We thank the reviewer for their constructive criticism of our manuscript, and would hereby like to respond to their concerns. Their comments are shown in italics, our response in regular type.

Main points of critics

> *What I do not really understand, is, why a new model variant has to be presented in two different papers? In my opinion – e.g., by reducing the details on presenting Stokes approximations that can be looked up in standard literature (e.g., Greve and Blatter, 2009) and numerical concepts in the Appendix – it would be more convenient for the reader to access all information (whatever is planned for part 2) from within a single manuscript, in particular as part 2 to me seems to contain information that appears to me as essential to understand statements on performance and scalability (or the lack of the latter).*

We think reviewer #2 summarised the reason for writing out the different Stokes approximations nicely: "…the authors put a lot of effort in writing out equations and discretizations. This is truly valuable, as the implementation on an unstructured triangular mesh adds quite some complexity." Indeed, while the equations for the Stokes approximations themselves can be found in existing literature, the discretisations of these equations as they have been implemented in UFEMISM cannot. We find it convenient to have these gathered in one place, in a consistent form which also matches the model code. The reason that we have decided to split the presentation of our new model into two parts, is that a single manuscript would become unreadably long. The 'nudging' approaches that we implemented in UFEMISM v2.0 are based on, but not identical to, earlier methods, so that they too require a comprehensive presentation of the underlying equations. While the manuscript of Part II has unfortunately been delayed (as two of the authors have had to switch contracts in the meantime, a consequence of the failing system of scientific funding in the Western world), even the current draft is already long enough to merit a separate publication.

> *The authors seem to have Message Passing paradigm (MPI) introduced into UFEMISM v2. I can find little information on how exactly this has been achieved. From the sentence "… data is distributed over many memory chips" I assume that they are applying something like a domain-decomposition to distribute the mesh over different tasks.*

Correct, v2.0 uses a domain decomposition. We will state this explicitly in the manuscript, and we will explain that v1.0 also used MPI to do parallel computations, but that in that version, only the *computations* were distributed over the processing cores (with the data gathered on a memory node that can be accessed by all cores), while in v2.0 the *data* is distributed too (thus necessitating the use of MPI communication at a lot more places in the code).

> *I understand that the solution step (should one apply approximations that demand solution of a matrix system) is taken over by PETSc (which comes with an MPI interface), but, for instance, in SIA the algorithm will have to evaluate hydrostatic pressure*

gradients across domain boundaries – how is this achieved? What kind of MPI communication (blocking, non-blocking) has been implemented for exchanging data?

The calculation of gradients (such as the gradient of the surface elevation that is needed to calculate ice velocities in the SIA) is done using (sparse) matrix multiplications. These are handled by PETSc, which we believe uses non-blocking MPI communication internally unless specified otherwise by the user. We will mention this in the manuscript.

What also confuses me, is the quote "UFEMISM offers a set of standardised routines that interface with the OpenMPI library (Gabriel et al., 2004) to facilitate this". MPI is a standard and there are several implementations on library-level, such as OpenMPI, MPICH or MVAPICH and also vendor specific MPI-libraries. What part in the code makes it necessary to particular utilize OpenMPI? This is in my opinion not unimportant, in particular in view of the initial claim in the abstract, that UFEMISM should be ready to be integrated into earth system models (ESM). ESM's generally (by the high demand from their atmospheric and ocean components) are run on dedicated supercomputers that in many cases restrict the usage of anything else than a vendor MPI (implied by the interconnection network in place) that not necessarily is derived from OpenMPI. I would ask to add more information on how exactly MPI has been implemented for the different approximations, how PETSc is integrated on MPI level and why there is a suggested restriction to a single MPI implementation (OpenMPI)?

We have clearly not explained ourselves well, for which we apologise. The UFEMISM code by itself simply calls routines from the MPI API. Whether a user has installed OpenMPI, MPICH, or another standard-compliant library should not matter (in the Nix set-up we provide, we use OpenMPI because that's what we use ourselves). The set of standardised routines we mentioned, is a collection of subroutines we wrote ourselves that provide some more convenience for aspiring developers. For example, gathering distributed data to a single processing core involves allocating the appropriate amount of memory (possibly after a reduction operation to determine how much memory is required), calling one of the MPI_gather routines, and (optionally) deallocating the memory of the distributed array. Our set of routines combine these operations to make a developer's life that much easier. We will explain this more clearly in the manuscript.

First, what model is run (1st order, hybrid SIA/SSA or DIVA) for the reported scalability test?

We ran the scalability tests with the spin-up phase of the (modified, plan-view) MISMIP experiment, using the DIVA with an 8-km resolution at the grounding line, for a period of 10,000 years. These simulations were run on the Snellius supercomputer on the AMD Rome 7H12 nodes (of 128 cores each). We will mention this in the manuscript.

Similar issue with the claim to run millennial Antarctic "basin-scale" simulations on a laptop – what approximation was used there and what size of problem are we looking at?

These are pan-Antarctic simulations of 20,000 model years that can run in less than 24 wall clock hours on 2 cores of a Macbook Pro M2 2023 using the GNU Fortran compiler version 13.2.0. They use the DIVA at a relatively low resolution between 16 and 50 km (e.g. grounding lines and East Antarctic interior, respectively), except over target basins (e.g. Pine Island and Thwaites areas) where the mesh resolution smoothly increases up to 3-5 km around the grounding line. We will mention this in the manuscript.

Secondly, what computational platform are the tests from Fig. 2 ran on? Judging by the amount of cores, to me it appears to be a single node of a computing-cluster. From my own experience, often there are situations with memory-bound codes (and generally finite volume, finite difference, finite element fall under that category) on, e.g., AMD-EPYC systems (with versions that exactly have 2 sockets a 64 = 128 cores per node), where performance on the single node drops first at 8 cores (due to L3 cache misses) and then at 32 cores because of insufficient or insufficiently used memory bandwidth (the architecture has 4 NUMA domains) – something I also can observe in Fig. 2., except that intra-node scaling completely breaks down past 32 cores, which raises my doubts that inter-node scaling can be achieved at all. Also, intra-node performance highly depends on the implementation (e.g., Byckling et al., 2017). A single non-performing serial section in an OpenMP threaded application has the potential to destroy scalability (Amdahl, 1960).

We agree that these results require some more information and discussion. Apart from the possible issues with PETSc, we add some discussion that strong scaling here also seems to be dependent on the architecture and communication latencies between cores. We will also mention that it is possible the experiment used in this test was too 'small', i.e with too few vertices, so that the communication time at large umbers of cores starts to dominate over the computation time. The reviewer correctly guessed the system the tests where run on (amd-epyc on snellius). We will mention this in the manuscript.

A further problem (should Fig 2 demonstrate runs confined to a single node) I have to point out that one cannot deduce inter-nodal from intra-nodal scalability – to answer this, I would ask the authors to provide scalability results run over several distributed memory nodes.

The code was reimplemented (from UFEMISM v1.0) with fully distributed memory MPI. This means no shared memory accesses are used as explained in section 3.1. Therefore the communication paradigm used is the same for intra and inter-nodal communication. However, the reviewer is correct to point out that the scaling will be different because for example intra-node communication is much slower than communication between two cpus on one node. We will mention this in the manuscript.

Third and final point of critics is that scaling of a dimensionally reduced flow-line problem as MISMIP in my opinion is not representative of a full Antarctic setup, assuming that applications like that are the final goal to achieve scaling with. The authors must have performed MISMIP+ spinup (ice0r) to get a starting point for the reported melt experiment 3 (ice1r) – why not reporting performance/scaling numbers from that setup?

This is the same 'modified' MISMIP experiment we describe later on, which extrudes the flowline set-up from the original paper radially to create a circular, cone-shaped island. A plan-view experiment, not a flowline. We will mention this in the manuscript.

To summarize, I would see the necessity of elaborating the circumstances (platform, compiler, compiler-optimization flags, and most important applied approximation to Stokes) that lead to those scalability results and (provided we are looking at intra-nodal scaling here) extend to inter-nodal scalability tests to be in a position to evaluate code scalability for distributed memory applications, if possible for applications that solve (parts of) whole ice-sheets rather than flow-line setups. Furthermore, I ask to provide wall-clock times concerning the performance, in particular of the most versatile approximation (1st order/Blatter-Pattyn) – which by ISMIP-HOM results to me appears to be the only option if sufficient accuracy is sought - on large scale applications, such as Antarctica or MISMIP+.

Following our earlier responses, we will add the requested information about our scaling experiments.

We have not done any dedicated performance tests with the Blatter-Pattyn Approximation. Preliminary tests, as well, as the ISMIP-HOM experiments, indicate that the BPA is easily up to 50 times slower than the DIVA, making it impractical to use in realistic applications – at least, until the scalability problems are sorted out.

To me it appears that for high-frequency disturbances in ISMIP-HOM Exp. A, the SIA/SSA as well as the DIVA approximation are significantly deviating from the ensemble (both the HO and even more the Stokes) and in case of the bedrock friction experiment (ISMIP-HOM C) somewhat (to me surprisingly) in the lower row of Fig. 4 the hydrostatic first order (Blatter-Pattyn) solution – despite the authors claiming that it is contained in the ensemble-range for all domain scales. In particular, with respect to conclusions of inaccuracies arising in both, the SIA/SSA and DIVA approximation at smaller disturbance length scale and to me also the Blatter-Pattyn solution showing deviations in Exp. C, I would ask to provide a discussion on the expected accuracy and the acclaimed verification of these approximation applied to ice-sheet simulation, also beyond synthetic intercomparison setups.

We will state more clearly in the manuscript that UFEMISM's solutions to the BPA lie outside the ISMIP-HOM model ensemble for some experiments. While we do not have a clear explanation for this deviation, we do think that the deviations are quite small (relative to the ensemble range); had UFEMISM v2.0 been included in the Pattyn 2008 intercomparison exercise, we do not think it would have been excluded from the ensemble.

Regarding the accuracy of the SIA/SSA and DIVA solutions, we do not think it is within the scope of this manuscript to discuss the relative merits of these different physical approximations. We aim to focus this manuscript on *verifying* our solution of the equations, rather than *validating* the equations themselves.

> *For the marine ice-sheet examples (MISMIP and MISMIP+) I could not find the information what approximation to the Stokes equation has been used to compute the results.*

All these experiments were performed with the DIVA. We will mention this in the manuscript.

> *I am confused by the output in Figure 5 (MISMIP), where a timescale of 30 kyr = 3.0 × 10^2 yr is depicted. If this should resemble Exp 2 in Pattyn et al. (2012), I am missing several further step-wise increases.*

We only did the single step-wise increase/decrease, as this is enough to assess the (unwanted) grounding-line hysteresis (or "numerical path- dependency", as the other reviewer would like it to be called) present in the model. We will mention this in the manuscript.

> *Also, the (here only two varying) values of the factor A, to me appear several orders of magnitude larger than those used in the MISMIP experiments.*

The values given by Pattyn et al. (2012) are in Pa^-3 s^-1, while ours are in Pa^-3 yr^-1.

> *Or - in view of the claimed code verification - run the MISMIP Exp 1 and Exp 2 according to the protocol, such that the reader can get a clear picture on how UFEMISM v2.0 behaves in view of the MISMIP ensemble.*

The reason we have opted to extrude the 1-D geometry radially, rather than transforming the original 1-D flowline into a 2-D flowband, is that, while this means the resulting grounding-line position no longer matches the (semi-)analytical solution provided by Pattyn et al. (2012), it offers the advantage of checking the full 2-D stress balance (instead of only the x-component). This particularly allows us to check the symmetry of the grounding line. A well-known (but, as far as we are aware, never published) issue with flux condition schemes in square-grid models is the "octagonal" grounding line. A similar undesirable dependency on the grid geometry could sometimes be seen in UFEMISM v1.0 (which used a flux condition scheme), but has since been fixed with the introduction of the sub-grid friction scaling scheme in v2.0. We will mention this in the manuscript.

All the investigated intercomparison setups focus on pure mechanics. Yet, a changing temperature field by the Arrhenius law has a significant impact on ice-dynamics (Schoof and Hewit, 2021). The manuscript is not mentioning the inclusion or even computation of heat transfer in a coupled thermo-mechanical context at all in the text. I would ask the authors to add a paragraph if and how temperature (or even damage) is accounted for or included in UFEMISM?

The thermodynamics module of UFEMISM v2.0 is unchanged from the version in v1.0, which solves the heat equation inside the ice (excluding horizontal diffusion and possible liquid water inside the ice column). Ice damage is currently not accounted for in any way. We will mention this in the manuscript.

Detailed list of requested changes or elaborations

What exactly do the authors mean by NetCDF standard? A certain convention, like CF?

We mean that the model only produces NetCDF-4 output files (whereas v1.0 produced NetCDF-3, as well as some .txt files), and only reads NetCDF input files (whereas v1.0 required a number of .txt input files). We will clarify this in the manuscript.

What constitutes the readiness for inclusion in ESM's? Does the code have coupler interfaces for online coupling to atmospheric or ocean models (e.g., Gladstone et al., 2021) implemented? From the scalability figures given in the text, I see a problem to run the code on large supercomputers, which I see as a necessity for inclusion in ESM's. The in my opinion unclear restriction with respect to MPI implementation (OpenMPI) most likely constitute a hurdle to run on Tier 0 or Tier 1 HPC facilities. From what is presented in this paper, I would not derive a readiness of UFEMISM v2.0 to be incorporated into ESM's. See my argumentation under major points #1 and #2.

While we acknowledge that the computational performance of UFEMISM still needs some attention, we do not think this is an issue for coupling to ESM's. In that context, the computational load of an ice-sheet model is typically negligible compared to the atmospheric and oceanic components.

As for the coupling itself, we were thinking more of a script-based coupling than of integrating UFEMISM with the code of a GCM into a single executable. The current code already goes quite some way to making the latter possible, including clean wrapper routines for running a single, externally defined coupling time-step, and a separate library for the various remapping routines to reproject data between an ESM's (supposedly) global lon/lat-grid and UFEMISM's adaptive mesh. However, in our experience a script-based coupling is typically much easier to realise, especially for exploratory projects. To make this kind of coupling easier, UFEMISM outputs data on a (user-defined) square grid, is flexible in accepting input files from different locations and grid types, and allows for "perfect restarting" (e.g. saving the auxiliary fields that are used to

calculate the dynamic time step) so that the repeated terminations and (re)initialisations do not affect the model results.

We will mention this in the manuscript.

> *What about calving-induced instabilities, like MICI (e.g. Crawford et al., 2021) - or are those subsumed under ice-dynamical processes?*

We will add calving to this list of possible physical sources of uncertainty.

> *May I point out that in the context of calving-front computations in Greenland, Todd et al. (2018) introduced a dynamic remeshing algorithm into Elmer/Ice. Additionally, in the first flowline marine ice-sheet full-Stokes experiments by Durand et al. (2009) adaptive meshing around the grounding line was already introduced 15 years ago in connection to the also in this manuscript discussed MISMIP setups. Furthermore, also ISSM includes mesh-adaptation, according to their web-site (https://issm.jpl.nasa.gov/documentation/mesh/).*

We will add references to Todd et al. (2018), Durand et al. (2009), Gladstone et al. (2010), and to dos Santos et al. (2019, for ISSM) to the manuscript, and change the phrasing accordingly.

> *For flow-line problems, mesh sensitivity of grounding line positions – even in the context of full-Stokes - was described in even earlier works by Durand et al. (2009)*

We will add this reference to this manuscript.

> *"Most complete" in what sense? And I would add Blatter (1995) as a citation here.*

In the sense that it neglects the fewest terms from the Stokes equations; we will mention this in the manuscript, and include the reference to Blatter (1995).

> *In my opinion, if the applied approximations to the Stokes equations are discussed in such details, it would be best to write out the complete Stokes equations to relate the approximations. But in my opinion – as I also mentioned in the General Impression - a reference to standard literature (e.g. Greve & Blatter, 2009) or presentation of the equations from section 2.2 in an appendix would be sufficient and significantly shorten the article.*

We will refer the reader to Greve and Blatter (2009) for a comprehensive description of the Stokes equations and the derivation of the different approximations.

> *The wording "generally very close" in my view is hard to interpret and easy to misinterpret. If taken by its spatial extend of validity on large ice-sheets, one could claim that also SIA is "generally" very close to Stokes, but it completely fails under ice-domes*

and ridges and in fast flow regions and shelves. The ISMIP-HOM reference (Pattyn et al., 2008) is a set of idealized synthetic benchmark cases and in my opinion does not justify a statement that could be interpreted that first-order approximation is a sufficient substitute to the complete Stokes solution in every situation – which does not apply, in particular where variations in vertical advection are of essence, like in thermo-mechanically coupled problems of ice-streams (Schoof and Mantelli, 2021), advection problems of tracers (Jouvet et al., 2021) and flow at ridges and domes (Seddik et al., 2011).

We will clarify that the BPA, and to a lesser extent the DIVA and the hybrid SIA/SSA, are generally able to describe the large-scale evolution of continental ice-sheet-shelf systems such as the Antarctic ice sheet, with the caveats mentioned by the reviewer.

To me it appears that this could be interpreted that BPA solves for all vertical variations of strain rates. My suggestion: … where those approximations parametrise or simply ignore the in BPA not neglected vertical derivatives of the horizontal velocity components.

We will clarify that this only applies to the vertical derivatives of the horizontal velocities and strain rates.

I would understand SIA to be solving column wise quadrature on a three-dimensional mesh (e.g., Greve and Blatter, 2009), so not being confined to a plane mesh.

In UFEMISM, as in (as far as we are aware) all models that offer an SIA-only option, the analytical solution to the SIA in the vertical column is used, so that it only needs to *evaluate* an expression, rather than *solve* an equation. However, the hybrid SIA/SSA, the SSA part of the solution still needs to be solved in the 2-D plane.

This links to major points of critics #4 that the authors seem to completely neglect the thermo-mechanical aspect of ice-sheet modelling, which in my opinion is of essence (Schoof and Hewit, 2013). In my opinion, one at least should mention that the rate factor, A(T,p) , is a function of the temperature and the pressure (and damage, if one wants to extend to that).

We will refer to the UFEMISM v1.0 model description paper, where the equations for the heat equation and the thermomechanical coupling are provided.

To me this sentence is a contradiction: either there is a zero stress boundary or there is friction with a resulting tangential stress applied. Suggestion: A similar dynamical boundary condition …

Accepted.

This links to my major point #3. As DIVA is introduced to be the default solver in UFEMISM, does that mean in a complementary conclusion that DIVA should not be deployed to mesh sizes below this threshold, as then accuracy is compromised? I would like to see some sort of deeper discussion in the with respect to the rest of the manuscript extremely brief section 5.

Technically, it is not the higher mesh resolution that invalidates the DIVA (or the SIA/SSA, or the BPA), but rather the appearance of smaller topographical features that can be resolved by a finer mesh. That said, we do not wish to move too far into the ice-dynamical territory where discussions about the validity and applicability of different Stokes approximations belong. While papers about these discussions are highly valuable, we think our manuscript is already long enough when limited to describing only a computer program that *solves* these approximations. Given this deliberate choice of scope, we believe the existing paragraph in Sect. 5 ("The ISMIP-HOM experiments presented here … thickness should not change.") adequately covers the subject.

To me that does not come clear from (8) (i.e., there is no exponent n=2 over the viscosity). Also, for consistency, the product $\beta \cdot F2$ in (7) should be dimensionless, which to me does not work out if $F2$ is proportional to the square of the inverse viscosity.

The description of the equation is incorrect, it should simply be "a (scaled) depth-integral of the inverse viscosity". We will change this in the manuscript. We have also checked the units and verified that the product beta*F_n is indeed dimensionless.

The reader might ask themselves where that would be. Are there other equations entering the system? In view of a more complex derivation of the equations of motion, I would suggest to drop everything around eqts. (7) and (8) and directly refer to look things up in Lipscomb et al. (2019).

We appreciate the suggestion, but we will keep the presentation of the physical equations in their current form.

I would like to have the assumption of no-slip motivated. I do not even understand it in case of hybrid SIA/SSA, as to my understanding there the sliding velocity should be provided by the SSA solution (hence non-zero).

In the hybrid SIA/SSA case (which is described in the next paragraph), the sliding velocities from the SSA are indeed added to the vertical shear velocities from the SIA. However, when UFEMISM is used in SIA-only mode (which is possible, but is not done in any of the experiments presented here), it is not possible to include a sliding law (meaning that we did not include the option in UFEMISM). We will clarify this in the manuscript.

This links to major point of critics #3. I would see it necessary to quantify this in terms of grid sizes that can be addressed with hybrid SIA/SSA.

See above for our reply regarding the scope of the manuscript.

> *That leaves me (and perhaps some readers) with the question on how water pressure is determined in UFEMISM v2.0? Is there a sub-glacial hydrology model included (e.g., Gagliardini and Werder, 2018) to provide that variable?*

There is not. Currently, the sub-glacial water pressure is defined as 96 % of the ice overburden pressure (following Winkelmann et al., 2011), optionally scaled with a bedrock elevation-dependent parameterization developed for Antarctica (following Martin et al., 2011), which we will mention in the manuscript. The inclusion of a sub-glacial hydrology model is high on the priority list for future updates.

> *May I point out that sub-grid friction parametrizations at grounding lines are wider spread in the ice-sheet model community. Also ISSM (Seroussi et al, 2014,) and Elmer/Ice (Gagliardini et al., 2016) deploy a sub-grid friction parametrization.*

We will add these references to the manuscript.

> *Equation (21) seems to be vertically integrated mass balance. Hence, $\boldsymbol{u}H$ being the vertically integrated, horizontally vector-valued, volume flux. Please, to inform the readers, add a definition of it to the text, also if/how the definition of this term differs between the available approximations to the Stokes equations.*

We will mention that **u** here is the vertically integrated, horizontally vector-valued ice velocity. The equations presented in this paragraph do not differ between the available approximations to the Stokes equations; since UFEMISM always assumes a uniform, constant ice density, only the vertically averaged ice velocity is needed in the continuity equation.

> *How does the predictor-corrector scheme link with the time-discretization schemes presented in section 2.5? I have the suspicion that the symbols Δt have different meanings in 2.5 and 2.6. Please, elaborate.*

Correct, the are some subtleties to the use of Δt. The different schemes in Sect. 2.5 require a value of Δt in order to yield H(t+Δt), while the P/C-scheme in Sect. 2.6 requires the rate of change dH/dt. Currently, the model solves this mismatch by taking H(t+Δt) (which is provided by whatever scheme, explicit, implicit, the user chooses), subtracting H(t) and dividing by the Δt that went into the scheme. This Δt is later on adapted by the P/C-scheme to yield the "final" Δt that is used by the model. We will mention this in the manuscript.

> *To me, this gives an over-simplified picture. Even in a shared-memory machine/node, generally, it is not a single chip containing the data. And, not all processing cores (which I would use as a term rather than processors) can access a certain chip in a similar fast way (NUMA domains). A better formulation in my view would be: … in shared memory*

architecture, where all parts of the memory are accessible via a common bus to all computing cores, in contrary to distributed memory architecture that demands communication between by memory separated computing nodes.

Accepted.

I would understand architecture as the hardware, which cannot directly be compared to a program (software implementation). For the latter, it very much depends on how the code is implemented. It is correct that on a pure hardware-level distributed memory access across nodes is slower (how much depends also on the performance of the interconnect-network and the memory-layout of the shared memory node) than the one of shared memory, yet, even shared memory parallelism (talking again about software) obeys Amdahl's law (Amdahl, 1965) and performance mainly hangs on the serial sequences of the code (which generally exist). Suggestion: Memory access within shared memory nodes outperforms message passing across shared-memory nodes.

Accepted.

To my understanding, not every approximation needs a matrix system to be solved, SIA does not. And the SSA/DIVA matrix must be significant smaller than the 1st-order system. I wonder: Does the number given apply to all discussed approximations? If not, I would ask to be more specific.

In SIA-only mode, this number would likely be a lot smaller. However, we do not foresee UFEMISM being used this way in any applications where computation time is a limiting factor (in fact, the only time we've ever used it is in the EISMINT-1 benchmark experiments described in the v1.0 paper). The 80% number is for the DIVA, which is (in our experience) the 'fastest' of all approximations except the SIA. For the hybrid SIA/SSA and the BPA, this number would be higher still. We will mention this in the manuscript.

This links to #1 of major issues. Why the constraint to the OpenMPI flavour? Does this mean UFEMISM cannot compile with another MPI-standard library, like IntelMPI or vendor specific MPI implementation? If so, please explain why.

We agree this phrasing was confusing; while we use OpenMPI on our system, UFEMISM should be able to compile and run with any other MPI-standard library. We will clarify this in the manuscript.

This also links to #1 of major issues. From the manuscript, I do not get enough information to be in a position to understand how the MPI parallelism in UFEMISM is organized. It would be interesting to the HPC inclined reader to learn how partitioning is done and - in particular with respect to the remeshing - the load balancing is guaranteed. To my knowledge, PETSc is well tuned to perform on multi-node clusters - what in particular do the authors suggest to be changed therein?

We do not think the problem lies with PETSc itself, but rather with the way we have implemented PETSc within UFEMISM. Mesh partitioning is currently done by simply partitioning the rectangular domain into equal-sized 'strips' along a single dimension. Optionally, the width of the resulting sub-domains can be tuned to make sure each process gets (approximately) equal numbers of vertices. We have done some (simple) load balancing experiments for version 1.0, which suggested that this approach works reasonably well. Where we suspect the current performance problem lies, is with how PETSc determines (or is told) the connectivity between the vertices, and by implication, the non-zero structure of the sparse matrices it must work with. In the current implementation, in every non-linear viscosity iteration (where the sparse matrix equation must be solved), PETSc is (re)initialized, the sparse matrices are constructed, the resulting matrix equation is solved, and PETSc is finalised. While our own time measurements show that it is the solving step that accounts for 99% of the computation time (of these combined steps), we still suspect that somehow storing the non-zero structure of the sparse matrices (which remains unchanged until the next mesh update) could help with performance. The reason we suspect this is because UFEMISM v1.0 was much faster in this regard, and the only significant change in this particular part of the code is the change from shared memory to distributed memory.
We will reflect these thoughts in the Discussion section of the manuscript.

> *This links to #2 of major issues. Firstly, I am missing the information what approximation applied allows one to run "basin-scale" (not sure what it means in terms of grid-sizes and spatial dimensions) on a laptop. Secondly, simulations on a laptop to me have a remote relevance to parallel performance/scaling on large machines, particular on distributed memory setups, which I understand this chapter to be about if the authors refer to "Large-scale practical applications".*

"Basin-scale" in this context means a single ice-sheet drainage basin. Apart from the ISMIP-HOM experiments, all simulations presented here are run with the DIVA. We will mention this in the manuscript.

Regarding the laptop vs. cluster question: we believe performance to be equally important in both settings. In our experience, a large fraction of a model developer's time is spent running relatively short, simple simulations on their laptop to debug and test new code. Only when that new code is judged mature enough do we move on to larger systems to run large-scale tests. However, already in the development phase, a significant fraction of our time is spent waiting for these simple test simulations to finish (which can take anywhere between a few seconds to a few hours, depending on the kind of work). Improving the model's performance, and thereby reducing this waiting time, would already be a big help to us. When we do eventually move on to 'application' simulations, about half of the time these are run on a laptop as well; only when the scale of the experiments (either in terms of resolution or duration, or in terms of number of simulations) becomes too large to be feasible run on a laptop do we move on to running on a cluster. Therefore, even for laptop settings, performance is important to us.

This figure is the main reason for point #2 in the list of major critics. This graph, in my opinion, needs way more explanation and discussion – also in the text, not only the caption. Like in other parts in the text, it is missing information on the approximation to the Stokes equations that is being studied here. Secondly, the informed reader might want to know on which computational platform this was run on and if we look at a single- or multiple node run. I already mentioned that a flow-line model in my view is a non-representative example for scalability if one wants to get a picture on how the code would perform and scale being applied to full ice-sheet problems. Yet, this seems to be the only place in the manuscript where the reader can get an idea on a performance baseline in terms of solved time-steps/wall clock time. I already suggested to do scalability tests with MISMIP+ if not on the full Antarctic setup. Furthermore, I would like to learn more on how the authors determine and separate ice dynamics and non-ice-dynamics parts in this figure. From a pure computational science point of view, I interpret the fact that a run on 64 and even 128 cores consumes a comparable wall-clock time as a 2 core run points to a serious issue in the parallel implementation that in my opinion prohibits production runs on compute clusters.

We agree that the text accompanying this figure needs to be more informative. In response to earlier comments, we will add the relevant information to the manuscript (platform, type of nodes, experimental set-up), plus some additional discussion regarding the poor scaling.

Do the authors mean that there is some automatic parsing of the meta-data of the NetCDF file (CF convention?) that deduces the coordinate system? Further question: is UGRID format meant when referring to triangular meshes?

The model parses the dimensions of the NetCDF file. If, for example, the file contains a dimension called “x” (or “X”, “x-dimension”, “x-coordinate”, etc.) and a dimension called “y”, it assumes the file contains data on a square grid, and tries to read it accordingly. If not, it looks for “lon” and “lat” (again with a set of alternatives for both) and if those are found, it assumes the file contains data on a global grid. Lastly, it looks for the set of dimensions used for the UFEMISM mesh format (which is not Ugrid; the option to output data in Ugrid format is planned for future work). We will mention this in the manuscript.

To me, in the lower row in Fig 4 displays the 1st order results (not tremendously, yet visible) surface velocities to exceed these of the ensemble. I would ask to explain why this is the case and – since the authors do not seem to raise any concern in the text – why it can be neglected.

…

I could not find any mentioning in Pattyn et al. (2012) of a lateral circular symmetry to apply to MISMIP flowline setups.

…

Like in the caption of Figure 5 on page 16, the information on what approximation to the Stokes-equation has been used for this resolution-experiments is missing.

…

As mentioned in the major points of critics #3, I would ask the authors to relate parameter choice and the reduced time-span of the experiments to the original MISMIP protocol and explain – also in light of the argument of verification – this deviation.
…
Like in the MISMIP chapter, information on the applied approximation to run the MISMIP+ experiments seems to be missing, also in the caption of Fig. 5. Please add this information.

See our previous answers to Major point of critique #3. All the requested information will be added to the manuscript.

If this is about computational performance, I have to disagree. From this paper I am lacking information to really judge the computational performance of the code. Deducing from Fig. 2, I would even conclude that there is an unresolved issue what comes to scalability of the code. If it is about code verification, I previously mentioned that the MISMIP tests to me do not provide the means to deliver on that aspect, as they deviate from the original protocol which prohibits comparison, which leaves ISMIP-HOM (with some approximation showing strong deviations) and MISMIP+, where I could not deduce what approximation has been used for intercomparison with ensemble results.

Our previous answers will see some additional information added to the manuscript to aid the reader in interpreting the results.

We agree that the poor computational performance, especially when moving to multiple nodes, is an unresolved issue. Since our budget for IT support has regrettably run its course, we do not foresee this being resolved in the near future.

As mentioned earlier (point #2 of major points), in my view this paper is lacking information to really judge about performance of the code, as the reader is not provided with a baseline value. In my opinion, wording like "much faster" are not conveying enough information to the reader that would enable quantification of the code's performance. To really judge about performance, the reader would need to get an idea on a performance-baseline. For instance, how much the solution of one ISMIP 6 scenario run for Antarctic ice sheet using a particular approximation (preferably the optimal one, hence BPN) needs wall clock time on one, two or even more nodes of a computing cluster. From figure 2 I would draw the conclusion that the code does not scale beyond 8 cores (of whatever platform it was run on).

It is difficult to come up with an experiment that would fairly compare the computational performance of v 1.0 and v2.0. v1.0 solved a simplified version of the SSA, where the gradients in the effective viscosity were neglected. This greatly improved the model's computational performance, but could significantly reduce the physical accuracy of the solution in certain cases (including, as it turned out, geometries with migrating grounding lines). We did, in the early stages of development, perform some preliminary experiments where we solved the same

simplified SSA in both v1.0 and v2.0, which is what the claim that v2.0 is "much faster" (which we agree is too vague) is based on. However, we regrettably did not save those experiments, and the option to use the simplified SSA has since been removed from v2.0 (due to its poor physical accuracy), so we cannot repeat them at this time. Additionally, v1.0 defined ice velocities on the grid edges (similar to an Arakawa-C grid), whereas v2.0 defines them on the triangle centers (similar to an Arakawa-B grid), which introduces an additional difference.

Accepting all these differences and simply comparing the total time for e.g. an ISMIP projection is difficult, because v1.0 lacks the modules for inverting for basal friction or sub-shelf melt, or for reading in external files describing those fields. Turning the problem around and taking an Antarctic experiment from v1.0 (which would do a non-nudged spin-up, use a greatly simplified ocean, and thereby result in a very different present-day ice geometry) would likewise be unfair, since the resulting geometry would be much 'smoother' and more stable than one resulting from a nudged spin-up, thereby artificially improving the model's stability and inflating its performance.

We will add a paragraph to the Discussion section reflecting these thoughts.

> *As I mentioned before, statements like this to me are impossible to evaluate without providing a baseline value. Just the fact that some algorithm takes 50 times longer (presumably using the same amount of computational resources) in my view does not imply that it is impossible to solve it – in particular if the code is claimed to run parallel (should in theory be able to use 50 times more resources, provided it scales) and comes with mesh-adaptation scheme.*

As mentioned in this paragraph, we do believe that using the BPA will become feasible if and when the scaling issue is resolved.

> *I wonder, should there be larger thickness changes in the active region, how does the model deal with the artificially imposed hydrostatic pressure gradients at boundaries with a one-sided fixed ice-thickness?*

If the thickness change in the active region would become large enough to significantly affect the surface slopes near the (artificially fixed) ice divide, then the assumption of a fixed divide would be invalid anyway. However, to be honest, we do not expect much use of this new option, as we believe the approach of simulating the entire ice sheet at a coarse resolution, and only the drainage basin one is interested in at a higher resolution (which UFEMISM can do), is much more practical.

> *To me this again lacks information to really get a clear picture on what type of simulations with what approximation and what accuracy can be achieved with what computational resources. There is only information on minimum resolution, but not the approximation used. Multi-day simulations are to me nothing that renders a computational problem impossible. But to me this sentence backs my suggestion that a*

fusion of the two papers would be beneficial to the reader to pick information that I conclude exists in the other manuscript (Bernales et al., in prep) to be able to relate statements presented in this one.

We hope the additional information provided in both our responses above and the revised manuscript satisfies the reviewer, as merging the two manuscripts is not a feasible option for us.

Looking at the distances of the polygon-points at the zoom-in over Ronne-Filchner ice-shelf, I conclude that this is far away from the acclaimed 5 km resolution. If this is really the resolution to start from, I would ask the authors to include a sentence on how the accuracy of the coastline is increased when increasing the mesh density: is it just linearly interpolated between existing points or is additional geometrical information added to the polygon?

The figure shows the coastline resulting from the BedMachine Antarctica v3 dataset at 40 km resolution, so using this to generate a 5-km mesh would not be useful. Choosing a higher resolution DEM would fill up the figure with dots, making it difficult to read. We will clarify this in the manuscript.

It is also important to note that a mesh, by itself, has no concept of 'bedrock elevation' or 'coastline'. When it is refined, it is simply presented with a set of line segments in the 2-D plane; where those segments originated does not matter. It is only when the data fields representing the Earth's geometry (bedrock elevation, sea surface elevation, and ice thickness) are projected onto the mesh, that the 'coastline' is defined. Setting up an ice model with a high-resolution coastline therefore requires 1) an input DEM with a suitably high resolution, 2) extracting the coastline from that DEM, 3) using that coastline as a refinement criterion for a mesh, and 4) projecting the DEM data fields to the resulting mesh. If the input DEM were to have a 40 km resolution, then we could theoretically use that to refine a mesh with a 5 km resolution around the coastline. Naively, we could then say that the resulting ice model resolves the coastline to 5 km, but the bedrock elevation itself in that region would be very smooth, as the input DEM did not have any information below the 40 km scale in the first place.

This is a large figure. I would try to simplify the composition resulting now into entry e) and thereby reduce (entries a – e) the size of the figure. Instead, what I would welcome to see included is a visual demonstration on how the remeshing algorithm could enhance mesh densities in areas of large derivatives of the velocity field, which turned out to be essential to resolve (thermo-)dynamics of fast outlet glaciers (e.g., Zhao et al. 2018) in other applications.

We agree that steps d and e are perhaps unnecessary; we will remove them.

UFEMISM currently does not include the option to use the velocity gradients as a mesh refinement criterion. However, the basic components to create this functionality are there. First you would need to define the 2-D polygons that envelop the regions where the strain rates

exceed a certain threshold (simply done with the existing calc_mesh_contours routine), and then simply use those polygons as refinement criteria (Fig. A2 only illustrates refinement around 0-D points and 1-D lines, but 2-D polygons are supported too; these are currently used to define resolutions over individual ice drainage basins).

> *This whole part appears abstract to me and without looking things up in Syrakos et al. (2017), difficult to understand. I would even suggest that, equally, a simple reference to the paper above and removing whole part B would shorten the manuscript. If the authors want to keep it, in my opinion it would help to have some figure of local grid configurations annotated with the node-indices and showing the most important features, like distances (Δxj, Δyj) and value entries ($f\ a\ j$) to better help illustrating the formulation of the stencils as presented in this appendix, also, with respect to the different discussed mesh-types and the coordination numbers of nodes/variables therein.*

Our approach expands upon the work by Syrakos et al. (2017) by adding the second-order derivatives, and by including the derivation for a staggered grid, so that a reference to that paper would not be sufficient.

We will add an illustration to help the reader with the different geometrical quantities.

> *In my view, it would be beneficial to the reader to explain what the third case "otherwise" means. With respect to definition in the text that indices $j \in [1, n]$ indicate all neighbours of i, I must miss something by interpreting that neighbours are connected by definition and "otherwise" being irrelevant for A-grids. Like mentioned before, some graphical display of a local mesh arrangement in my opinion could aid the understanding and prevent misinterpretations.*

There was a typo in Eq.s B9, B10, and B15-17. In all these equations, the subscripts in the second case (if j is connected to i) should be j, not c. Together with the new figure illustrating the local mesh geometry, this fix should clear up the confusion.

> *As remeshing/-mapping seems to be one of the main new features of UFEMISM v2.0, I would suggest to move this part (at least the non-mathematical) into the main section of the paper and rather take out other parts from there (I already suggested earlier)*

We believe the main body of the manuscript should focus on the ice dynamics and user experience of the model, and that the underlying math can remain in the appendix.

> *Like before, I find this section relatively abstract and difficult to read. Similar as before, I would be of the opinion that some graphics on the mesh-to-mesh projections showing the domain, its boundary and the two (source and destination) meshes to get a picture what this is about.*

Some readers that have not dealt with dual graphs might not know what a Voronoi cell is. Displaying this in a graph (see point above) and defining it in the text, in my view, would improve the readability of this chapter.

We will add an illustration of the geometry of the two meshes to aid the reader.

List of less important issues

I guess the authors want to express that the code can now be run on distributed memory architecture (referring to architecture as being the hardware, rather than the program itself)? If referring to code, I would suggest to change to: … from a shared- to a distributed memory implementation.

Accepted.

In my opinion, a statement like this would demand references to be included.

Although we definitely remember seeing this approach in several papers (and conference talks, etc.) we cannot seem to find the references. We will adjust the statement in the manuscript.

The brackets show the vertical derivatives of the components constituting the horizontal strain rates (and not membrane stresses).

We will change the phrasing.

page 11 – line 297: processors - I would use cores

Accepted.

One group of experiments does include flat beds (C,D,F), the other (A,B - as correctly pointed out in the sentence to follow) compute on an undulated bed and Experiment E on a glacier flowline (which is not flat, either).

We will change the phrasing.

Since remapping is mentioned here, I would suggest to make a reference to Appendix C for the reader's convenience to quickly look things up.

Accepted.

nV at this stage appears to me as being undefined (guess, it relates to some correlation number)

nV is the number of vertices in the mesh; we will mention this in the manuscript.

In all equations the lower symmetry entries in the matrix are omitted. Despite the redundancy, in my view it is preferable to either spell things out in the equation or mention the unusual notation in the text.

We have encountered this notation regularly.

Just a suggestions, but one could elegantly use the Kronecker delta to define $Dij = \delta ij\, fi$ *to achieve a better typesetting result in that line.*

Accepted.