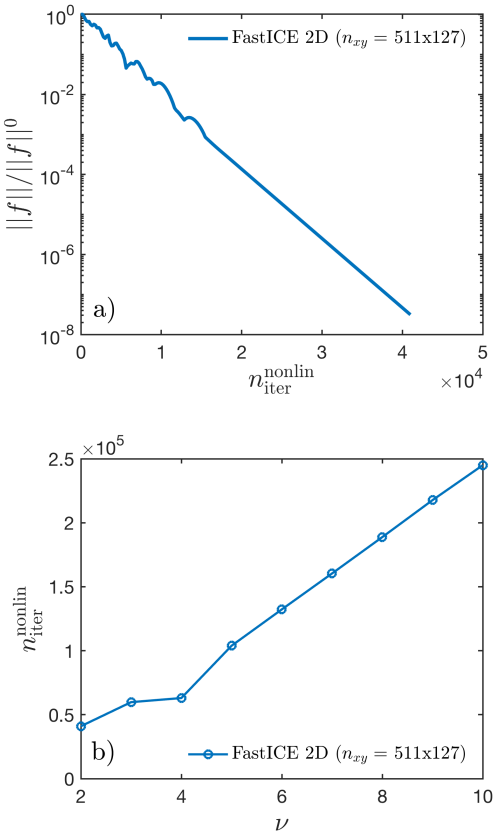


Figure 16. Residual evolution and convergence efficiency of the 2-D FastICE GPU-based implementation for a numerical grid resolution of 511×127 grid points targeting a relative nonlinear tolerance of $\text{tol}_{\text{nonlin}} = 1e{-}8$. a) Relative total non-linear residuals $f = \max(f_P, f_{v_i}, f_T)$ as function of non-linear iterations and b) the nonlinear iteration count as function of the damping parameter ν (Eq. 19).

5.6 The computational performance

We used two metrics to assess the performance of the developed FastICE PT algorithm: the effective memory throughput (MTP_{eff}) and the wall-time. We first compare the effective memory throughput of the vectorised MATLAB CPU implementation and the single-GPU CUDA C implementation. We employ double-precision (DP) floating-point arithmetic in CUDA C for fair comparison. Second, we employ the wall-time metric to compare the performance of our various implementations (MATLAB, CUDA C) and compare these to the time-to-solution of the Elmer/Ice solver.

We use two methods to solve the linear system in Elmer/Ice. In the 2-D experiments, we use a direct method and in 3-D, an iterative method. The direct method used in 2-D relies on the UMFPACK routines to solve the linear system. To solve the

3-D experiments, we employ the available bi-conjugate gradient stabilised method (BICGstab) with an ILU0 preconditioning. We employ the configuration in Experiment 1 for all the performance measurements. We use an Intel i7 4960HQ 2.6 GHz (Haswell) four-core CPU to benchmark all the CPU-based calculations. For simplicity, we only ran single-core CPU tests, staying away from any CPU parallelisation of the algorithms. Thus, our MATLAB or the Elmer/Ice single-core CPU results are not representative of the CPU hardware capabilities, and are only reported for reference.

The FastICE PT solver relies on evaluating a finite-difference stencil. Each cell of the computational domain needs to access neighbouring values in order to approximate derivatives. These memory access operations are the performance bottleneck of the algorithm, making it memory-bounded. Thus, the algorithm's performance depends crucially on the memory transfer speed, and not the rate of the floating-point operations. Memory-bounded algorithms place additional pressure on modern many-core processors, since the current chip design tends to large flop-to-byte ratios. Over the past years and decades, the memory bandwidth increase has been much slower compared to the increase in the rate of floating-point operations.

As shown by Omlin (2017) and Räss et al. (2019a), a relevant metric to assess the performance of memory-bounded algorithms is the effective memory throughput ($\mathrm{MTP}_{\mathrm{eff}}$) (Eq. 22). The $\mathrm{MTP}_{\mathrm{eff}}$ determines how efficiently data is transferred between the main memory and the arithmetic units and is inversely proportional to the execution time:

$$\mathrm{MTP}_{\mathrm{eff}} = \frac{(n_x n_y n_z) n_{\mathrm{iter}}\ n_{\mathrm{IO}}\ n_{\mathrm{p}}}{1024^3\ t_{\mathrm{nt}}} \qquad [\mathrm{GB/s}] \tag{22}$$

where $(n_x n_y n_z)$ stands for the total number of grid-points, n_{iter} is the total number of numerical iterations performed, n_{p} is the arithmetic precision (single – 4 bytes or double – 8 bytes), t_{nt} is the wall-time in seconds needed to compute the n_{iter} iterations, and n_{IO} is the performed number of memory accesses. It represents the minimum number of memory operations (read-and-write or read only) required to solve a given physical problem. For instance, in the mechanical Stokes solver for Experiment 1, we have to update (read-and-write) three arrays (v_x, v_z and P) at every iteration in 2-D and four arrays (v_x, v_y, v_z and P) at every iteration in 3-D. Thus, the update of the mandatory arrays requires a minimum of six (eight) read-and-write operations in 2-D (3-D). One additional read-and-write is needed to resolve the non-linear viscosity; thus, $n_{\mathrm{IO}} = 10$ in 2-D case and $n_{\mathrm{IO}} = 12$ in 3-D.

We report $\mathrm{MTP}_{\mathrm{eff}}$ values obtained with the ~~PT~~ FastICE algorithm for both the vectorised MATLAB (CPU) and the CUDA C (GPU) implementations in double-precision arithmetic (Figure 17a). We also show the GPU performance using single-precision arithmetic (Figure 17a – green diamonds). The results we obtain should be compared to the peak memory throughput value $\mathrm{MTP}_{\mathrm{peak}}$ for the specific hardware used. The $\mathrm{MTP}_{\mathrm{peak}}$ reports the memory transfer rates delivered only by performing memory copy operations with no computations. This value reflects the hardware performance limit and the maximal effective memory bandwidth. We measure $\mathrm{MTP}_{\mathrm{peak}}$ values for the Intel i7 4960HQ CPU of 20 GB/s, and of 260 GB/s for the Nvidia Titan X GPU. The single-core vectorised MATLAB CPU implementation achieves about 0.7 GB/s, and the CUDA C implementation 16 GB/s. Thus, the MATLAB single-core CPU implementation reaches 3.5% of the (CPU) hardware peak value, and the CUDA C (GPU) implementation at about 6.15% and 11% of the (GPU) hardware peak value using double-precision and single-precision arithmetic, respectively. Further improvement of the GPU $\mathrm{MTP}_{\mathrm{eff}}$ values can be achieved by optimising the GPU code using more on-the-fly calculations and advanced kernel scheduling.

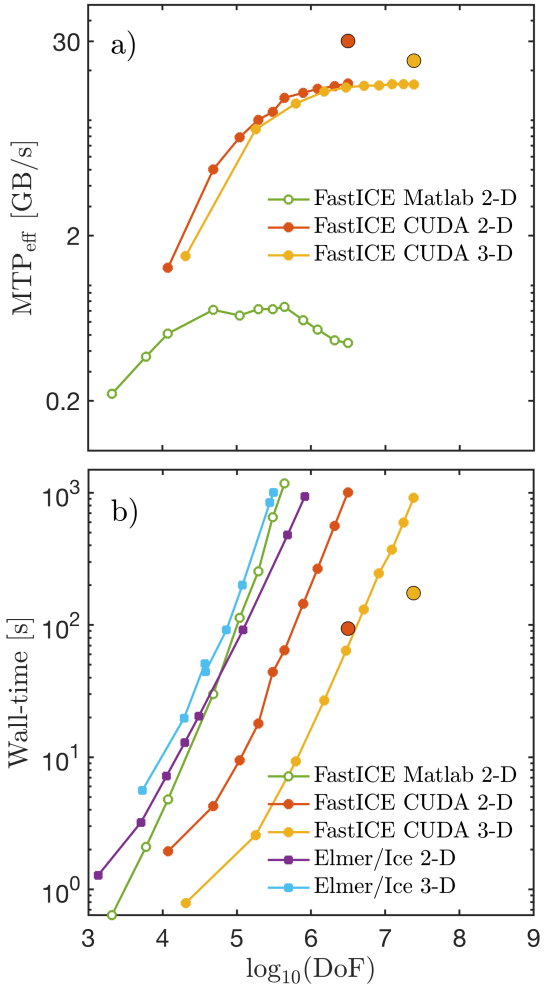


Figure 17. Performance evaluation of the FastICE mechanical solver in terms of: a) the effective memory throughput $\mathrm{MTP}_{\mathrm{eff}}$ in GB/s and b) the wall-time (in seconds) to converge the Stokes solver to a relative non-linear tolerance of $\mathrm{tol}_{\mathrm{nonlin}} = 10^{-8}$. We report the results obtained using a 2-D CPU-based single-core vectorised MATLAB implementation of FastICE, a 2-D and 3-D GPU-based CUDA C implementation of FastICE and a 2-D (direct) and 3-D (iterative) solver within the Elmer/Ice FEM single-core CPU-based model. The CPU codes are executed on an Intel i7 4960HQ CPU processor with 8 GB RAM, and the GPU codes are launched on an Nvidia Titan X (Maxwell) GPU with 12 GB on-board memory. All the computations are performed in double-precision arithmetic, with the only exception for the two single-precision GPU-based runs depicted with larger red (2-D) and orange (3-D) symbols. The single-core FastICE CPU MATLAB and Elmer/Ice results are shown for reference; they are not meant for performance comparison because we did not enable multi-threading in MATLAB and did not have access to a parallel version of Elmer/Ice.

We investigate the wall-time to solve one time step with the ~~PT GPU-based~~ FastICE GPU solver for both the 2-D and the 3-D configurations (Figure 17b). We found wall-times of about 15 minutes to solve $\approx 2.4 \times 10^7$ DOFs with double-precision arithmetic and only three minutes when using single-precision arithmetic on a Nvidia Titan X (Maxwell) GPU. In future investigations, one may consider comparing wall-times obtained by CPU algorithms fully enabling all cores of the CPU against wall-times for GPUs within the same price and power consumption range.

The 3-D performance results obtained on various available Nvidia GPUs are summarised in Figure 18~~)~~. We performed all the calculations using double-precision arithmetic. We compare the $\mathrm{MTP}_{\mathrm{eff}}$ and wall-time values as functions of the DOF. We tested GPUs from various price ranges and chip generations, targeting entry-level GPUs such as the Nvidia Quadro P1000 (Pascal), high-end gaming cards such as the Nvidia Titan Black (Kepler) or the Nvidia Titan X (Maxwell), and data-centre-class GPU accelerators such as the Nvidia Tesla V100 PCIe (Volta). The MATLAB implementation peak $\mathrm{MTP}_{\mathrm{eff}}$ values are about 0.46 GB/s, the Quadro P1000 (Pascal) values about 4.3 GB/s, the Titan Black (Kepler) 12.4 GB/s, the Titan X (Maxwell) 16.7 GB/s, and the Tesla V100 (Volta) 83.2 GB/s. The $\mathrm{MTP}_{\mathrm{eff}}$ values directly impact on the wall-time, since the memory bandwidth was the bottleneck. We solved a 3-D problem involving $511 \times 255 \times 127$ grid-points ($6,6 \times 10^7$ DOF) in about one hour on the Titan Black GPU, 40 minutes on the Titan X GPU, and only eight minutes on the Tesla V100 GPU. Notably, at this resolution, we employed about 4.5 GB of memory to solve the isothermal Stokes model. The results suggest that more recent GPUs such as the data-centre Tesla V100 (Volta) offer a significant (order of magnitude higher) performance increase than entry-level GPU accelerators, such as the Quadro P1000.

We share the performance of the GPU-MPI implementation of ~~our solver~~ FastICE to execute on distributed memory machines. We ~~achieved~~ achieve a weak scaling parallel efficiency of ~~93~~99% on the ~~128 Nvidia Titan X (Maxwell~~512 Nvidia K80 (Kepler) GPUs on the ~~*octopus*~~*Xstream* ~~supercomputer at the Swiss Geocomputing Centre, University of Lausanne, Switzerland. As~~ Cray CS-Storm GPU compute cluster at the Stanford Research Computing Facility. As our baseline, we ~~employed~~ use a non-MPI single GPU calculation. We then ~~repeated~~ repeate the experiment using 1 to ~~128 MPI~~ 512 MPI processes (thus GPUs) ~~processes~~ and report the normalised execution time (Figure 19). The effective drop in parallel efficiency is only ~~4~~1% involving 1 to ~~128~~ 512 MPI processes. We ~~achieved~~ achieve this close-to-optimal parallel efficiency by overlapping MPI message communication and local domain stencil calculations. We specifically ~~employed a CUDA stream~~ employ distinct CUDA streams in order to execute the communication and computation overlap asynchronously. We ~~performed~~ repeat similar experiment on both the *volta* node, an 8 Nvidia Tesla V100 32 GB (Nvlink Volta) ~~based computer~~ GPUs compute node (analogous to Nvidia's DGX-1 box) ~~, reporting a~~ and the *octopus* supercomputer hosting 128 consumer electronics Nvidia Titan X (Maxwell) GPUs at the Swiss Geocomputing Centre, University of Lausanne, Switzerland. On the *volta* node, we report a weak scaling parallel efficiency of 0.985% for a single MPI process running at 0.99% of the non-MPI reference. On the *octopus* supercomputer, we report a parallel efficiency of 95.5% with an effective drop in parallel efficiency of only 2% involving 1 to 128 MPI processes.

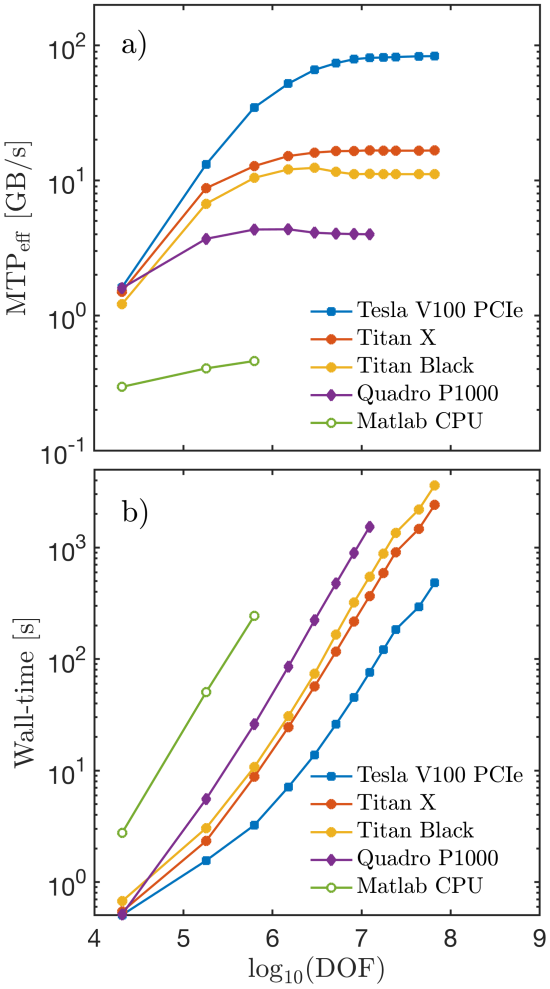


Figure 18. Performance evaluation of the FastICE mechanical solver in terms of: a) effective memory throughput $\mathrm{MTP}_{\mathrm{eff}}$ in GB/s and b) wall-time (in seconds) to converge the Stokes solver to a relative non-linear tolerance of $\mathrm{tol}_{\mathrm{nonlin}} = 10^{-8}$. We report the results from a 3-D CPU-based single-core vectorised MATLAB implementation and a 3-D GPU-based CUDA C implementation of FastICE running on different GPU chip architectures. The CPU codes are executed on an Intel i7 4960HQ CPU processor with 8 GB RAM. The GPU codes were launched on an Nvidia Titan Black (Kepler) GPU with 6 GB, an Nvidia Titan X (Maxwell) GPU 12 GB, an Nvidia Quadro P1000 (Pascal) 4 GB and an Nvidia Tesla V100 PCIe (Volta) 32 GB.

6 Discussion

Numerically resolving thermomechanical processes in ice is vital for improving our understanding of the ~~complex behaviour of ice sheets and glaciers~~physical processes that govern the transition from fast to slow ice in a changing climate. To date, very

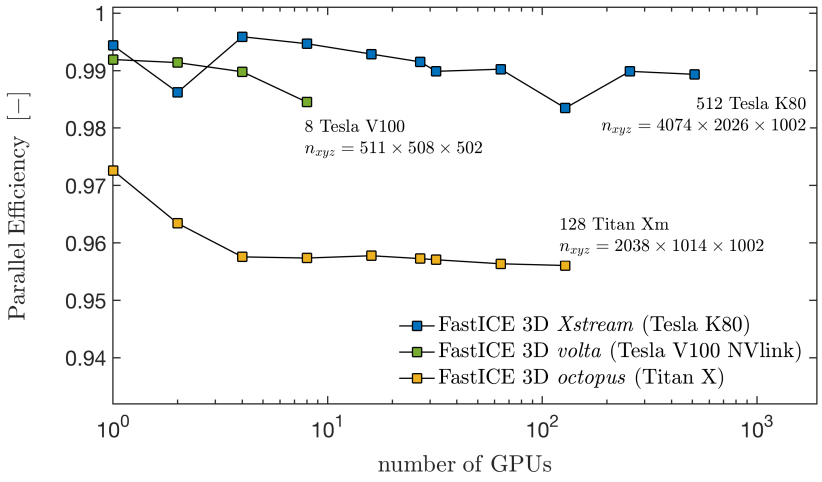


Figure 19. MPI weak scaling of the 3-D thermomechnically coupled GPU-based FastICE software. We report the parallel efficiency [-] of the numerical application on three different Nvidia hardware accelerators, the 1-512 Tesla K80 12 GB data-centre GPUs, the 1-8 Tesla V100 32 GB Nvlink data-centre GPUs and the 1-128 Titan X (Maxwell) 12 GB consumer electronics GPUs. These accelerators are available via the *Xstream* supercomputer, the *volta* node and the *octopus* supercomputer, respectively. Note that the execution time baseline used to compute the parallel efficiency represents a non-MPI calculation. We report the highest numerical grid resolution n_{xyz} achieved on each distributed memory machine.

few studies have investigated the numerical aspects of thermomechanically coupled Stokes solvers (e.g., Duretz et al., 2019). Existing assessments (e.g., Zhang et al., 2015) usually employed low spatial resolution, and did not address the influence of the numerical implementation of multi-physics coupling strategies or the role of numerical time integration. To avoid the significant computational expense of a thermomechanically coupled full Stokes model, many studies relied either on the computationally less expensive shallow ice approximations, linear or linearised Stokes models, or low spatial resolutions. None of the approaches have resolved the multi-physics and multi-scale processes governing the boundaries of streaming ice, including shear margins, grounding zones and the basal interface.

To address these limitations, we have developed ~~a new numerical model~~ FastICE, a new parallel GPU-based numerical model. The goal of FastICE is to better understand the physical processes that govern englacial instabilities such as thermomechanical localisation at the field-site, rather than the regional, scale. It hence targets other scientific problems than many existing land-ice models and complements these previous models. FastICE is based on an iterative pseudo-transient finite-difference method. Our discretisation yields to a concise matrix-free algorithm well suited to use the intrinsic parallelism of modern hardware accelerators such as GPUs. Our choices enable high-resolution 2-D and 3-D thermomechanically coupled simulations to efficiently perform on desktop computers and to scale linearly on supercomputers, both featuring GPU accelerators.

The significant temperature dependence of ice's shear viscosity leads to pronounced spatial variations in the viscosity, which affects the convergence rate of our iterative PT method. Resolving shear flow localisation is challenging in this context, since

it requires the simultaneous minimisation of errors in locations of the computational domain that are governed by different characteristic time scales. Our PT approach allows us to capture the resulting spatial heterogeneity and offers a physically-motivated strategy to locally ensure stability of the iterative scheme using local pseudo-time steps, analogous to diagonal preconditioning in matrix-based direct approaches. The conciseness and simplicity of the implementation allows us to explore influences of various coupling methods and time integrations in a straight-forward way. Similar arguments suggest that the PT approach is an interesting choice for educational purposes.

We quantify the scalability of our approach through extensive performance tests, where we investigated both the time-to-solution and the efficiency of exploiting the current hardware capabilities at their maximal capacities. To verify the accuracy and the coherence of the proposed results, we performed a set of benchmark experiments, obtaining excellent agreement with results from the widely used glacier flow model Elmer/Ice. Experiment 3 verifies that, under the assumption of periodic configurations, both 1-D, 2-D and 3-D models return matching results.

Further, we have tested the accuracy of our numerical solutions for different time integration schemes, including forward (explicit) and backward (implicit) Euler and different physical time steps. The value of the numerical time step must be chosen as sufficiently small so as to resolve the relevant physical processes. We limited the maximal time step in the explicit time integration scheme by the CFL stability criterion for temperature diffusion. For high spatial numerical resolutions, the CFL-based time step restriction is sufficient to resolve the coupled thermomechanical process. However, this conclusion is not valid for low spatial resolutions (e.g., fewer than 20 grid-points). At low resolution, the CFL-based stability condition predicts time step values larger than the non-dimensional time (2×10^8) needed to raise the temperature. Thus, we did not sufficiently resolve the physical process. An implicit scheme for the time integration remedies the stability issue, but does not guarantee accuracy. Independent of the numerical time integration scheme used, the range of time step values that resolve the coupled physics is close to the explicit stability criterion.

Our multi-GPU implementation of the thermomechanical ~~PT solver achieved~~ FastICE solver achieves a close-to-ideal parallel efficiency featuring a runtime drop of only ~~4%~~ 1% and 2% compared to a single MPI process execution ~~(a 7%~~ on 1-512 Nvidia K80 GPUs and on 1-128 Nvidia Titan X (Maxwell) GPUs, respectively (representing a 1% and 4.5% deviation from a ~~single~~ non-MPI single GPU runtime). We achieve this optimal domain decomposition parallelisation by overlapping communication and computation using native CUDA streams. This CUDA feature enables asynchronous compute kernel execution. Similar implementation and parallel scaling results were recently ~~achieved~~ reported for hydro-mechanical couplings (Räss et al., 2019a, c). Discrepancies in the parallel efficiency among the three tested distributed memory machines mainly results from the various hardware type and age, as well as the from the interconnect specifications. The *Xstream* supercomputer features Nvidia Tesla K80 GPUs based on Kepler chip architecture launched in late 2014 as well as single-rail Mellanox FDR Infiniband interconnect. The *octopus* supercomputer features consumer electronics Nvidia Titan X GPUs based on the Maxwell chip architecture launched in mid 2015 as well as dual-rail Mellanox FDR Infiniband interconnect. The *volta* node features latest Nvidia Tesla V100 GPUs based on Volta chip architecture launched in mid 2018 and Nvlink technology as intra-node interconnect. More recent chip architectures reduce the relative computation time and may provide less room for hiding the MPI communication. Dual-rail interconnect doubles the inter-node throughput and thus reduces the communication time

among distinct compute nodes. Note that *Xstream* features 16 GPUs per node which may reduce the inter-node communication compared to *octopus* that features 4 GPUs per node.

7 Conclusions

We have developed In this study, we develop FastICE, an iterative solver to efficiently exploit that efficiently exploits the capabilities of modern hardware accelerators such as GPUs. We report achieve rapid execution times on single-GPUs single GPUs monitoring and optimising memory transfers. We achieved a achieve close-to-ideal parallel efficiency (93% 99% and 95.5%) on a weak scaling test up to 512 and 128 GPUs on heterogenous hardware by overlapping MPI communication and computations. We implemented the coupled thermomechanical PDEs using our iterative PT approach in a straight-forward way from the mathematical model. The technical advances and utilisation of GPU accelerators enabled us to investigate the thermomechanical coupling and to resolve the first-order physics governing the enable us to resolve thermomechanically coupled ice flow in 3-D on a at high spatial and temporal resolution.

We benchmarked the benchmark mechanical solver of the coupled model against a community standard FastICE against the community model Elmer/Ice in a set of experiments specifically designed to test the mechanical solver. We further investigated explicit and , focusing specifically on explicit as opposed to implicit coupling and time integration strategies. We report find that the physical time step must be chosen with care. Sufficiently high temporal resolution is mandatory necessary in order to accurately resolve the coupled physics. Although minor differences arise among uncoupled and coupled approaches, we observe less localisation for uncoupled models compared to the fully coupled ones.

We established that In additional to high temporal resolution, a relatively high spatial numerical resolution is necessary to resolve the non-linear and spontaneous localisation of thermomechanically coupled ice flow, including of more than 100 grid-points in the vertical direction . We stress that spatial variations in the horizontal plane can significantly impact on the ice flow dynamic, justifying high spatial numerical resolution in all directions. We finally reported that considering the full 3-D stress tensor can significantly slow down the process of thermal runaway, which can ultimately be hindered by considering phase transitions.

GPUs are compact, affordable and relatively programmable devices that offer high performance throughput (close to TB/s peak memory throughput) and a good price to performance ratio. GPUs offer an attractive alternative to conventional CPUs owing to their massively parallel architecture featuring thousands of cores. is necessary to resolve thermomechanical localisation for typical ice-sheet thicknesses on the order of hundreds of meters. The presented models lever this modern technology and enable us to gain further process-based understanding of ice-flow localisation. Resolving the coupled processes at very high spatial and temporal resolutions provides future avenues to address current challenges in accurately predicting ice sheet dynamics.

Code availability. The FastICE software developed in this study is licensed under GPLv3 free software license. The latest version of the code is available for download from Bitbucket at https://bitbucket.org/lraess/fastice/ and from http://wp.unil.ch/geocomputing/software/. Past and future FastICE versions are available from a permanent DOI repository (Zenodo) at https://doi.org/10.5281/zenodo.3461171. The FastICE software includes code examples based on the PT method in both the MATLAB and CUDA C programming languages. The GPU routines run on a CUDA-capable GPU device. The multi-GPU version of the 3-D code requires CUDA-aware MPI to be installed. On the *octopus* GPU supercomputer, we have CUDA 10.0 installed and built Open MPI 2.1.5 with CUDA 10.0, GCC 6.5 on a CentOS 6.9 system.

Author contributions. LR participated in the early model and numerical method development stages, implemented the MPI version of the code, performed the scaling analysis, and reshaped the final version of the manuscript. AL realised the first version of the study, performed the benchmarks, and drafted the manuscript outline as the second chapter of his PhD thesis. FH and YP supervised the early stages of the study. JS contributed to the capped thermal model and provided feedback on the manuscript in the final stage. All authors have reviewed and approved off the final version of the manuscript.

Competing interests. The authors declare that they have no conflicts of interest.

Acknowledgements. We thank Dr. Samuel Omlin, Dr. Thibault Duretz and Mathieu Gravey for their technical and scientific support. We thank Dr. Thomas Zwinger ~~for his~~ and two anonymous reviewers for valuable comments, which enhanced the study. We acknowledge the Swiss Geocomputing Centre for computing resources on the octopus supercomputer and are grateful to Philippe Logean for continuous technical support. LR acknowledges support from the Swiss National Science Foundation's Early Postdoc Mobility Fellowship 178075. This research was supported by the National Science Foundation through the Office of Polar Programs awards PLR-1744758 and PLR-1739027. This material is based upon work supported by, or in part by, the U. S. Army Research Laboratory and the U. S. Army Research Office under contract/grant number W911NF-12-R0012-04. This work used the XStream computational resource, supported by the National Science Foundation Major Research Instrumentation program (ACI-1429830).

References

Bassis, J.: Hamilton-type principles applied to ice-sheet dynamics: new approximations for large-scale ice sheet flow, Journal of Glaciology, (56)97, 497–513, 2010.

Brædstrup, C., Damsgaard, A., and D.L., E.: Ice-sheet modelling accelerated by graphics cards, Computers and Geosciences, 72, 210–220, 2014.

Brinkerhoff, D. J. and Johnson, J. V.: Data assimilation and prognostic whole ice sheet modelling with the variationally derived, higher order, open source, and fully parallel ice sheet model VarGlaS, The Cryosphere, 7, 1161–1184, 2013.

Brinkerhoff, D. J. and Johnson, J. V.: Dynamics of thermally induced ice streams simulated with a higher-order flow model, Journal of Geophysical Research: Earth Surface, 120, 1743–1770, 2015.

Bueler, E. and Brown, J.: Shallow shelf approximation as a "sliding law" in a thermomechanically coupled ice sheet model, Journal of Geophysical research, 114, F03 008, 2009.

Bueler, E., Brown, J., and Lingle, C.: Exact solutions to the thermomechanically coupled shallow-ice approximation: effective tools for verification, Journal of Glaciology, 53(182), 499–516, 2007.

Chorin, A. J.: The numerical solution of the Navier-Stokes equations for an incompressible fluid, Bulletin of the American Mathematical Society, 73, 928–931, 1967.

Chorin, A. J.: Numerical solution of the Navier-Stokes equations, Mathematics of computation, 22, 745–762, 1968.

Clarke, G. K. C., Nitsan, U., and Paterson, W. S. B.: Strain heating and creep instability in glaciers and ice sheets, Reviews of geophysics and space physics, 15, 235–247, 1977.

Cook, S.: CUDA Programming, Morgan Kaufmann, Elsevier, 2012.

Crank, J. and Nicolson, P.: A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type, Mathematical Proceedings of the Cambridge Philosophical Society, 43, https://doi.org/10.1017/S0305004100023197, 1947.

Cundall, P., Coetzee, M., Hart, R., and Varona, P.: FLAC users manual, Itasca Consulting Group, pp. 23–26, 1993.

Duretz, T., Räss, L., Podladchikov, Y., and Schmalholz, S.: Resolving thermomechanical coupling in two and three dimensions: spontaneous strain localization owing to shear heating, Geophysical Journal International, 216, 365–379, 2019.

Egholm, D., M.F., K., Clark, C., and Lesemann, J.: Modeling the flow of glaciers in steep terrains: The integrated second-order shallow ice approximation (iSOSIA), Journal of Geophysical Research: Earth Surface, 116, F02 012, 2011.

Frankel, S. P.: Convergence rates of iterative treatments of partial differential equations, Mathematical Tables and Other Aids to Computation, 4(30), 65–75, 1950.

Gagliardini, O. and Zwinger, T.: The ISMIP-HOM benchmark experiments performed using the finite-element code Elmer, The Cryosphere, 2, 67–76, 2008.

Gagliardini, O., Zwinger, T., Gillet-Chaulet, F., Durand, G., Favier, L., de Fleurian, B., Greve, R., Malinen, M., Martín, C., Råback, P., Ruokolainen, J., Sacchettini, M., Schäfer, M., Seddik, H., and Thies, J.: Capabilities and performance of Elmer/Ice, a new-generation ice sheet model, Geoscientific Model Development, 6, 1299–1318, 2013.

Gerya, T.: Introduction to Numerical Geodynamic Modelling, Cambridge University Press, Cambridge, United Kingdom, 2009.

Gerya, T. V. and Yuen, D. A.: Characteristics-based marker-in-cell method with conservative finite-differences schemes for modeling geological flows with strongly variable transport properties, Physics of the Earth and Planetary Interiors, 140(4), 293–318, 2003.

Gilbert, A., Gagliardini, O., Vincent, C., and Wagnon, P.: A 3-D thermal regime model suitable for cold accumulation zones of polythermal mountain glaciers, Journal of Geophysical Research:Earth Surface, 119, 876–1893, 2014.

Glen, J. W.: The flow law of ice from measurements in glacier tunnels, laboratory experiments and the Jungfraufirn borehole experiment, Journal of Glaciology, 2, 111–114, 1952.

Goldberg, D.: A variationally-derived, depth-integrated approximation to the Blatter Pattyn balance, Journal of Glaciology, 57(201), 157–170, 2011.

Gong, Y., Zwinger, T., Åström, J., Altena, B., Schellenberger, T., Gladstone, R., and Moore, J. C.: Simulating the roles of crevasse routing of surface water and basal friction on the surge evolution of Basin 3, Austfonna ice cap, The Cryosphere, 12, 1563–1577, 2018.

Harlow, F. H. and Welch, E.: Numerical calculation of time-dependent viscous flow of fluid with free surface, Physics of Fluids, 8(12), 2182–2189, 1965.

Hindmarsh, R. C. A.: Stress gradient damping of thermoviscous ice flow instabilities, Journal of Geophysical Research:Earth Surface, 111, B12 409, 2006.

Hindmarsh, R. C. A.: Consistent generation of ice-streams via thermo-viscous instabilities modulated by membrane stresses, Geophysical research letters, 36, L06 502, 2009.

Hutter, K.: Theoretical glaciology: material science of ice and the mechanics of glaciers and ice sheets, vol. 1, Springer, 1983.

Huybrechts, P. and Payne, T.: The EISMINT benchmarks for testing ice-sheet models, Annals of Glaciology, 23, 1–12, 1996.

Isaac, T., Stadler, G., and Ghattas, O.: Solution of Nonlinear Stokes Equations Discretized by High-order Finite Elements on Nonconforming and Anisotropic Meshes, with Application to Ice Sheet Dynamics, SIAM Journal on Scientific Computing, pp. B804–B833, 2015.

Jarosch, A.: Icetools: a full Stokes finite element model for glaciers, Computers and Geosciences, 34, 1005–1014, 2008.

Jouvet, G., Picasso, M., Rappaz, J., and Blatter, H.: A new algorithm to simulate the dynamics of a glacier: theory and applications, Journal of Glaciology, 54, 801–811, 2008.

Kelley, C. T. and Keyes, D. E.: Convergence Analysis of Pseudo-Transient Continuation, SIAM Journal on Numerical Analysis, 35(2), 508–523, 1998.

Kelley, C. T. and Liao, L.-Z.: Explicit pseudo-transient continuation, Pacific Journal of Optimization, 9(1), 77–91, 2013.

Kiss, D., Podladchikov, Y., Duretz, T., and Schmalholz, S. M.: Spontaneous generation of ductile shear zones by thermal softening: Localization criterion, 1D to 3D modelling and application to the lithosphere, Earth and Planetary Science Letters, 519, 284–296, https://doi.org/10.1016/j.epsl.2019.05.026, 2019.

Larour, E., Seroussi, H., Morlighem, M., and Rignot, E.: Continental scale, high order, high spatial resolution, ice sheet modeling using the Ice Sheet System Model (ISSM), Journal of Geophysical research, 117, 1–20, 2012.

Leng, W., Ju, L., Gunzburger, M., and Ringler, T.: A parallel high- order accurate finite element nonlinear Stokes ice sheet model and benchmark experiments, Journal of Geophysical research, 117, F01 001, 2012.

Leng, W., Ju, L., Gunzburger, M., and Price, S.: A Parallel Computational Model for Three-Dimensional, Thermo-Mechanical Stokes Flow Simulations of Glaciers and Ice Sheets, Computer Physics Communications, 16(4), 1056–1080, 2014.

McKee, S., Tomé, M., Ferreira, V., Cuminato, J., Castelo, A., Sousa, F., and Mangiavacchi, N.: The MAC method, Computers & Fluids, 37, 907–930, https://doi.org/10.1016/j.compfluid.2007.10.006, 2008.

Morland, L.: Thermomechanical balances of ice sheet flows, Geophysical and Astrophysical Fluid Dynamics, 29, 237–266, 1984.

Nye, J. F.: The flow law of ice from measurements in glacier tunnels, laboratory experiments and the Jungfraufirn borehole experiment, Proceedings of Royal Society A, 219, 477–489, 1953.

Ogawa, M., Schubert, G., and Zebib, A.: Numerical simulations of three-dimensional thermal convection in a fluid with strongly temperature dependent viscosity, Journal of Fluid Mechanics, 233, 299–328, 1991.

Omlin, S.: Development of massively parallel near peak performance solvers for three-dimensional geodynamic modelling, Ph.D thesis, University of Lausanne, 2017.

Patankar, S.: Numerical Heat Transfer and Fluid Flow, Comput. Methods Mech. Thermal Sci. Ser., CRC Press, Boca Raton,Fla, 1980.

Pattyn, F., Perichon, L., Aschwanden, A., Breuer, B., de Smedt, B., Gagliardini, O., Gudmundsson, G. H., Hindmarsh, R. C. A., Hubbard, A., Johnson, J. V., Kleiner, T., Konovalov, Y., Martin, C., Payne, A. J., Pollard, D., Price, S., Rückamp, M., Saito, F., Soucek, O., Sugiyama, S., , and Zwinger, T.: Benchmark experiments for higher-order and full-Stokes ice sheet models (ISMIP- HOM), The Cryosphere, 2, 95–108, 2008.

Payne, T. and Baldwin, D.: Analysis of ice-flow instabilities identified in the EISMINT intercomparison exercise, Annals of Glaciology, 30, 204–210, 2000.

Payne, T., Huybrechts, P., Abe-Ouchi, A., Calov, R., Fastook, J., Greve, R., Marshall, S., Marsiat, I., Ritz, C., Tarasov, L., and Thomassen, M.: Results from the EISMINT model intercomparison: the effects of thermomechanical coupling, Journal of Glaciology, 46(153), 227–238, 2000.

Perego, M., Gunzburger, M., and Burkardt, J.: Parallel finite element implementation for higher order ice-sheet models, Journal of Glaciology, 58(207), 76–88, 2012.

Poliakov, A. N. B., Cundall, P. A., Podladchikov, Y. Y., and Lyakhovsky, V. A.: An explicit inertial method for the simulation of viscoelastic flow: An evaluation of elastic effects on diapiric flow in two- and three-layers models, Flow and Creep in the Solar Systems: Observations, Modeling and Theory, pp. 175–195, 1993.

Pollard, D. and DeConto, R. M.: Description of a hybrid ice sheet-shelf model, and application to Antarctica, Geoscientific Model Development, 5, 1273–1295, 2012.

Räss, L., Simon, N., and Podladchikov, Y.: Spontaneous formation of fluid escape pipes from subsurface reservoirs, Scientific Reports, 8, 2018.

Räss, L., Duretz, T., and Podladchikov, Y. Y.: Resolving hydro-mechanical coupling in two and three dimensions: Spontaneous channelling of porous fluids owing to decompaction weakening, Geophysical Journal International, https://doi.org/10.1093/gji/ggz239, 2019a.

Räss, L., Licul, A., Herman, F., Podladchikov, Y., and Suckale, J.: FastICE, https://doi.org/10.5281/zenodo.3461171, 2019b.

Räss, L., Omlin, S., and Podladchikov, Y. Y.: Resolving Spontaneous Nonlinear Multi-Physics Flow Localization in 3-D: Tackling Hardware Limit, https://developer.nvidia.com/gtc/2019/video/S9368, GTC Silicon Valley - 2019, 2019c.

Robin, G. d. Q.: Ice movement and temperature distribution in glaciers and ice sheets, Journal of Glaciology, 2, 523–532, 1955.

Saito, F., Abe-Ouchi, A., and Blatter, H.: European Ice Sheet Modelling Initiative (EISMINT) model intercomparison experiments with first-order mechanics, Journal of Geophysical Research, 111, F02 012, 2006.

Schäfer, M., Gillet-Chaulet, F., Gladstone, R., Pettersson, R., Pohjola, V., Strozzi, T., and Zwinger, T.: Assessment of heat sources on the control of fast flow of Vestfonna ice cap, Svalbard, The Cryosphere, 8, 1951–1973, 2014.

Schoof, C. and Hindmarsh, R.: Thin film flows with wall slip: an asymptotic analysis of higher order glacier flow models, The Quarterly Journal of Mechanics and Applied Mathematics, 63(1), 73–114, 2010.

Shin, D. and Strikwerda, J. C.: Inf-Sup conditions for finite-difference approximations of the Stokes equations, Journal of the Australian Mathematical Society. Series B. Applied Mathematics, 39(01), 121–134, 1997.

Solomon, S., Qin, D., Manning, M., Chen, Z., Marquis, M., Averyt, K., Tignor, M., and Miller, H.: The physical science basis p.235–337, IPCC report AR4 , New York and Cambridge: Cambridge University Press, 2007.

Suckale, J., Platt, J., Perol, T., and Rice, J.: Deformation-induced melting in the margins of the West Antarctic ice streams, Journal of Geophysical Research:Earth Surface, 119, 1004–1025, 2014.

Tezaur, I. K., Perego, M., Salinger, A. G., Tuminaro, R. S., and Price, S. F.: Albany/FELIX: a parallel, scalable and robust, finite element, first-order Stokes approximation ice sheet solver built for advanced analysis, Geoscientific Model Development, 8, 1197–1220, 2015.

Virieux, J.: P-SV wave propagation in heterogeneous media: Velocity-stress finite-difference method, GEOPHYSICS, 51, 889–901, https://doi.org/10.1190/1.1442147, 1986.

Watkins, J., Tezaur, I., and Demeshko, I.: A study on the performance portability of the finite element assembly process within the Albany Land Ice solver, Elsevier, 2019.

Weinan, E. and Liu, J.-G.: Projection method I: convergence and numerical boundary layers, SIAM journal on numerical analysis, pp. 1017–1057, 1995.

Zhang, T., Ju, L., Leng, W., Price, S., and Gunzburger, M.: Thermomechanically coupled modelling for land-terminating glaciers: a comparison of two-dimensional, first-order and three-dimensional, full-Stokes approaches, Journal of Glaciology, 61(228), 702–711, 2015.

Zwinger, T., Greve, R., Gagliardini, O., Shiraiwa, T., and Lyly, M.: A full Stokes-flow thermo-mechanical model for firn and ice applied to the Gorshkov crater glacier, Kamchatka, Annals of Glaciology, 45, 29–37, 2007.