Outline of the 3D calving algorithm in Elmer/Ice

1 Introduction

This document outlines the new 3D calving in Elmer/Ice detailing the underlying modelling choices not possible to include in the associated publication that has been submited to Geophysical Model Development. The bulk of this document forms part a chapter in Iain Wheel's PhD thesis which will be available on the Elmer/Ice wiki. For users, detailed documentation outlining the revelant keywords for each solver is avaiable in the Elmer/Ice GitHub repository (`./elmerice/Solvers/Documentation/`). The relevant documents are `Calving_lset.md`, `CalvingGlacierAdvance3D.md` and `CalvingRemeshMMG.md`. The associated test case also be found in the Elmer/Ice repository (`./elmerice/Tests/Calving3D_lset`). Finally, work is ongoing to fully parallelise the entire algorithm. The current progress is detailed at the end.

2 Front advance

The front advance solver computes the advection of the terminus over a given timestep. The original front advance solver was updated since it did not allow the full calving front to migrate (Todd et al., 2018). A new solver was created, `CalvingGlacierAdvance3D.F90` to implement the lateral margin migration (van Dongen, 2021). For the majority of the calving front this is fully Lagrangian, with front advance being implemented based on the computed velocity fields. Here, the displacement vector ($\boldsymbol{d}$) is the velocity multiplied by the timestep. However, at the intersections of the lateral and calving boundaries of the glacier, the displacement vector cannot be based

purely on the velocity since the terminus advance must follow the fjord walls. This could be solved as a contact problem analogously to the grounding line (Durand et al., 2009), but this would increase computational requirements as the non-linearity of the problem increases. Instead, fjord walls are user defined before the simulation and the velocity is projected along that margin (Fig 4.2). As such the displacement in the x and y plane at the lateral margins ($\boldsymbol{d_l}$) is defined by

$$\boldsymbol{d_l} = |\mathbf{u}| \times \vec{f}, \tag{1}$$

where $\boldsymbol{u}$ is the velocity and $\vec{f}$ is the direction of the fjord wall. Although this is very computationally efficient, it may lead to artificial mass change as the lateral boundary corners are not guaranteed to obey the incompressibility condition. This artificial mass change is relatively low, as the velocity remains parallel to the lateral margin because of the standard lateral boundary condition of zero normal velocity. The only location where velocities are not parallel to the lateral margin are the lateral boundary corners. This is because the normal vectors are poorly defined at these locations. This provides a much better solution than having the lateral corners fixed in place (van Dongen, 2021). The original concept and implementation were completed by Eef van Dongen and Joe Todd but further developments are all my work. On its conception the new solver, `CalvingGlacierAdvance3D.F90`, produced front advance for simple geometries. I developed it further, increasing the robustness and adding new functionality to deal with complex geometries and the capability to move boundary elements (see Sections 4.4.1; 4.4.3; 4.4.4).

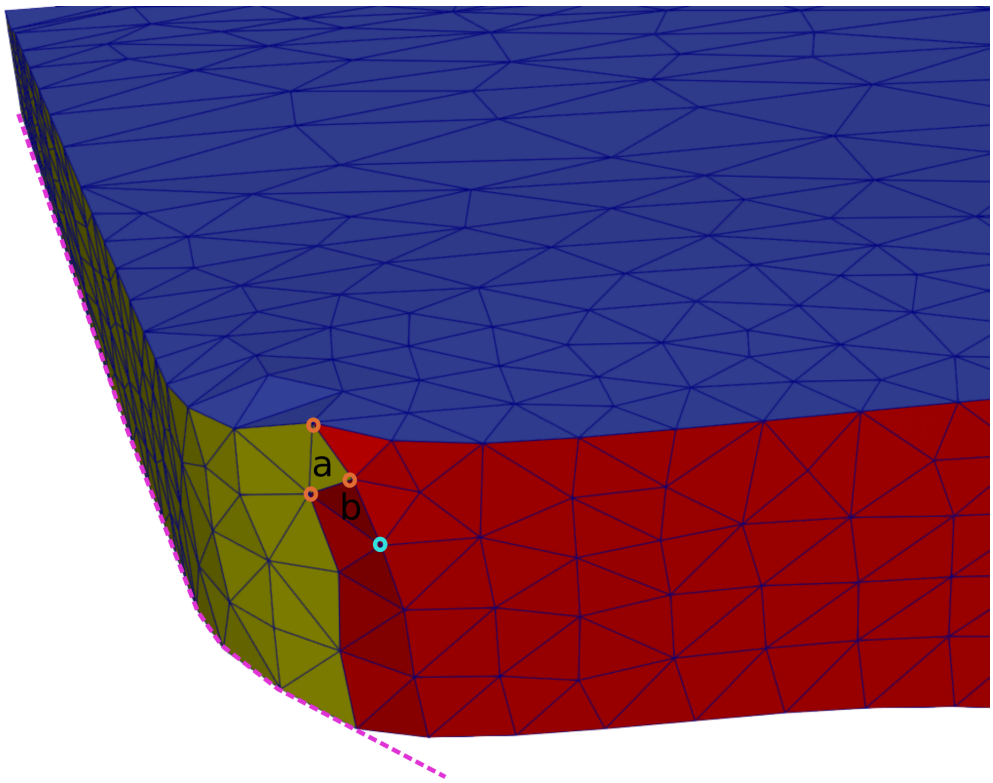

Figure 1: An example of lateral element transfer on a narrowing fjord. The red elements are those on the front boundary, the yellow elements those on the right lateral boundary and blue are the surface elements. The pink dotted line shows the fjord rails used to direct the lateral advance. This is only present on the xy plane and has no z coordinates so should be thought of as a plane projecting along the z axis. It is only shown as a line to make visualisation easier. The red circles indicate nodes present on the rails and the blue circle shows a node not on fjord rails. Element (a) has been transferred to the lateral boundary as all nodes are on the rails. Element (b) remains on the front boundary since one node is not on the rail. Once this node indicated by the blue circle reaches the rails element (b) will be transferred to the lateral boundary.

2.1 Transfer of boundary elements

Due to the friction on the lateral margins, the centre front of the glacier advances more quickly than the corners of the glacier front (Fig. 4.3). With lower velocities present on the lateral margin, the Lagrangian movement of the calving front can reach the fjord walls. This can be clearly shown on a narrowing fjord when the front will reach the fjord wall before the leading lateral node reaches the same position (Fig. 4.2; 4.3). Nodes only present in calving front boundary elements are projected in a Lagrangian way but they must be checked to ensure they do not intersect the fjord boundaries. Once they intersect the fjord walls they are instead projected along the margins. If all nodes in a calving front boundary element are present on the fjord walls it is converted to a lateral boundary element, in order to ensure lateral boundary conditions are maintained along the fjord walls as the glacier advances (Fig. 4.2). It should be noted the fjord walls are only present as a line in the xy plane as shown in Fig 4.2 rather than a 2D surface.

As the domain of the model evolves over time, the position of the boundary elements moves. Given the varying boundary conditions applied to each boundary, it is essential to ensure elements which have moved have the correct boundary identification. Consequently, the model firstly requires to be able to determine elements that need to be moved, before secondly, moving the nodes to the correct boundary. It is quite complex to describe every scenario where this may occur, but a coherent example is provided in Section 4.4.3 for a narrowing fjord.

2.2 Mesh deformation

This section explains how the glacier domain is advanced, before examples illustrating it are given in Section 4.4.3. Once the predicted terminus advance has been calculated, the nodes in the mesh are physically moved altering the domain to represent the advance seen at tidewater glaciers. Mesh nodes are moved but the elements are not altered. This deforms the glacier mesh rather than producing a new mesh for the domain as is the case in remeshing. The deformation is applied across the domain rather than just the terminus so element stretching is spread throughout. Otherwise, the distortion would be concentrated on frontal elements. Based on the simulated terminus advance, the mesh is adjusted across a 2D x - and y - plane using the solver `MeshSolve.F90`.

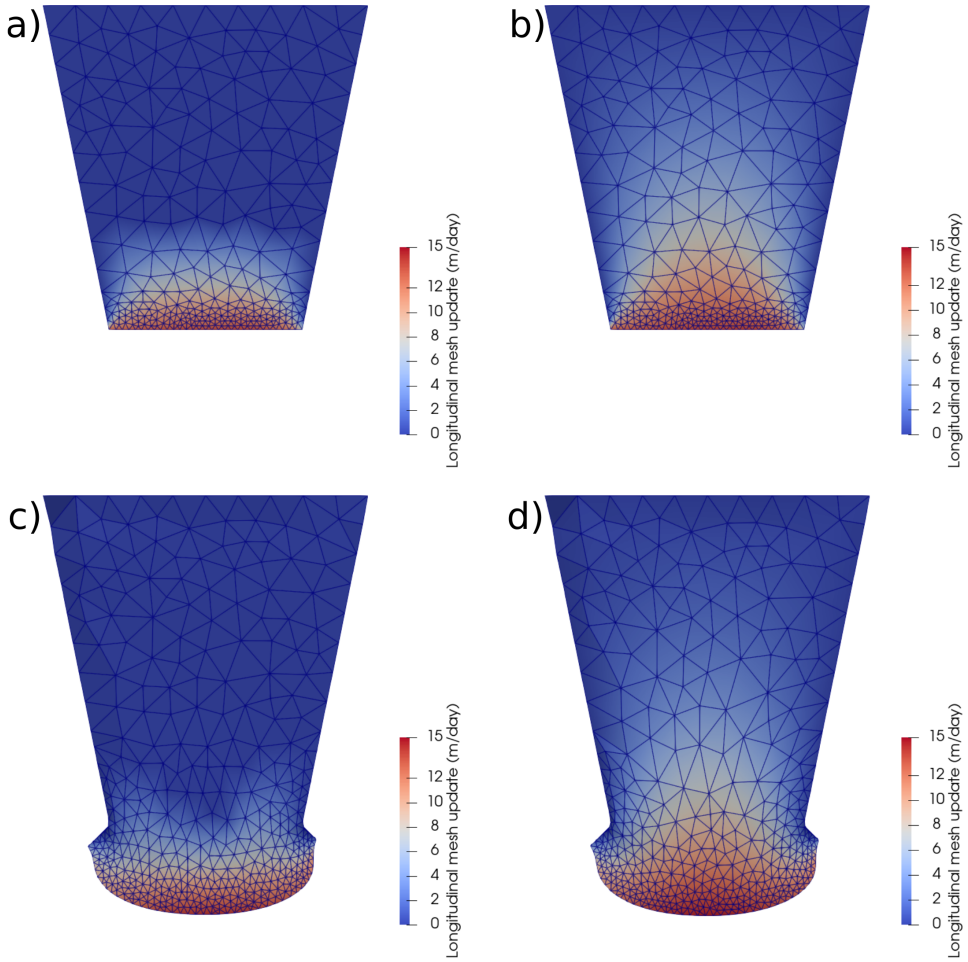

Figure 2: An example of different mesh deforming techniques when using the glacier advance routines. The simulations were run for a year, with calving suppressed, on an example domain of a widening fjord. a) and b) show the initial identical geometries. Although the geometries are identical, the area of mesh that is deformed varies. c) and d) show the final geometries after one year of simulation. Note the corresponding geometries should be identical even if the element quality differs. a) and c) show the simulation when only the area being remeshed is deformed while b) and d) show a simulation where the mesh deformation is applied to the whole domain. Note the elongated elements appearing in d) when compared to c). Full simulations can be seen in S4.3.1 and S4.3.2 (see Appendix).

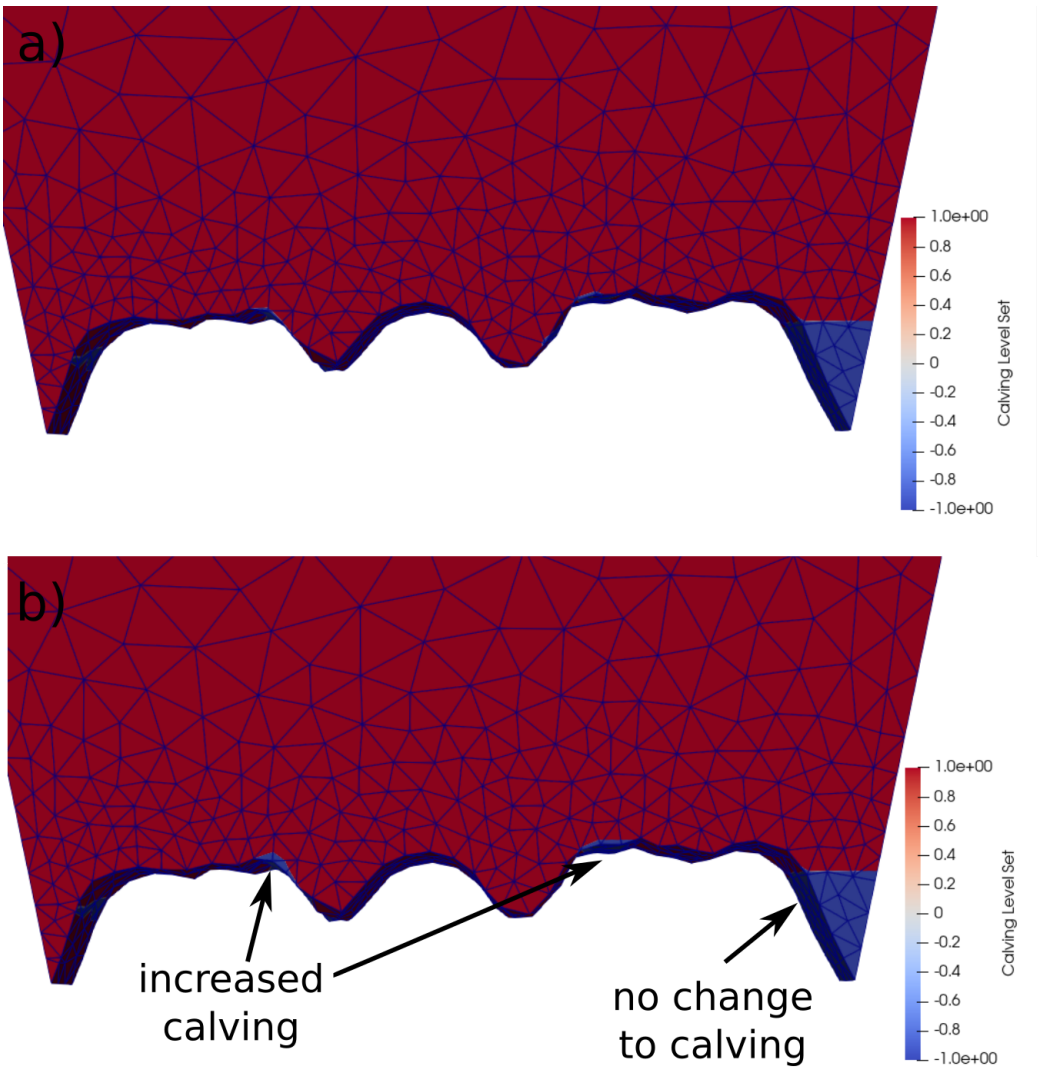

Figure 3: Comparison of the resultant calving when different mesh deformation are applied when using the glacier front advance routines. a) The predicted calving shown by the calving level set variable when mesh deformation is applied to the area of the glacier being remeshed. b) The same set up but with mesh deformation applied only to the glacier front. Note the increased calving if the iceberg depth is less than the element edge length while there is no change if the iceberg depth is greater than the element edge length. The reduced range of the calving level set variable is just for viewing purposes so the predicted new calving front can be seen clearly.

There are two components to `MeshSolve.F90`. The first component determines the best solution to adjust nodes smoothly across the domain given a set of boundary conditions. The second component physically adjusts the mesh.

Allowing `MeshSolve` access to the whole domain, over the course of multiple timesteps, allows mesh adjustment that can lead to very elongated elements in the direction of the advance. It assumed remeshing is successful enough to prevent this occurring only within the remeshing area. Therefore, it is recommended that the mesh update is only applied to the area being remeshed and not the whole glacier (see section 4.6.1 for details on how the remeshing area is defined). If applied to the whole glacier, regions far upstream beyond the remeshing area will slowly degrade and become more elongated and distorted over time (Fig. 4.3). Initial conditions can be set to limit `MeshSolve` access to only nodes within the remeshing area. If nodes are not adjusted using `MeshSolve` and instead just the boundary nodes simply advance prior to remeshing, elongated front elements lead to a very different solution to the calving projection (Fig. 4.4). Examples of each of these situations can be seen in Fig. 4.3 and Fig. 4.4.

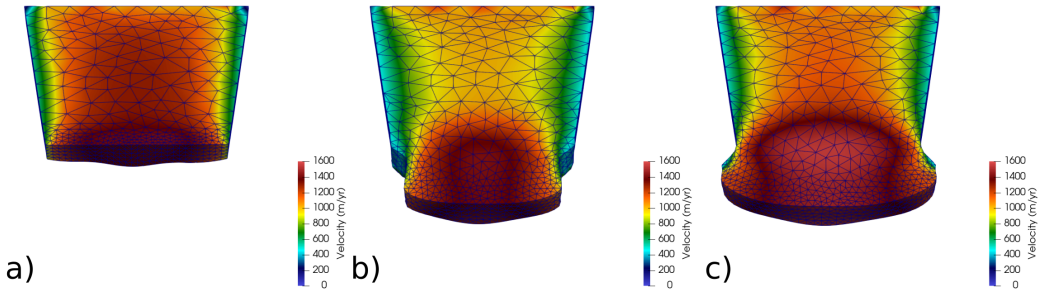

Figure 4: Two examples showing the capabilities of the front advance routines over one year simulations. Calving is suppressed for both simulations so the resultant geometry is purely from glacier advance. a) The initial geometry prior to simulation. Final geometry and velocity field for b) a fjord with a narrowing and c) a widening fjord. The full simulations can be seen in S4.5.1 and S4.5.2 (see Appendix).

Lateral advance in a narrowing fjord

A narrowing fjord provides a much more robust test of `CalvingGlacierAdvance3D`. In a narrowing fjord, front nodes will often 'hit' the fjord walls requiring them to be reprojected along these walls. Boundary elements may also need to be transferred as well. As expected, a narrowing fjord causes the glacier to decelerate, especially further upstream and in the margins close to the narrowing (Fig. 4.5b). The highest velocity is seen in the centre, in the narrow section of the fjord. It is the speed of this ice which limits the flow of the rest of glacier. Again, the velocity fields look as expected for a glacier flowing into a narrowing fjord (Fig. 4.5b).

2.3 Frontal melting

Within the solver `CalvingGlacierAdvance3D.F90`, frontal melt rates can be specified as a scalar variable. The direction of melt is always assumed to be normal to the terminus. Following the Lagrangian implementation of the glacier advance, melt is prescribed such that

$$\boldsymbol{d} = \boldsymbol{u} - \hat{m}, \tag{2}$$

$$\boldsymbol{d_l} = |\boldsymbol{u}| \times \vec{f} - \hat{m}, \tag{3}$$

where $\hat{m}$ is the melt normal to the front. Melting with any form of vertical or horizontal profile can be implemented. Given the simplicity of this method there are very few issues that can arise when applying melt. Degenerate elements will only be produced if the melt per timestep is larger than the element length in the normal direction.

I have written an independent solver (`GetFrontMelt.F90`) that incorporates a plume and background melt profile. The solver applies the background melt across the terminus and additional plume melt is applied over a set width centred at the plume location. There are a multitude of ways melt can be calculated in Elmer based on basal hydrology or fjord conditions (e.g. Cook et al. 2022; Todd et al. 2018). For this study, the melt is not calculated in Elmer; instead melt profiles are applied from other studies.

A simple example of central plume of 500m with a constant background melt is shown in Fig. 4.6. As the glacier advances the increased melt provided

by the plume profile leads to substantial undercutting and a melt toe at the base (Fig 4.6).

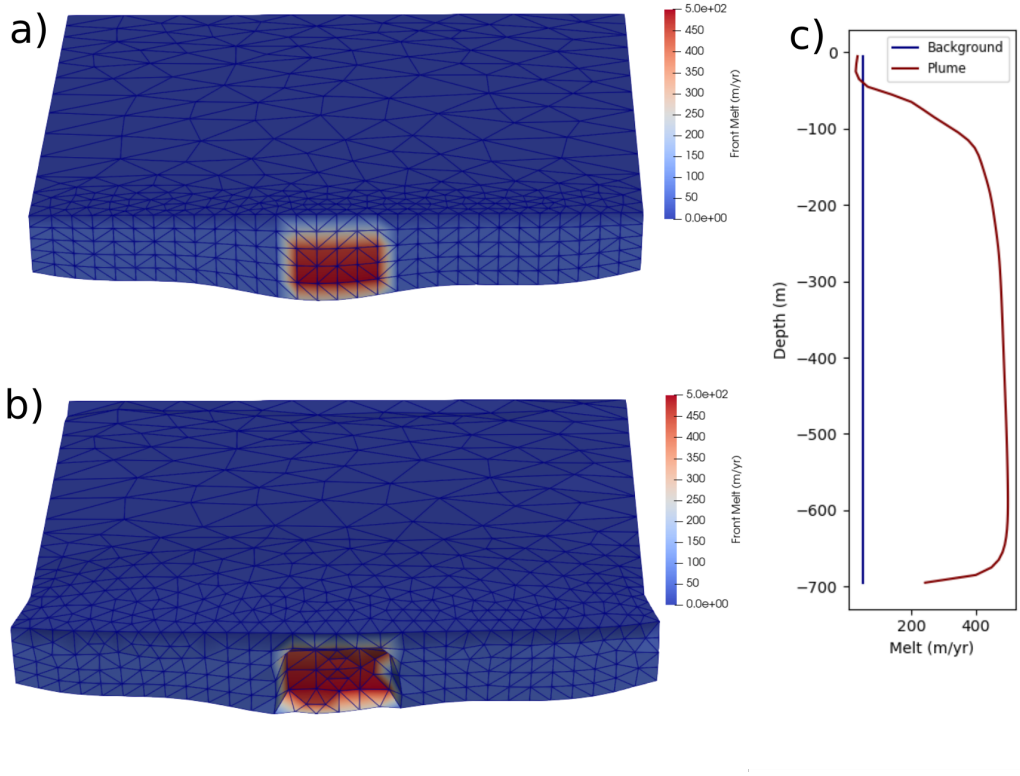

Figure 5: A simple example of a melt field with a central plume applied to glacier advance routines. Calving was suppressed and the simulation ran for 100 days. a) The initial geometry showing the melt field across the terminus. b) The geometry after 100 days with a large area of undercutting where the plume is located. c) The melt profiles applied. The background melt shown by the blue line is set to a constant value of 50m/yr and the plume melt shown is red is limited to a maximum value of 500m/yr. The full simulation can be seen in S4.6 (see Appendix).

3 Calving Projection

This section focuses on the definition of the calving front as defined by the crevasse depth calving law (CDL) as opposed to implementation of the calving. The complexity outlined here is due to the physical basis of the calving law. One could replace this with any calving law and a simple rate-based calving law would be easy to implement. The calving projection is the difference between the new terminus determined by the CDL and current terminus position. The calving projection outlined here is heavily based on Todd et al. (2018) but with key developments that allow for non-projectible calving. This development is included in a new solver (`Calving3D_lset.F90`) and is outlined in Fig. 4.7. This solver takes in the Cauchy stress tensor as an input and calculates the predicted calving via an output calving level set variable. The CDL is unchanged from Todd et al. (2018).

3.1 Calving Criterion

The physical calving criterion remains unchanged from Todd et al. (2018) and is based on the crevasse depth calving criterion (Benn et al., 2007; Nick et al., 2010). Calving occurs under two conditions: (1) a surface crevasse penetrates to the depth of the waterline (Benn et al., 2007) and (2) when basal crevasses extend to reach surface crevasses (Nick et al., 2010). A modified Nye criterion is used to predict crevasse depth since it assumes only extensional stress will lead to crevasse opening (Nye, 1957). Crevasses occur when the effective principal stress (σ_p) is positive. A negative principal stress will result in the ice remaining intact. The principal stress for basal and surface crevasses is defined as

$$\sigma_{p,surf} = \sigma_1, \tag{4}$$

$$\sigma_{p,basal} = \sigma_1 + P_w, \tag{5}$$

where σ_1 is the largest Cauchy stress and P_w is the water pressure at basal crevasses. Water pressure is added to Eq. 4.5 as hydrofracture is a key component of basal crevasses formation (Nick et al., 2010). During the melt season, water is also known to occur in surface crevasses although its presence and quantity is hard to predict. However, unlike basal crevasses, those on the surface can open without the addition of water pressure. Given the complexity and unknown importance of the role of water pressure in surface crevasses, it is not included. The basal water pressure in the near terminus

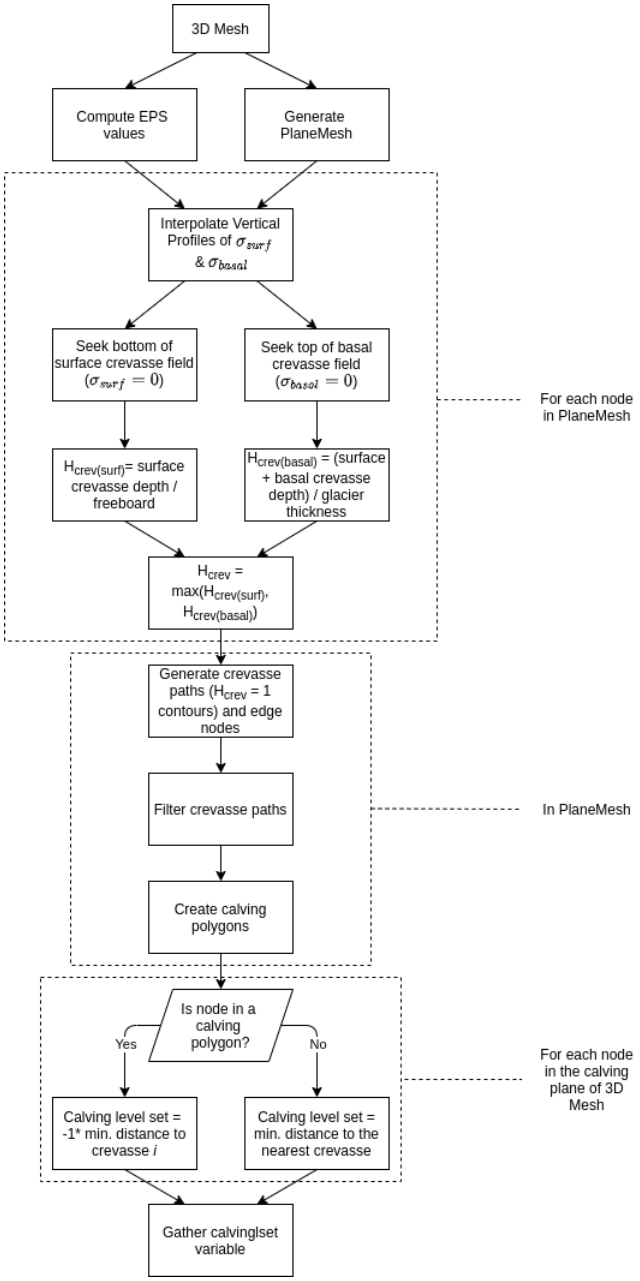

Figure 6: Diagram outlining the calving prediction based off the crevasse depth calving law. Stress values are taken from the 3D glacier mesh to the 2D `PlaneMesh` and back onto the 3D mesh as the calving level set variable. Crevasse i is the crevasse corresponding to the calving polygon the node sits inside. Adapted from Todd et al., 2018.

area is assumed to be equivalent to the sea pressure (Todd et al., 2018). Basal water pressure (P_{wb}) is therefore

$$P_{wb} = (z_{sl} - z_b)\rho_{sw}g, \tag{6}$$

where z_{sl} and z_b are the elevations of the sea level and glacier base respectively. The density of seawater is ρ_{sw} and g the acceleration due to gravity. It is also assumed basal crevasses fill with fresh meltwater from an efficient drainage system (Todd, 2016; Todd et al., 2018). Internal basal crevasse water pressure (P_w) is defined as

$$P_w = P_{wb} - (z - z_b)\rho_{fw}g, \tag{7}$$

where ρ_{fw} is the density of freshwater.

Although the above calving criterion is chosen, others could be implemented easily within the architecture of the calving algorithm.

3.2 Prediction of calving

`Calving3D_lset` calls the `ProjectCalving.F90` solver which remains unchanged from Todd et al. (2018), though some optional features have now been added. This solver runs after the flow and stress fields are computed using `FlowSolver` and `ComputeDevStress`, respectively. (See Section 4.8 for a typical simulation set up). Based on the Cauchy stress tensors calculated in Elmer, an effective principal stress for both basal and surface cases is calculated for each node within the 3D mesh. A positive effective principal stress indicates ice fracture and crevasse presence, while a negative stress value means that the ice remains intact. A 2D square mesh known as `PlaneMesh` is created to cover the x-y plane of the 3D mesh. The 2D square mesh extends to a user-defined distance away from the terminus. Vertical ray casting through the 3D mesh is used to create σ_p profiles for each node in `PlaneMesh` (Todd et al. 2018; Fig. 4.8) from which surface and basal crevasse penetration ($H_{crev,surf}$ and $H_{crev,basal}$) are calculated. $H_{crev} = 1$ is equivalent to a crevasse penetrating the full thickness of the glacier at that node:

$$H_{crev,surf} = \frac{d_{surf}}{z}, \tag{8}$$

$$H_{crev,basal} = \frac{d_{surf} + d_{basal}}{H}, \tag{9}$$

$$H_{crev} = max(H_{crev,surf}, H_{crev,basal}), \tag{10}$$

where the depth of surface and basal crevasses are d_{surf} and d_{basal} respectively, z is the ice freeboard and H is ice thickness. The use of vertical ray casting means crevasses are always assumed to be vertical.

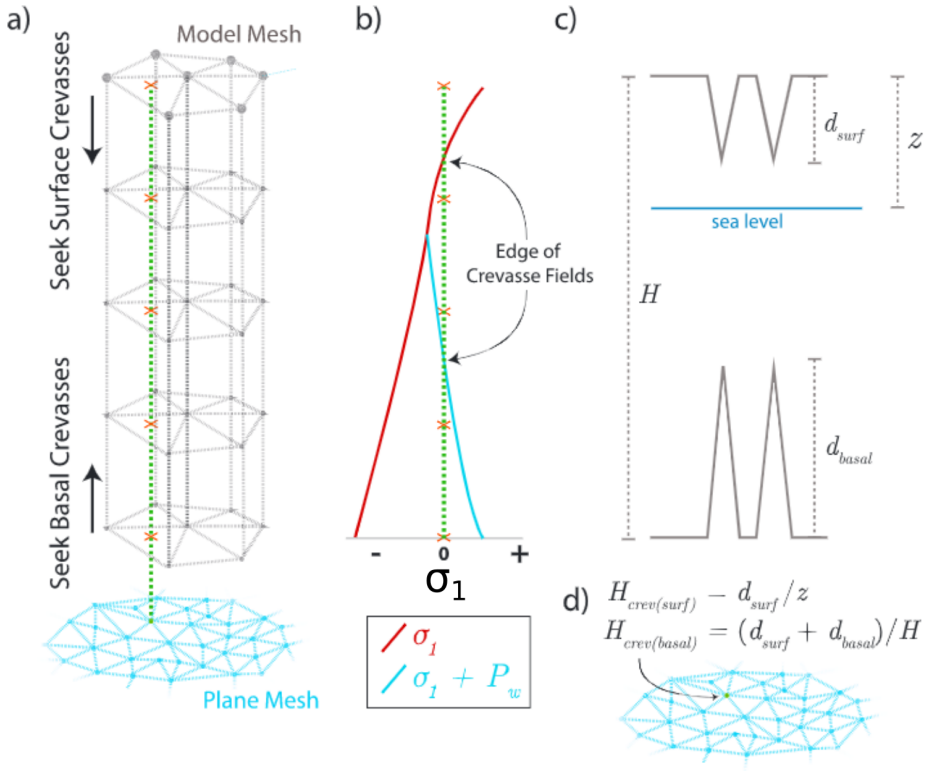

Figure 7: Schematic from Todd et al., 2018 showing the vertical ray casting and transfer of information from the 3D glacier mesh to the 2D `PlaneMesh`. a) Vertical ray casting in the 3D glacier mesh corresponding to the position on the `PlaneMesh`. Note the 3D mesh here is extruded as per the Todd calving algorithm. In the new calving algorithm the 3D glacier mesh has tetrahedral elements. b) Vertical profile of the net stress with the line showing the surface crevasse law and blue basal crevasses. c) Ice column profile showing the percentage of crevasse penetration which is prescribed onto the `PlaneMesh` using d).

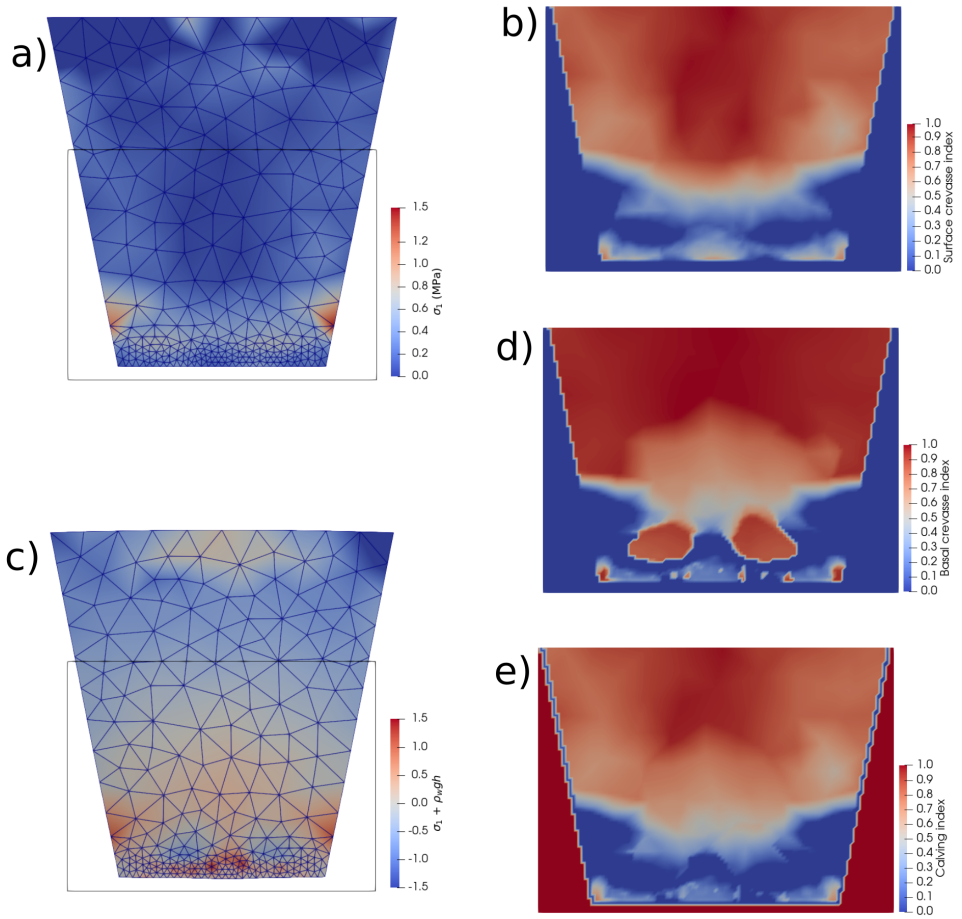

Figure 8: Simple visualisation of the relationship between stress values and the crevasse penetration. a) Top down view of the 3D glacier mesh showing the first principal stress. This forms the basis of the prediction of surface crevasses. b) The penetration of surface crevasses diplayed on the `PlaneMesh` (Surface crevasse index is $H_{crev,\,surf}$). c) Bottom up view of the 3D glacier mesh showing the first principal stress plus water pressure. This forms the basis of the prediction of basal crevasses. d) The 2D `PlaneMesh` showing the basal crevasses (Basal crevasse index is $H_{crev,\,basal}$). e) The 2D `PlaneMesh` showing the calving index. This a combination of both surface and basal crevasse prediction and antipode of H_{crev}. The black boxes in a) and c) indicate the areas where crevasses are projected from the stress field and match the domain of b), d) and e).

H_{crev}, the maximum of the surface and basal crevasse penetrations , is mapped on the 2D `PlaneMesh` as the calving index (Fig. 4.9). The 'Calving index' is the antipode of H_{crev} as representing the percentage of intact ice - where an index of one indicates a completely intact ice column and zero indicates full thickness crevassing. The $H_{crev} = 1$ contours are isolated and treated to post processing validation (see Section 4.5.3). The isolation of the crevasses has been modified to allow the user to define the crevasse penetration required to induce calving and is described in Section 4.5.4. These isolated contours are referred to as crevasses. Any other potential crevasses that do not reach the isolation threshold are not considered in the model and will not be referred to again. Future mentions of crevasses only refers to those that have been isolated above the calving threshold. This process is repeated for each timestep, so crevasses have no history and do not advect over time. Instead, crevasses should be thought of as representing a localised stress regime that could cause full thickness crevassing.

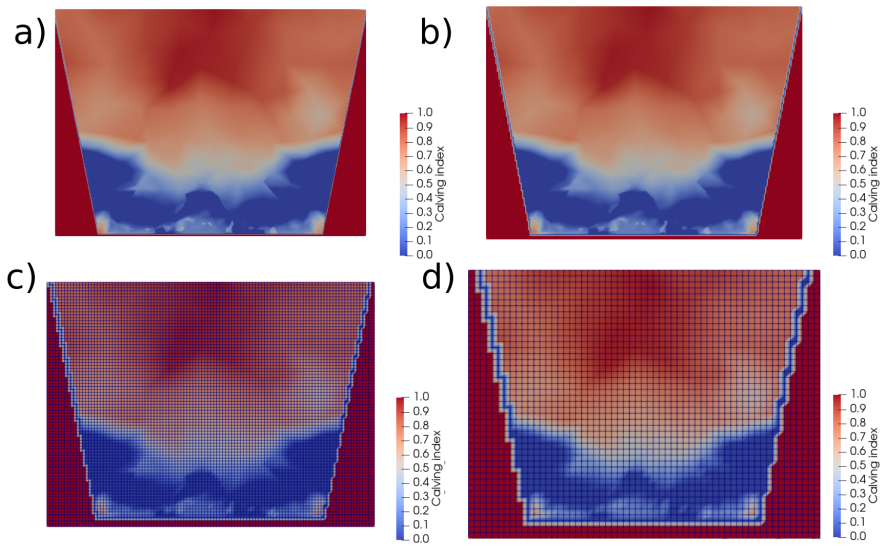

Figure 9: The impact of `PlaneMesh` element size on calving prediction. Each panel shows the crevasse penetration on the 2D `PlaneMesh` with a different element size: a) 10m, b) 20m, c) 40m and d) 80m. Only c) and d) show the element outlines since the elements are too small for this to be visualised in a) and b).

Advantageously, the PlaneMesh is distinct from the 3D mesh so the resolution can be increased without affecting the computational expense of solvers such as FlowSolve on the 3D mesh. The size of the PlaneMesh can be altered using the keywords, "PlaneMesh Grid Size", in the Calving3D_lset solver to alter the resolution of the crevasses (Fig. 4.10). It is noted that the resolution is not fully independent from the 3D mesh element size from which the values are projected. Comparatively coarse 3D mesh elements mean PlaneMesh values follow intra element interpolation rather than distinct 3D node values.

Crevasse validation

Crevasses are defined as isolines where the crevassing threshold has been met. The validation of crevasses follows Todd et al. (2018) but with some modifications to account for the potential for non-projectable calving fronts and changes in the lateral geometry as the glacier advances. Firstly, crevasses whose ends lie on the same lateral boundary are removed since there is no evidence lateral calving occurs in glaciers. Unlike Todd et al. (2018), crevasses which are located downstream of another crevasse are removed only at the end of a routine since their area could change with the calving filter.

The calving filter is used to prevent unrealistic calving and is applied after the crevasses are gathered from the 2D PlaneMesh (Todd et al., 2018). The Todd et al. (2018) calving filter is adjusted, with the mesh rotated to the orientation of the angle of the two boundary crevasse nodes, for each crevasse. Any constrictions can then be removed using the Todd et al. (2018) method. Briefly, the algorithm runs along the crevasse from right to left, marking nodes which lead to a counterclockwise turn (Fig. 4.11). Once marked, these nodes are inspected as pairs to see if any face each other. If a pair face each other, they form a constriction and the distance between them is calculated. If any node between the marked constriction on the crevasse path is farther from either constriction node than the constriction distance it is removed (Fig. 4.11).

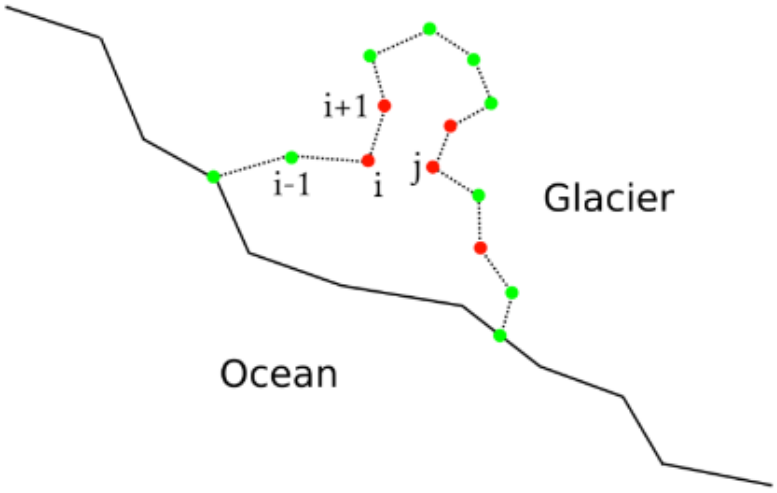

Figure 10: Schematic of the constriction algorithm. From Todd et al., 2018. Conceptual example of a $H_{crev} = 1$ contour with a constriction. Nodes which form a counterclockwise turn are checked to see if they produce a constriction. They are marked in red. Nodes marked in green form clockwise turns so cannot form constrictions. In this example the constriction is between i and j so all the nodes in between are removed.

Importantly, an option has been added to allow the attaching of the lateral boundary of the glacier to those crevasses that end on a lateral margin. This can be turned on or off using the keywords, "Lateral Calving Margins". When turned on, the `PlaneMesh` nodes along the lateral boundary that lie just outside the glacier domain are added to the crevasse prior to the realistic calving filter. This greatly reduces calving when the fjord walls are narrowing but has no effect otherwise (Fig. 4.12). Inclusion of a narrowing lateral margin often confines the evacuation of the iceberg reducing the calving volume.

Next, crevasses that lie downstream of another crevasse need to be removed so the level set surface can be properly defined. However, this is no longer trivial since the calving front is non-projectable (Fig. 4.12). When the calving variables are mapped onto the `PlaneMesh` using vertical ray casting, the number of element hits from the 3D mesh at each `PlaneMesh` node is also

calculated (Fig. 4.8). This variable is the "hit count". In other words, for a given `PlaneMesh` node, the hit count value is the number of 3D glacier elements that contain the `PlaneMesh` node in the x-y plane. `PlaneMesh` nodes that lie just outside the original 3D mesh domain are marked as edge nodes. These are defined as a node that has a hit count of zero but has a neighbouring node with a hit count greater than zero. The resultant edge node line is used to create 'calving polygons' (Fig. 4.13). Crevasses are combined with the edge nodes that link the crevasse ends to create a polygon. If any crevasse lies within the area of another calving polygon it is removed. The winding number algorithm (Alciatore and Miranda, 1995) is used to check if any point lies within a calving polygon. It is important that polygons are used since the calving front is no longer projectable, and so crevasse extents are not enough to determine if a point lies within a region predicted to calve (Figs. 4.12, 4.13).

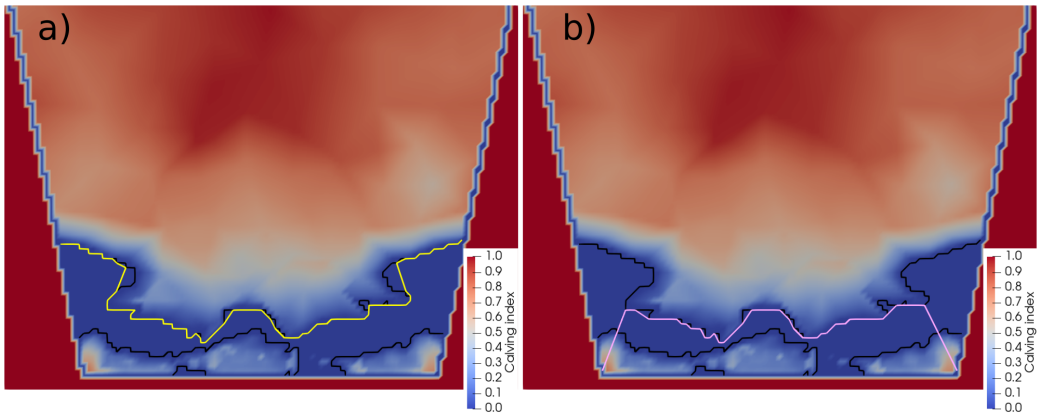

Figure 11: Validating crevasses. a) The `PlaneMesh` with the crevasse penetration map. The black lines show the unvalidated crevasses and the yellow lines show the validated crevasses not accounting for the lateral margins. It should be noted the unrealistic geometry of this configuration to calve in a single event. b) The same set up as a) but with the pink line denoting the validated crevasse after accounting for the narrowing lateral margins.

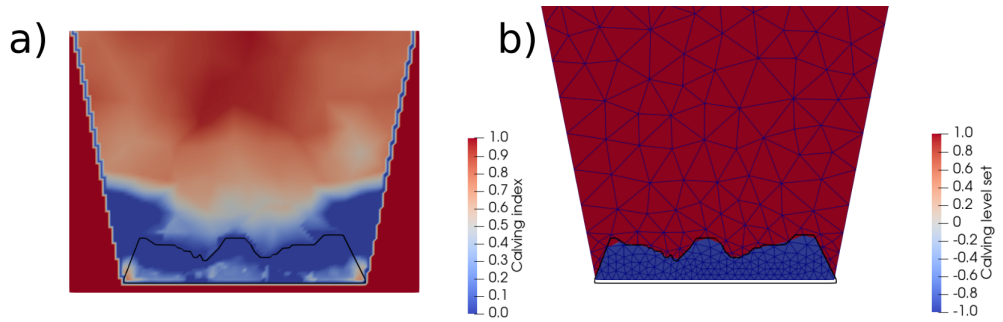

Figure 12: Visualisation of the translation of the calving polygons into the level set variable. a) The 2D `PlaneMesh` from which the crevasses are derived. The calving polygon is the validated crevasse with the addition of the edge nodes which connect its two ends. b) The calving level set variable on the 3D glacier mesh and the calving polygon overlaid. The calving level set is defined as the distance from the calving polygon with negative values inside and positive outside. The reduction of the viewing range of the variable is purely so it can be seen that the zero contour follows the calving polygon.

Lagrangian movement at the margins

As the glacier advances downstream, the velocity profile is highest in the centre before reducing towards the lateral margins due to the friction from the fjord walls. The front slowly distorts to match this profile. The slow-moving ice at the lateral margins becomes unsupported. It often calves before it reaches the fjord wall allowing the transfer of boundary elements, and so providing the buttressing support (Fig. 4.14). As the model has no concept of the geometries beyond the calving event this will occur even if the newly calved iceberg is only 1m away from the fjord wall and calving directly towards it. This consequently prevents the full front of the glacier from advancing, as it suppresses the lateral margins from realistic advance. To correct this, if calving is predicted, each iceberg is checked to see if its interior depth in the direction of evacuation is greater than the distance to the fjord wall along the same vector. If the iceberg depth is greater than the distance to the lateral margin, this particular crevasse is removed and calving is prevented (Fig. 4.14).

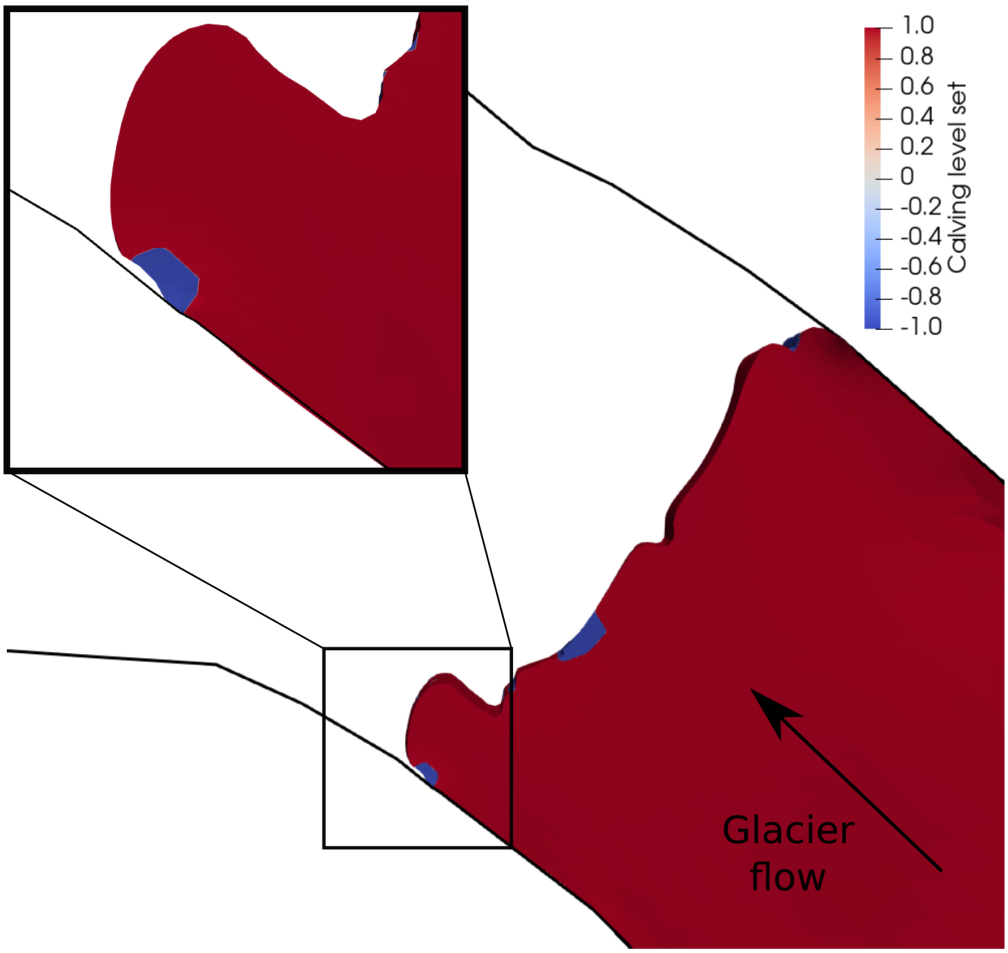

Figure 13: Calving is predicted at the portion of the lateral margin shown in the inset. Because of the proximity of the fjord walls, the calving event would jam and the iceberg would be prevented from being evacuated. Such a calving event is suppressed in the model.The black lines at the edge of the glacier show the fjord wall rails. The calving level set variable is limited in view to show the calving events. The top left panel shows a zoomed view of the area shown by the black box.

Level set variable calculation

After crevasse validation is complete, the calving level set variable is defined. Calving polygons and crevasse locations are first shared between all processes to allow the level set to be calculated in parallel. This is important for the scalability of the model. Each node in the 3D mesh is first checked to see if it lies within any calving polygon. If it does, the level set value is defined as the shortest distance between the corresponding crevasse multiplied by negative one. If the node lies outside all calving polygons, the level set is defined as the shortest distance to any crevasse (Figs. 4.7; 4.13). By calculating the level set in this order, errors are prevented as the closest crevasse may not be the one causing the node to calve.

3.3 Variations in the calving law

The large number of uncertainties involved in modelling calving from a physical basis in 3D means there will likely always be a number of processes missed or not included in any prediction. The CDL can be made more sensitive by reducing the percentage of the ice column required to crevasse to induce calving. This can be done to make the calving law more sensitive if calving is routinely underestimated (Fig. 4.15). A new set of keywords, "Crevasse Penetration", has been added in the `Calving3D_lset` solver that takes a value between one and zero. For example, if Crevasse Penetration is set to 0.75 then crevasses only need to penetrate three quarters of the ice column to induce calving - as opposed to the full thickness penetration needed by default. The use of the feature can be justified by the underestimation of the CDL (Cook et al., 2022, 2023; Todd et al., 2018). This is often hypothesised to be either because of the model's inability to account for stress concentrations (Slater and Benn, 2022; Van Der Veen, 1998) or ice history (Cook et al., 2023; Todd et al., 2018). The simple tuning of the crevasse penetration threshold allows for greater sensitivity of the calving law to mimic the effect of these processes without requiring further substantial model developments.

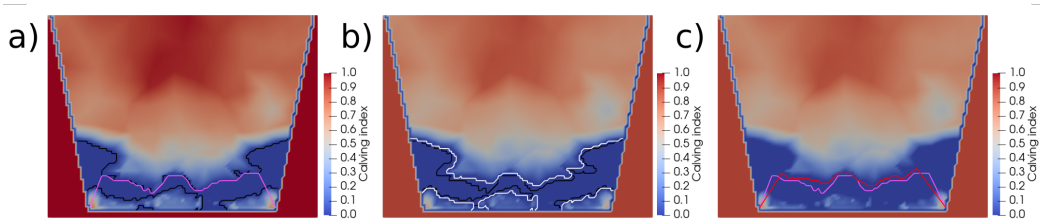

Figure 14: The effects of altering crevasse penetration requirement in the crevasse depth calving law. a) The `PlaneMesh` showing the crevasse penetration map with an unaltered calving law (i.e., full thickness penetration required for calving to occur). The black lines show the unvalidated crevasses and the pink line is the validated crevasse, accounting for the lateral margins. b) The crevasse penetration map on the `PlaneMesh` when the crevasse penetration required to induce calving is reduced to 90%. The black lines show the unvalidated crevasses from a) while the white lines show the unvalidated crevasses produced from this new configuration. The validated crevasse comparison is shown in c) where the pink line is the unaltered calving law (note this is from a)) and the red line is the altered one.

4 Serial Calving Algorithm

The calving algorithm is defined as the implementation of calving, or the removal of glacial ice as icebergs. It takes a level set or signed distance variable where the zero contour is the new calving front to produce a new mesh with all model variables interpolated across onto it. Importantly this new calving algorithm is not limited by iceberg or frontal geometries and consequently is not tied to a particular calving law. I will show its use with the crevasse depth calving law as outlined above as this represents the best physically based calving law in 3D.

Similar to the calving front advance, it was previously not possible for calving to occur at the lateral margins. Additionally, the Todd et al. (2018) model relied on a 'calving variable' which required the calving front to remain projectable, which is not a true representation of glacier termini. The solution implemented here to overcome the problems encountered by Todd et al. (2018) was to cut the mesh using a level set function (Osher and Fedkiw, 2001; Sethian, 1999).

The other major issue with the previous algorithm was element degeneracy

for certain geometries. These issues stemmed from the use of a vertically extruded mesh (Todd et al., 2019). Once the calving variable had been defined, the mesh was compressed or stretched to match vertical terminus geometry changes. This was particularly problematic when submarine melting was applied leading to a non-vertical terminus. In such geometries an average front position was determined from which a 2D footprint was created and meshed. Several filters had to be applied to move and delete nodes to reduce the prevalence of degenerate elements. The footprint was extruded vertically and then deformed to match the front geometry of the old mesh. Full details of the old calving algorithm are in Todd (2016). To overcome these issues, the remeshing software Mmg is used in the new calving algorithm to produce a fully 3D domain without the need for vertical extrusion (Dapogny et al., 2014). An additional benefit of using Mmg is that the calving algorithm is no longer reliant on command line calls as Mmg provides a Fortran interface. To use the calving algorithm Mmg (version 5.5.4 or later) must be compiled with Elmer.

The major limitation of using Mmg is its requirement to run in series. This vastly reduces the scalability of the calving algorithm. However, large-scale testing has shown this is not an insurmountable problem at present, as solving the full-Stokes equations is still the major computational requirement for any simulation. A new parallel version of Mmg known as ParMmg has been released. Currently it is very basic and does not allow the use of a level set function. However, it is being actively developed and should be a priority to implement when possible. The current reliance on a serial version on Mmg is a major hindrance when scaling up the number of processes used. The use of ParMmg to produce a fully parallel calving algorithm is described in Section 4.7.

Further alterations in the calving algorithm centre around compensating for these two fundamental changes, the use of a fully unstructured mesh and the removal necessity for the terminus to remain projectable. The use of a full unstructured 3D mesh means the boundaries are not linear in the z axis. Instead, there are undulations between elements dependent on the distance from the idealised boundary position and fineness of the mesh. Secondly, projectability of the calving front is no longer required. The adjusted calving algorithm with routines within the new solver `CalvingRemeshMMG.F90` is shown in Fig. 4.16 and 4.17.

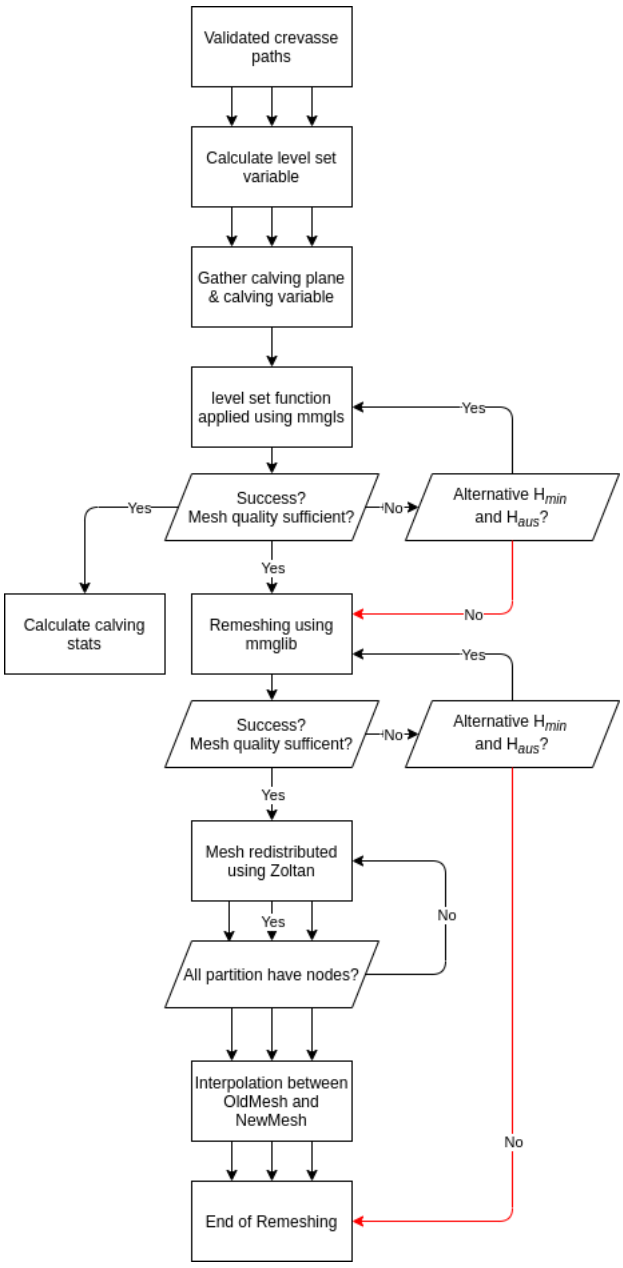

Figure 15: Diagram outlining the steps involved in the calving algorithm. A visual representation of an example simulation can be seen in Fig. 4.24. Red arrows indicate paths where calving is suppressed. Single arrows indicate stages which occur in serial and three arrows indicate stages that can occur in parallel.

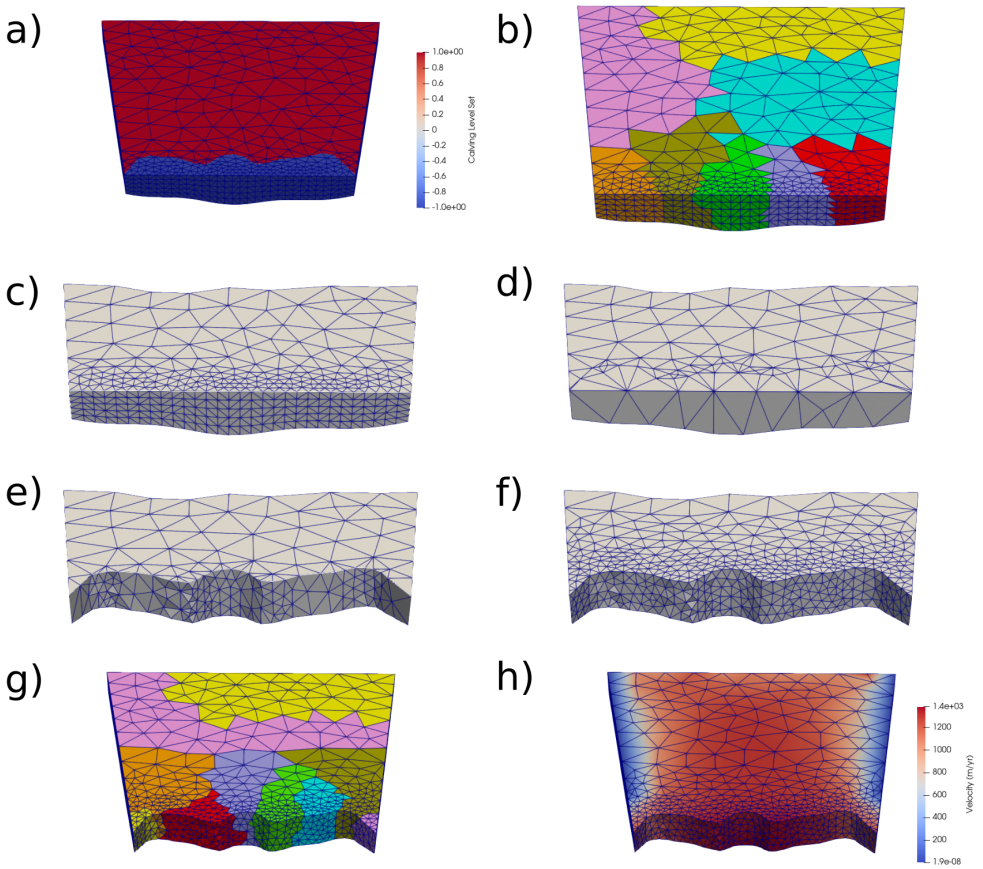

Figure 16: A visualisation of the steps involved during remeshing in the calving algorithm for a simulation run on 8 cores. a) The 3D glacier mesh showing the calving level set variable defining the predicted calving. The reduced range of the calving level set variable is just to clearly show the predicted calving. b) The same distributed mesh showing the 8 partitions. c) The gathered mesh on one core. The upstream size is defined by the user defined remeshing distance from the new calving front. d) The first remeshing step where element edges are aligned along the new calving front as predicted by the calving level set variable shown in a). e) After the first remeshing step the nodes with negative calving level set values are cut (e.g. glacier ice that would calve as icerbergs is removed). f) The second remeshing step where the mesh quality is improved. The remeshing is anisotropic based on user defined variables. g) After remeshing the mesh is rebalanced so each partition has a roughly equivalent partition. h) The variables are interpolated to the new mesh in parallel and the old mesh is deallocated from memory.

4.1 Remeshing algorithm

The new solver, `CalvingRemeshMMG.F90` was created to handle the conversion of the Elmer mesh into Mmg formats and call Mmg libraries. This solver is run following the validation of crevasse paths. Currently, remeshing is completed in two separate steps. The first step realigns element edges along the zero level set contour using a module called `mmgls`. This will be referred to as implementing the level set variable. The second stage is complete anisotropic remeshing to improve the mesh quality using the module named `mmglib`. The full remeshing algorithm is shown in Fig. 4.16 and visualized in Fig. 4.17. To reduce the computational requirements, only the area within a user defined distance from the terminus is remeshed. Unfortunately, Mmg must run in serial so the Elmer mesh partitions must first be gathered onto one process (Fig. 4.17b-c). Essentially, for both remeshing stages, the nodes and elements on the upstream partition boundary of the gathered mesh are fixed and cannot be altered. This means that when converted back into the Elmer mesh format, the new mesh still has the same partition boundaries with the upstream parts of the mesh which have not been altered.

Mmg Variables

The output mesh is controlled by a series of user defined parameters set during the level set implementation and remeshing steps. The minimum element edge size is controlled by H_{min}, whilst H_{max} is the maximum edge size of any element. The maximum difference in size between the edge lengths of any element is controlled by H_{grad} where

$$\frac{1}{H_{grad}} \leq \frac{e_1}{e_2} \leq H_{grad}, \tag{11}$$

where e_1 and e_2 are the two edge lengths. Finally, the Hausdorff distance, H_{ausd}, is the maximum distance from the reconstructed boundary to the idealised boundary. This controls the mesh refinement at the calving boundary. Remeshing is anisotropic in order to improve computational efficiency. The minimum and maximum metrics in the x and y planes are user defined with smaller elements at the calving front and larger elements upstream. The vertical metric can also be defined by the user. Finally, a minimum element quality can be set for both `mmgls` and `mmglib`. If an output mesh has a minimum mesh quality below the threshold it is treated in the same way as

if remeshing fails. Using the anisotropic element quality function in Mmg, tetrahedral element quality is defined as

$$Q = \alpha \frac{V_M}{(\sqrt{\sigma_{e_i} l_i^2})^{\frac{3}{2}}}, \tag{12}$$

where Q is the element quality, α is the normalisation coefficient ($12\sqrt{3}$), V_M is the volume of the mesh metric, e_i is the element edges, l_i is the length of each edge.

Unlike the other input parameters, H_{ausd} and H_{min}, can have multiple fall back options that can be assigned in the Elmer input file for both `mmgls` and `mmglib` (Fig. 4.16). These are usually in decreasing order so to increase mesh fineness with each iteration. This allows the model to retry the Mmg package if it fails with increased fineness. This often resolves any remeshing issues and allows calving to occur. The output mesh quality is assessed prior to conversion to the Elmer mesh format. A minimum mesh quality can be specified by the user and if an element is below this threshold the mesh is treated as though the particular remeshing stage failed (Fig. 4.16).

Remeshing stage 1: implementation of the level set variable

The level set surface is implemented using the `mmgls` module. Once the mesh has been gathered on one processor, `mmgls` is called to apply the level set function. Rather than cutting the mesh through individual nodes, the level set function cuts the mesh across the zero contour. Cutting the mesh through element boundaries produces poor quality elements, meaning further remeshing is required (Fig. 4.17). If `mmgls` fails, it is reattempted using the second set of user defined input parameters. If `mmgls` fails on all the user defined inputs, remeshing occurs but calving does not take place (Fig. 4.16).

Although the remeshing method implemented here is in two stages, recent advances in the Mmg package have allowed the level set variables and remeshing to be applied in one step. Unfortunately, remeshing occurs on both calved and non-calved areas when using a one-step method. This makes it more computationally expensive than a two-step method. Another advantage of the two-step method is increased robustness with model breakdown far less common. Occasionally, there is a problem with `mmgls`, but if the output mesh can be saved this can be recovered by secondary remeshing

after post-processing. Furthermore, if remeshing is completed in one step then an error in the level set function cannot be recovered. Input parameters can also be tailored for `mmgls` and `mmglib` separately producing a more robust method. Consequently, remeshing is implemented in two stages.

The mesh is converted to the Elmer format once a mesh of sufficient quality has been produced. Further mesh quality checks are made to ensure boundary information, neighbour information and other important mesh information is maintained. Any calved icebergs are removed (i.e., nodes with a negative calving level set value) and post-processing checks ensure that there are no unattached parts of the mesh.

Remeshing stage 2: Improving mesh quality

Following the first stage of remeshing, the new intermediate mesh is read back into Mmg. Anisotropic remeshing is done using `mmglib`. Along with the standard Mmg input variables, a mesh metric is provided to allow mesh density to vary. A specific user function, `USF_GlacierMeshMetric.F90`, has been created to provide an input metric for the mesh based on a minimum desired mesh density at the terminus to the maximum desired element size further upstream. The user defines minimum and maximum distance along with a minimum and maximum element edge length. The element size then increases linearly between the minimum and maximum distance. Beyond these distances the target edge length is set to the minimum or maximum as specified by the user. `USF_GlacierMeshMetric.F90` additionally allows a constant vertical resolution to be set. The anisotropic remeshing not only reduces the computational cost of the new mesh but also reduces the cost as the glacier advances or retreats as element size is dependent on the distance to the front and not previous element size.

Remeshing is always attempted even if the implementation of the calving level set variable fails (Fig. 4.16). Additionally, if no calving is predicted to occur, or is suppressed, remeshing will still take place. This is essential to maintain high mesh quality as the mesh deforms as the glacier advances. For some geometries `mmglib` is unable to produce a mesh above the minimum quality required. If this happens, remeshing is reattempted with second set of H_{ausd} and H_{min} inputs as set by the user (Fig. 4.16). If, by the final iteration of the input variables, a mesh of sufficient quality is not produced calving

does not occur for that timestep. Calving is suppressed when `mmglib` fails even if `mmgls` was successful at implementing the calving level set variable. Given that remeshing using `mmglib` must occur in serial this method is not particularly efficient. Future development should try to utilise ParMmg to allow remeshing to occur in parallel.

Following successful remeshing, the new mesh is converted back into the Elmer mesh format. As the upstream elements and nodes have been fixed in place the new mesh still fits with the unaltered upstream elements left over during the mesh gathering phase. This mesh could be used as it is, but would be very computationally expensive given the large number of elements present in one partition. The mesh must therefore be rebalanced to distribute it evenly among the processes. The rebalancing phase performs a mesh load balancing between the processors.

4.2 Rebalancing the new mesh

As the current implementation of remeshing using `mmglib` must run in serial, the gathered mesh must be redistributed using Zoltan (Devine et al., 2009). A rebalancing algorithm aims to rebalance the mesh evenly in terms of computational requirements among all active processes whilst trying to reduce parallel communication. The mesh distribution, element and node locations are fed into Zoltan which then returns a suggested distribution giving a partition number for each element. The mesh is then redistributed into partitions accordingly. Zoltan has an input value named the 'Imbalance Tolerance' which controls the relative load disparity between partitions. In massively parallel simulations such as those run on a supercomputer, Zoltan can fail to assign elements to a partition. If this is detected, rebalancing is retried with a lower imbalance tolerance. This process occurs iteratively until the imbalance tolerance is equal to one (e.g., the requirement for exact balance of the partitions). Elmer returns a fatal error if this final rebalancing iteration does not provide elements on every partition. However, this has never occurred in practice – to date one of the rebalancing iterations has provided all partitions with elements, as the imbalance tolerance becomes stricter with each iteration.

ParMetis and Zoltan Libraries

Using the inbuilt Zoltan load balancing algorithms has led to some problems on larger jobs (e.g., those that are run on 128 processors). It provided poor partition quality with many orphan nodes and orphan elements, and this increased the parallel communication required for all solvers (Fig. 4.18). This led to a noticeable increase in the time taken to solve the full-Stokes equations. High quality rebalancing is essential given the current necessity of mesh gathering for the remeshing step. In order to overcome this issue, the ability to use ParMetis for rebalancing was added. For large jobs this gave a significant improvement in partition quality but there was no noticeable difference on small local scale jobs (e.g., those using 8 processors).

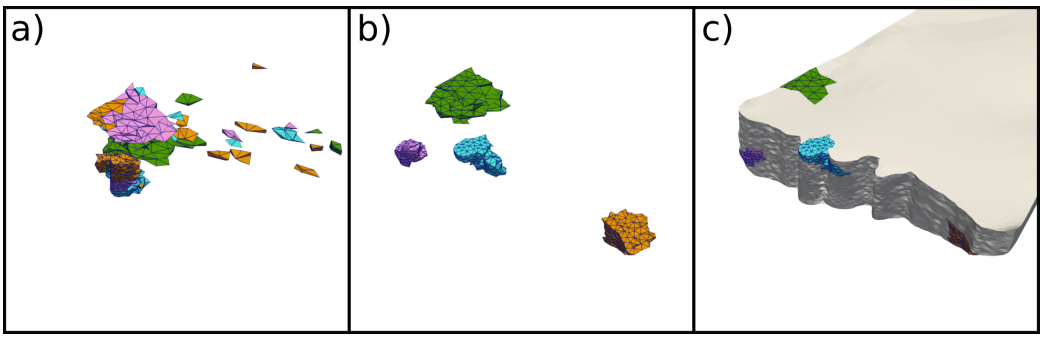

Figure 17: An example of rebalancing partition quality on a large-scale simulation of Jakobshavn Isbrae run on 128 processes. The images are after the simulation has run for 70 days, giving it time for the mesh partition quality to degrade after the initial serial partitioning prior to simulation. a) A poor quality partitioned mesh showing the first four partitions of the mesh. Note the numerous isolated elements. Rebalancing was done using Zoltan with the PHG graph package using a repartition approach. b) The same geometry as a) again showing the first four partitions but rebalanced using ParMetis. Note the intact partitions. c) The same partitions as b) but with the rest of the mesh shown in grey to give the context.

This option can be selected using the key words, "Repartition Method" in the input file with the two current options "Zoltan" or "ParMetis". Using ParMetis assumes it has been compiled with the Elmer installation, otherwise

the solver will default back to Zoltan. When using Zoltan, the keywords "Repartition Zoltan Library" and "Repartition Zoltan Graph Package" allow the user to select the library and graph package offered by Zoltan (Devine et al., 2009). The level of rebalancing undertaken can be altered using, "Repartition Approach", with the options "Partition", "Repartition" and "Refine" defining the time spent on rebalancing. When using ParMetis, the library can be selected using "Repartition ParMetis Library". Given the expense of solving the full-Stokes equations, it is very important to select a rebalancing algorithm that maintains a high-quality distributed mesh throughout the simulation.

4.3 Interpolation to the new mesh

After the rebalancing, variables from the old mesh are interpolated across the new mesh (Figs. 4.16, 4.17). This step occurs in parallel.

Several problems arise from using the old method of interpolation, which assumes the calving front is projectable along with an extruded mesh (Todd et al., 2018). Large sections have been rewritten to improve their robustness and parallel communication. Surface and bottom variables are still projected from the old mesh as both surfaces must maintain projectability. This is assumed as free-surface solvers are used on both. The free surface cannot be solved on a non-projectable boundary. However, with the possibility of complete remeshing, there are occasionally sections of the surface and bottom boundaries that are not covered by the old mesh. Nodes here are individually extrapolated. A new development was made so that the calving front variables are no longer projected on the new mesh from the old. The use of the level set variable means there are no important variables (e.g., calving variable) on the calving front boundary elements that are not recalculated from scratch in the next timestep (e.g., front advance).

The majority of the bulk elements are interpolated directly from the old mesh, but again, nodes can be extrapolated individually if there is some surface adjustment resulting from the remeshing. It is assumed that the model is not sensitive to the values of outlying variables at the terminus that would cause abnormal values in the extrapolation, since variables are recalculated at the start of the next timestep. This process purely provides a neater output at the end of the timestep. Following the interpolation, the

old mesh is replaced by the new, fully interpolated mesh.

Enforcing grounded mask

Following the interpolation, the new mesh nodes that have a grounded mask value indicating they should be grounded are manually fixed onto the base. This prevents areas rising following element splitting across the grounding zone during remeshing. This also prevents areas rising if the base-mesh surface is adjusted slightly (by the maximum of the Hausdorf distance) during remeshing. Even a seemingly insignificant movement could lead to changes in the `GroundedMask` as it is calculated at the start of each timestep, because the grounding-line problem is solved as a contact problem during the flow solution. Therefore, adjustment after remeshing prevents areas of the mesh becoming ungrounded through non-physical processes.

Enforcement of lateral margins

As the glacier advances along prescribed fjord boundaries as calculated by `CalvingGlacierAdvance3D.F90`, lateral nodes are fixed to the rails but elements can straddle across corners (Fig. 4.19). This does not cause any issues if the solver is used without remeshing, as elements are realigned with prescribed rails when the trailing node passes the corner. However, if remeshing occurs the element can be split. This will lead to nodes outside the prescribed fjord walls (Fig. 4.19). Over multiple timesteps the mesh increasingly diverges away from the rails. To prevent this, a new subroutine, `EnforceLateralMargins`, was introduced within the `CalvingRemeshMMG.F90` architecture. Every lateral node is checked to see if it lies within the fjord boundaries. If not, it is moved to the closest point on the rail. This is checked and corrected every timestep. As a result, nodes are only moved small distances because prolonged movement of nodes off the rails is prevented. Therefore, mass loss is inconsequential and element degeneracy will not occur. The solver is turned on or off using the keywords 'Fix Nodes on Rails' in the solver input file.

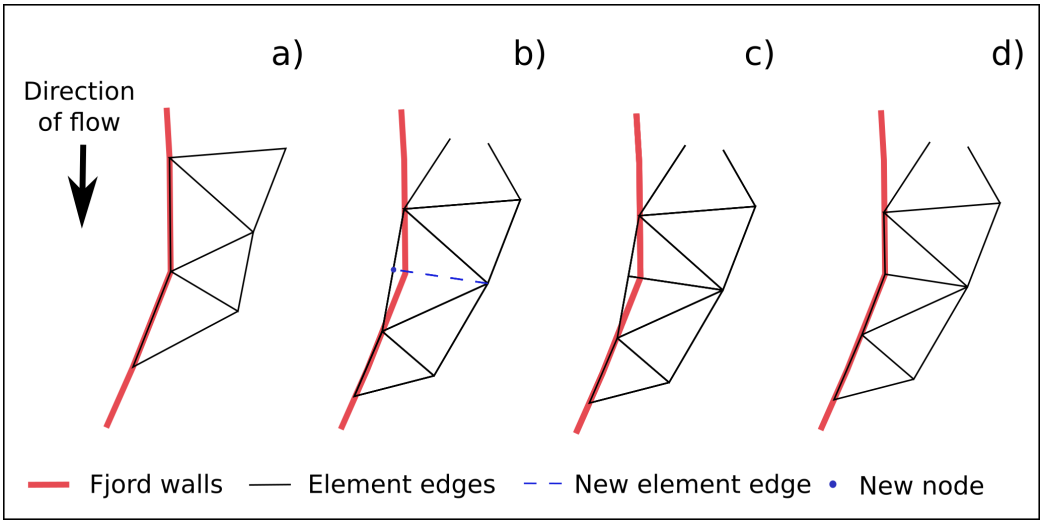

Figure 18: A 2D schematic highlighting the potential problem with remeshing coupled with the front advance routines when advancing around corners. The red line indicates the fjord rails with the black lines showing the element edges. The dashed blue line shows a potential new element edge and the blue dot is a new node produced from element splitting during the remeshing. a) A simple depiction of four elements near the fjord walls. b) The same elements after the mesh has been deformed, and the nodes deformed based on the front advance routines. Note the central element has two nodes on the rail but its edge cuts the corner. If remeshing is invoked there is the potential for a new node (blue dot), which would produce a new element edge along the dashed blue line. c) The result of remeshing with a node being produced beyond the fjord walls. d) Nodes outside the fjord walls are moved to the closest point on the rail by `EnforceLateralMargins`.

Boundary element transfer from the base boundary

Basal topography is known to be important for a multitude of glaciological processes (e.g., Goldberg et al. 2009; Gudmundsson 2013), but it can cause issues with model robustness. For example, as the glacier advances on a retrograde slope, the nodes on the base are being moved at each timestep to match the basal topography. This movement is problematic at the lateral and base boundary joint where base elements can become near vertical. The lateral and base boundary joint is not always clearly defined by a sharp

ridge. In this instance, the free surface cannot be solved once base boundary elements have a near-vertical orientation (i.e., has a normal in the z plane close to horizontal) or the base-boundary element becomes inverted. The free surface cannot be solved in this instance because of the assumption of vertical projectability. This issue is dealt with by marking base elements that have nodes that are also present in lateral or calving front elements with a normal in the z-plane close to the horizontal. These marked elements are moved to either the calving front or lateral boundary. The moving of boundary elements in this way prevents problems when solving the free surfaces.

Robustness of remeshing

There will be instances of remeshing failure given the complexity of potential geometries arising from calving at a tidewater glacier. It is not possible to prevent this for every scenario that may arise, so additional focus must be placed on the robustness of the calving algorithm to cope with remeshing failure. A major advantage of performing the level set variable implementation and anisotropic remeshing in two steps is the ability to isolate a source of potential failure. If there is an issue, remeshing can be rerun without affecting the level set variable implementation. The purpose of providing the user with the option to select multiple H_{min} and H_{aus} values is to increase remeshing robustness. If a failure occurs during the level set variable implementation or anisotropic remeshing, the process is retried with a finer input parameter. This often results in success (Fig. 4.16). Failure occurs for two reasons.

The first reason that failure occurs is that Mmg is unable to return a saved mesh. Secondly, remeshing failure can occur if the element quality does not meet the user defined minimum quality. However, there can be times when level-set implementation fails across the range of input parameters provided. Calving cannot occur in this case. Remeshing still occurs to try to improve the mesh quality. Similarly, if remeshing fails on all input parameters, calving does not occur even if the level set variable implementation has been successful. Both situations can lead to the glacier falsely advancing. This is not a major issue as calving will likely occur on the subsequent time steps. If remeshing fails over multiple timesteps the front advance routine can lead to element degradation. This makes it increasingly difficult to recover a high-quality mesh.

A final potential issue with the remeshing can result if a mesh passes through the various quality checks but has some physical imperfection or poor element quality that leads to problems in the Stokes solver. Element quality checks such as Eq. 4.12 attempt to prevent such instances but are not fool proof. Such imperfections often cause unrealistic velocity solutions that in turn can cause unrealistic calving events. An additional step in the calving algorithm has been added to check the new flow solution, which ensures that a mesh imperfection does not slip though (see Section 4.6.5).

An additional change has been made to the Elmer model to allow easier recovery if the model breaks down. This provides checkpointing for model simulations. Often a remeshing defect will persist for several timesteps prior to it breaking the model. Previously, since the mesh changes every timestep, the Elmer result file is overwritten each time, making historical recovery impossible. A separate result file will now be saved for each individual timestep with the addition of the logical keyword, 'Output File Each Timestep' in the simulation section of the solver input file.

4.4 Calving statistics output

The information of each individual iceberg that is calved is calculated (Fig. 4.16). Nodes marked with a negative level set value (those in the calved area) are looped though. Nodes that are connected (i.e., share the same element) are gathered in groups. Each of these groupings should represent an individual iceberg. The volume of the bulk elements in each group is calculated to give the volume of the respective iceberg. The maximum extent across the xy plane is also calculated for each iceberg so identification of individual icebergs when comparing to the 3D output is possible. The name of the calving statistics output can be set with the keywords, 'Calving Stats File Name'.

4.5 Saving front geometry

Similarly, to the calving statistic output, the position of the new calving front is saved. All the nodes on the new calving front that are also present on the top surface boundary are iterated over and ordered from left to right. Their x and y coordinates are then saved in an output text file at each time step. This provides a set of easily accessible data to observe the evolution of the

calving front without having to extract it from the heavily partitioned 3D output files. The name of the terminus position output file can be set with the keywords, 'Output Terminus File Name'.

4.6 Subsequent calving

Prior to the removal of the calved elements, the volume and extent of each iceberg is calculated (Fig. 4.20). This information is then used for adaptive time stepping where large calving events lead to changes in the model timestep size if domain changes lead to consequential calving.

It is well known that a large calving event can induce further calving upstream (Benn et al., 2017a; O'Leary and Christoffersen, 2013). Elmer/Ice cannot intrinsically capture rapidly cascading calving sequences because calving is only calculated at input time stepping. Developments have been made to capture secondary calving events. If the volume of the largest calved iceberg is above a user defined threshold, the non-calving solvers are turned off and the timestep is reduced (Figs. 4.20, 4.21). The iceberg volume threshold is set using the keywords, "Pause Solvers Minimum Iceberg Volume". The new timestep size is set by the keywords, "Pseudo SS dt".

Any solver not required for the calving algorithm which modifies the mesh domain is turned off. This includes any free surface, mesh update or front advance solver. To ensure this happens, mesh update variables (e.g., Mesh Update Solver 1 = ...) and free surface variables (FreeSurface Variable n) must be specified in the `CalvingRemeshMMG` solver section. The equation name of these solvers must also be given by the keywords, "Switch Off Equation n". These keywords are given to ensure that `CalvingRemeshMMG` solver can access the necessary solvers/variables to turn them off/on.

If the solvers are paused, the stress and velocity fields will be recalculated at the next timestep but the glacier geometry will not evolve. This allows for the representation of potential subsequent calving (Fig. 4.20). The maximum number of timesteps that these solvers can be paused is set by the keyword, "Calving Pause Max Steps". The default is five timesteps. Solvers are turned back on and the original timestep restored under the following conditions:

1. The volume of the largest iceberg is below the value of Pause Solvers Minimum Iceberg Volume.

2. Remeshing or level set variable implementation fails.

3. Solvers have been paused for the value given by Calving Pause Max Steps.

Importantly, the core of the model, `ElmerSolver.F90`, has been modified to allow the addition of extra timesteps during a model run. An extra timestep is added to that assigned for the model run with each timestep that the solvers are paused, and subsequent calving is checked for. This ensures the model runs for the period of time dictated in the input file rather than finishing earlier. For example, if a run is set up for 100 timesteps at 1-day intervals and the solvers are paused for 20 of those timesteps, Elmer will run for 120 timesteps thus running for the correct 100-day simulation period assigned.

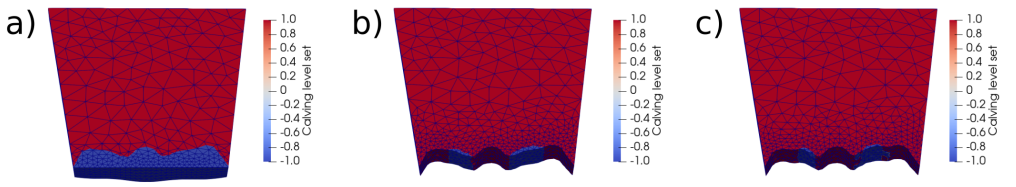

Figure 19: An example of the adaptive time stepping in the calving algorithm. a) The calving level set field for the initial time step. The iceberg volume is larger than the threshold for pausing time so the subsequent timesteps are reduced. b) The flow and stress fields are recalculated on the new geometry but there are no mesh deforming solvers active. Again, the maximum iceberg size is above the threshold so the time step remains reduced. c) The process repeats but the calving volume is below the threshold. Therefore the time stepping is returned to normal and mesh deforming will be active for the next time steps. Note the field is limited from -1 to 1 for viewing purposes only so the calving area can be seen clearly.

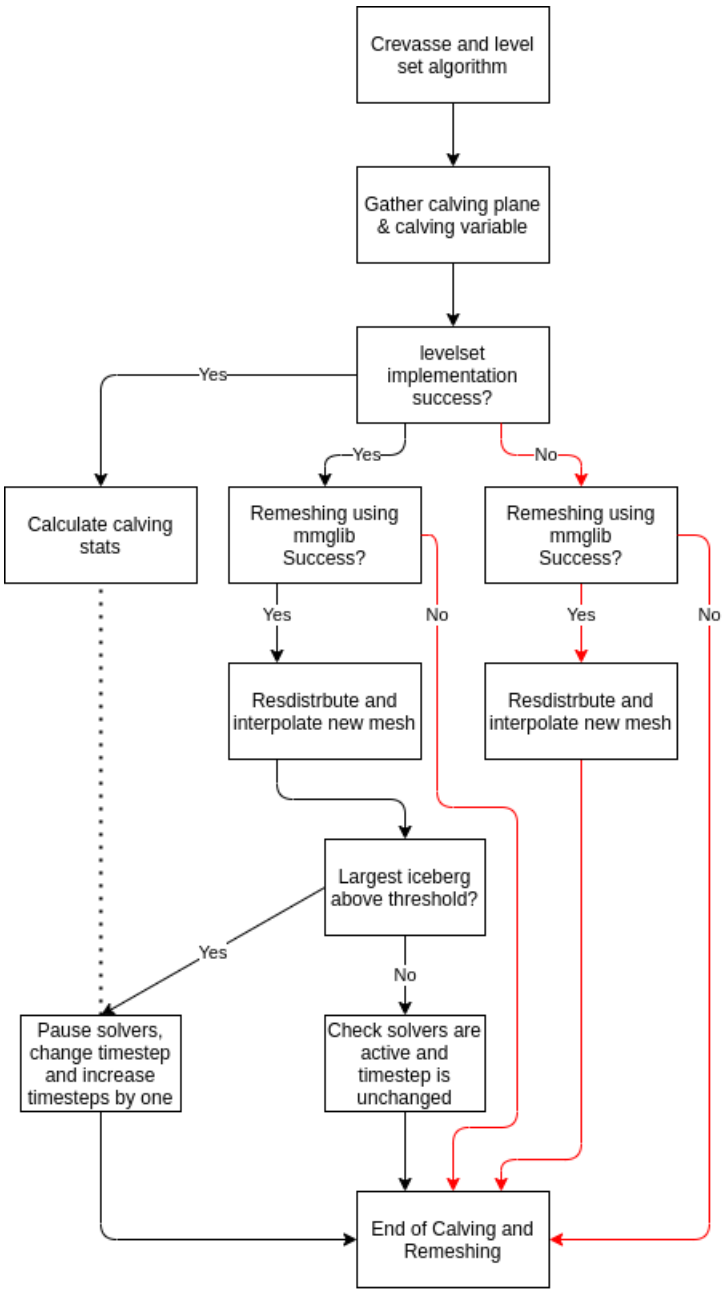

Figure 20: Diagram outlining the steps involved in the calving algorithm. The remeshing is condensed into one step with the detailed steps involved provided in Fig. 4.16. The red arrows indicate routes where calving is suppressed.

4.7 Check flow convergence

A new solver, `CheckFlowConvergenceMMG`, was written to check the full-Stokes solution. It runs after `FlowSolve` and before any subsequent solver that relies on the flow solution, such as the projection of calving (Fig. 4.24). There are three clear conditions that indicate whether the flow solution is problematic: 1) the nonlinear iterations of `FlowSolve` have not converged; 2) the maximum velocity of the solution across all nodes is above a user defined limit, and the flow solution is marked as suspect; 3) the maximum velocity value diverges significantly between time steps. The divergence tolerance can be set in the solver input file. If any of these instances arise, the treatment by the solver is the same and the flow solution is marked as problematic.

One final change was made to the legacy solver "FlowSolve" to help with the process. A simple keyword was added, 'Nonlinear System Max Norm Return', to allow the user to set a normal threshold at which the `FlowSolve` will exit the non-linear iterations and provided a logical marker identifying a non-converged state. Returning this prevents additional time being spent on a non-converging solution and prevents the solution from reaching NaN values. This should only be used in conjunction with the remeshing routines as it returns a nonsense flow solution which cannot be used.

If the flow solution is determined to be inadequate (non-converged), `CheckFlowConvergenceMMG` suppresses mesh deforming and calving solvers except the anisotropic remeshing routines. Within `CheckFlowConvergenceMMG`, mesh input variables are refined and flow solution residuals are removed, so that the velocity is calculated from scratch. The mesh input variables refined are H_{min}, H_{aus}, H_{grad}, the remeshing distance and the mesh metric inputs, if present. These refinements result in a finer mesh and an extension of the remeshing distance. The latter is extended in case the fixed elements are causing the flow-convergence issue. The amount the refinement is set by the user defined variable, 'Mesh Percentage Change'. Additionally, the model time is set back to the time at the start prior to the current, problematic, time step plus one second (Fig. 4.24). An extra time step is added to the required time steps in `ElmerSolver`. (This is a similar process to the adaptive time stepping discussed in Section 4.6.4.) Following the subsequent remeshing step the mesh quality will usually have improved enough to provide

a flow solution. As this point, the solver will unpause the mesh deforming and calving solvers and return the Mmg mesh input variables (e.g., H_{min}) to their original values.

5 Parallel Calving Algorithm

A parallel calving algorithm is currently in the development stage, and it is not used in the following chapters. However, significant effort has been spent on it producing important progress. It is therefore detailed in its current state for this section. Conceptually it follows the serial calving routine but undertakes all computationally expensive routines in parallel. In some ways this vastly simplifies the calving algorithm as calving is always implemented in parallel rather than switching between serial and parallel routines. This cuts out the need to gather and redistribute the mesh along with reducing the complexity of the additional new functionality. Routines in `Calving3D_lset.F90` remain unchanged with formation and validation of crevasses remaining in serial. However, those routines are computationally light so it is not worth upgrading to parallel.

When moving to the parallel algorithm, the gathering and serial remeshing routines in `CalvingRemeshMMG.F90` are replaced with a new parallel solver called `CalvingRemeshparMMG.F90` (Fig. 4.22). The subsequent subroutines that take place after the rebalancing to check and alter the mesh (e.g., enforcing the `GroundedMask`, boundary element transfer or enforcement of the lateral margins) are unchanged.

5.1 Mesh gathering

Given the computational cost of remeshing, even in parallel, the user has an option to define the area from the calving front to remesh. If specified, the mesh present on this area is copied and rebalanced across all active processes. If not, then the whole mesh is used, and it is assumed the mesh is balanced. This differs from the serial algorithm where the remeshing area is gathered on only one processor.

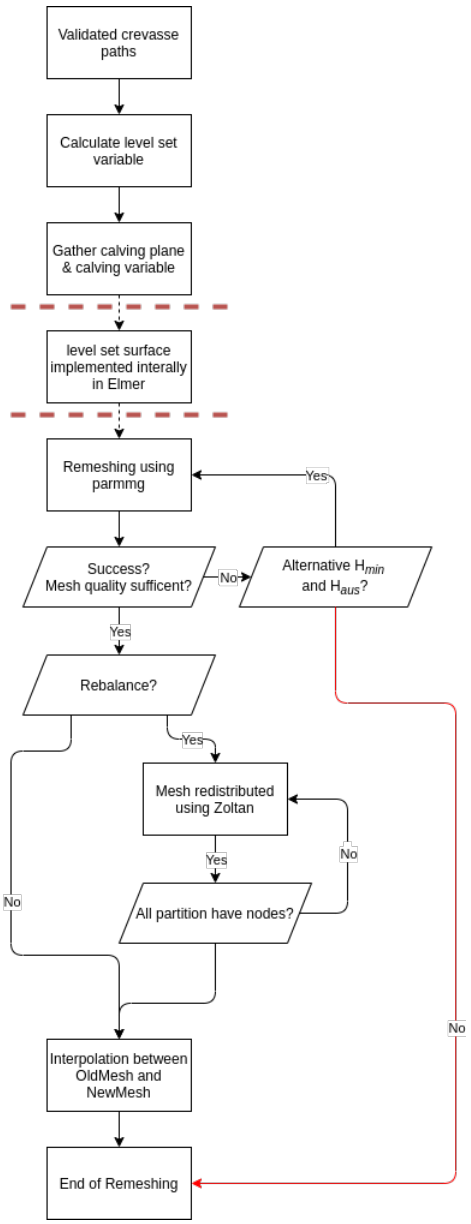

Figure 21: Diagram outlining the current parallel calving algorithm. The solid red arrow indicates when calving is suppressed. The dashed red lines and dotted arrows indicate the one stage which is yet to be complete - the implementation of the level set surface in the mesh. All stages are in parallel.

5.2 Implementation of calving

There is no parallel version of `mmgls` to implement calving in a similar way to the serial calving algorithm. Instead, a new routine is being written for Elmer. This routine will subdivide the tetrahedral elements so the edges of the daughter elements are aligned to the zero level set contour. Currently, the resultant elements can be of poor quality although a limit on the minimum element angle is set. It is assumed that the remeshing using ParMmg is robust enough with poor quality elements. This is where development focuses at the moment. Once complete, a fully-parallel calving algorithm will be available.

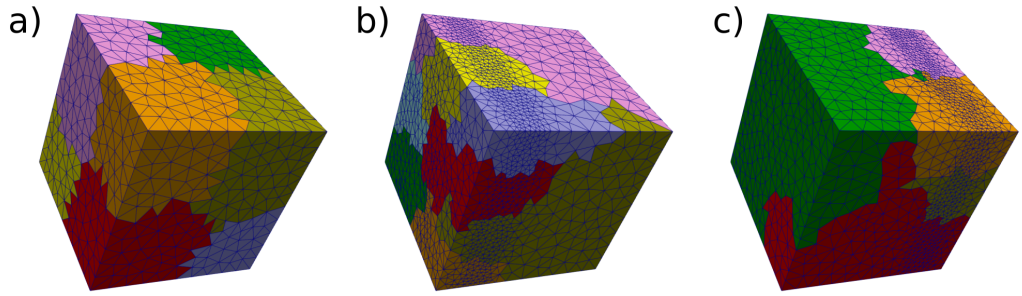

Figure 22: A simple example of parallel adaptive remeshing in Elmer on 8 cores on a cube. A simple advection is prescribed across the x axis, which is used to define the mesh metric that produces an area of fine mesh that moves at each time step. a) The cube prior to remeshing. b) After the first timestep. c) After the final time step. The different colours show the different partitions of the mesh associated with individual cores. The full simulation can be seen in S4.23 (see Appendix).

5.3 Parallel remeshing

Remeshing is implemented using ParMmg, a parallel version of Mmg via the use of its Fortran interface (Balarac et al., 2022; Dapogny et al., 2014). In order for the API to work with Elmer, some changes were made that have been accepted by the authors of Mmg. At the time of writing only the modified branch (feature/tetFromTria-API) of the program works with Elmer. Unfortunately, ParMmg is not actively developed, and is only maintained at the time of writing. This is in part why this development has not progressed beyond its current stage. Although operational and quicker than the serial

version ParMmg is still computationally expensive for each process used. Its optimisation is imperative for further uses in Elmer and Elmer/Ice.

In the work to date, a distributed mesh is given to ParMmg. ParMmg remeshes each partition in parallel. Partition boundary elements are fixed before a negotiation between neighbouring partitions. This process is iterated to improve the boundary element quality. The number of iterations can be adjusted using the keywords, "Remeshing Iterations" in the solver input. I wrote a testcase available in the official Elmer github repository (`https://github.com/ElmerCSC/elmerfem/tree/devel/fem/tests/ParallelRemesh`), which adaptively remeshes a cube in parallel based on a simple advection along the x-axis (Fig. 4.23). This represents the first successful use of adaptive parallel remeshing in Elmer and is potentially very important for general engineering uses. As a result, this development is very exciting beyond the glaciological applications.

6 An outline of a typical simulation

Putting together the new advance, calving and remeshing mechanisms, a typical simulation can be outlined (Fig. 4.24). After model initialization and set up, the timesteps follow the algorithm. First, the velocity field is solved. It is then checked by `CheckFlowConvergenceMMG` to assess it for any abnormalities. If abnormalities exist, the recovery mechanisms outlined in Sections 4.6.1 and 4.6.5 are followed.

Assuming the flow solution converges, it is used to solve the stress fields from which calving is predicted. After the calving prediction, the top and bottom free surfaces are solved using the built-in Elmer free-surface solver. The output for each surface is then fed into `MeshSolve` to adjust the mesh vertically. Prior to the vertical mesh deformation, the advance solver (`CalvingGlacierAdvance.F90`) is invoked using the flow solution to give the predicted front advance. The front advance variable is given to a second call of `MeshSolve` to deform the mesh horizontally.

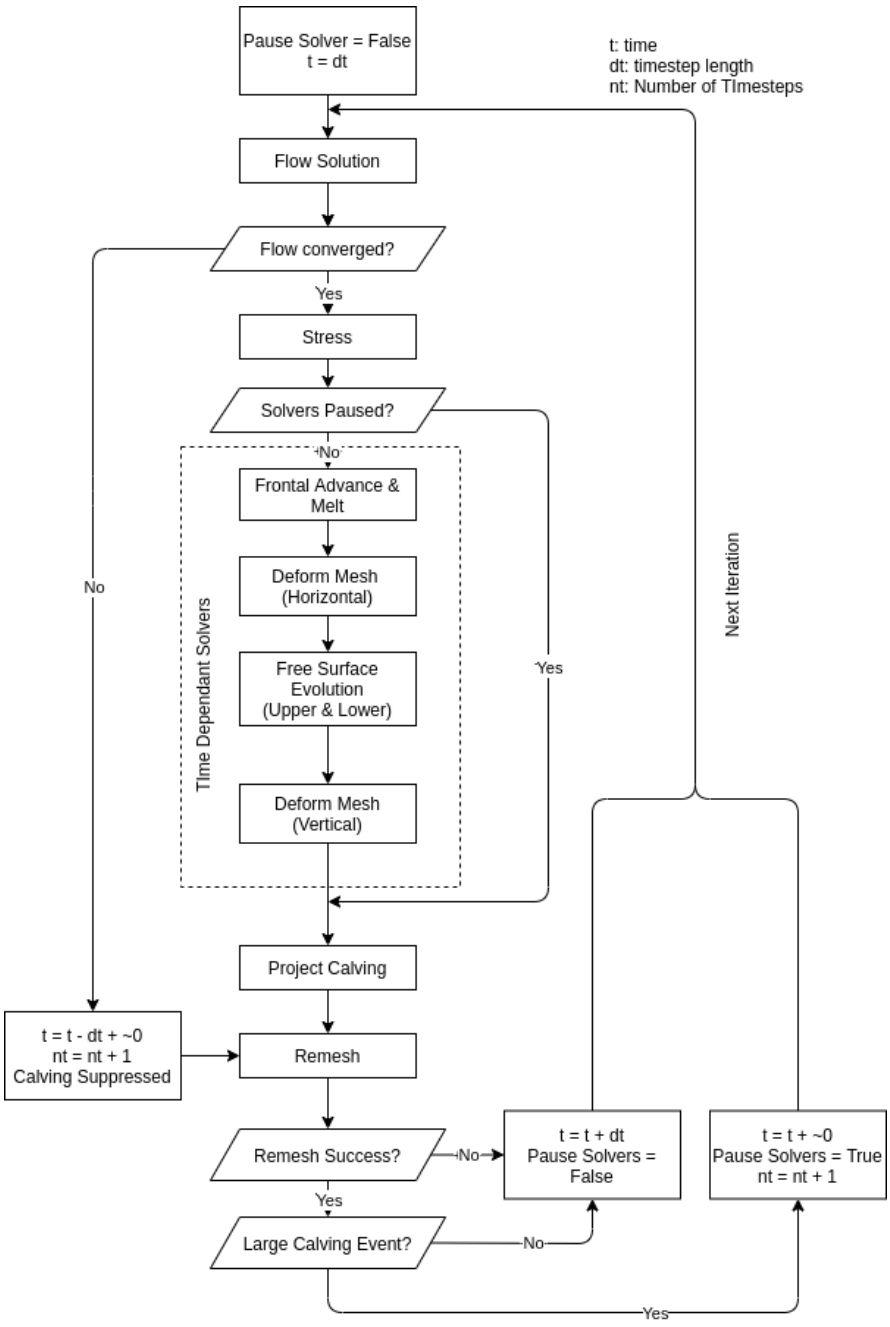

Figure 23: Diagram showing the stages involved in a typical simulation using the calving algorithm and front advance routines in tandem.

Ideally, the flow solutions would be recalculated after the front adjustment since the calculated velocities are based on a different geometry. This means that, currently, the calving prediction is based on the stress field that was calculated for a slightly different geometry. This is the approach used currently to save computational requirements, and velocities are assumed to match the geometry (Fig. 4.24). Unless time steps are extremely large for the size of domain this should not impact the results.

After the mesh deformation stage, the calving algorithm is called. Calving is suppressed and the model moves onto the next time step if remeshing is unsuccessful at any stage. If any solvers were previously paused, they are turned back on and the timestep is checked to make sure it is the original input. If remeshing is successful but only insignificant calving occurs the timestep is not altered. However, if an iceberg calves above the user defined threshold, mesh deforming solvers are paused, the timestep is reduced and an additional timestep is added to the model to allow the simulation to run for required time (Fig. 4.24).

7 Summary of model capabilities and potential

Putting together the new calving algorithm along with the upgrades in the calving projection gives us a new model with unparalleled capabilities of simulating calving over a 3D continuum. The advancements of the model include:

1. Unlimited advance or retreat now possible to simulate in 3D.
2. Unrestricted 3D calving geometries can be utilised by the model.
3. Any calving law could be implemented
4. Features or variables can be advected as part of the mesh.
5. Any 3D melt field can be applied to the glacier front.

7.1 Unlimited retreat and advance

The ability to model unrestricted retreat and project advance in 3D is a major step forward for simulating the dynamics of calving glaciers. Previous

3D calving models have been limited by technical hurdles that prevented retreat or advance beyond a certain domain (Todd et al., 2018). In the most extreme warming experiments at modelling experiments at Store Glacier the simulated retreat caused model breakdown (Todd et al., 2019). The model breakdown prevented a full comparison of calving dynamics through the perturbation of the input variables.

A second important opportunity that is now available with unlimited retreat and advance is the ability to model any glacier or ice shelf in the world. The Todd calving model has only been in applied at Store Glacier in its various forms (Cook et al., 2022; Todd et al., 2018, 2019). The model was limited to stable glaciers that do not undergo large seasonal or interannual variability. The new calving algorithm has now been applied at Jakobshavn Isbrae (Chapter 6). This large outlet glacier undergoes kilometres of retreat and advance on seasonal cycles (Joughin et al., 2020). The application of the algorithm to ice shelves should also be feasible. It would, however, probably require a new or altered calving law to simulate ice-shelf fracturing.

7.2 Unrestricted calving

The ability to calve unrestricted geometries of icebergs both in the horizontal and vertical planes should be treated as a distinct feature of this calving algorithm. This means any potential configuration of calving or front geometry is possible. Again, this ability opens up the possibility to model any complex scenario or situation seen in the real world. For example, glaciers with complex front geometries such as Bowdoin Glacier (Kangerluarsuup Sermia) (Van Dongen et al., 2019) or large fan-shaped ice tongues which are non-projectable can now be modelled.

The new calving algorithm also offers the possibility to create unrestricted synthetic calving geometries to see how the glacier dynamics respond to forced calving events. In the same vein, other calving laws could be implemented. Their implementation would not be restricted by possible calving predictions, no matter how complex the resulting geometry.

7.3 Use of other calving laws

The calving algorithm is not restricted to the crevasse depth law implementation outlined above. Any calving law could be implemented through the production a level set variable or signed distance variable that is given in the calving algorithm. Despite the relative ease by which a new calving law could be implemented, only the crevasse depth calving law has been used up to now. This is because it is currently the only possible option based on physical processes (Benn et al., 2017b). Other popular calving laws are based on calving rates as opposed to calving position and they could be used in conjunction with this calving algorithm. The ease with which another calving law could be implemented in the future proves the algorithm should advance calving law theory. The calving algorithm provides an easily accessible framework for which various calving laws could be compared in 3D.

More likely, the alterations or improvements will need to be added to the crevasse depth law as coupled modelling of tidewater glaciers advances (Cook et al., 2023). Some known problems with the crevasse depth law, such as ice history or stress concentrations, have yet to be overcome (Cook et al., 2020; Todd et al., 2018, 2019). However, this new algorithm provides the best framework to approach these problems as 3D modelling is no longer restricted by technical hurdles. As such we can now focus on improving the calving laws along with assessing which missing processes are important in calving prediction.

7.4 Feature advection

The advances in remeshing techniques allow complete anisotropic remeshing of a particular glacier part or complete glacier. This allows mesh quality to maintained even if nodes are moved in a Lagrangian manner. Some issues remain related to element degeneracy at the lateral margins, but this does not apply to ice shelves such as Thwaites which are laterally unconstricted (Scambos et al., 2017). There is very exciting potential for use of the remeshing techniques to advect variables such as damage downstream in order to better predict calving, especially on ungrounded ice sheets (Cook et al., 2023). Unfortunately given the time constraints of the PhD it remains an unachieved goal to experiment using a basic Lagrangian advection method in 3D similar to that implemented in 2D studies (Berg and Bassis, 2022). The

basic building blocks are there but further work is needed to experiment with an altered crevasse depth law with built-in ice history.

7.5 Use for melt simulations

Beyond the calving component of the new algorithm, the ability to model melt effects on glacier dynamics in 3D alone is very exciting. The limited availability of full-Stokes glacier models mean the effects of 3D melt fields on glacier dynamics is rarely researched. The ability of the algorithm to have a non-projectable front means any melt field could be applied for any length of time without model breakdown.

More commonly, the importance of submarine melt is often associated with its ascertained ability to increase calving (O'Leary and Christoffersen, 2013). Melt undercutting from buoyant plumes is often connected to full terminus retreat through calving, especially at the lateral margins (Cowton et al., 2019). The importance of submarine plume melt undercutting on calving is only possible in a 3D calving model because of the importance of terminus geometry. Previous limitations on glacier retreat and front geometry are overcome in this algorithm removing technical hurdles preventing such studies being carried out on realistic domains (Cowton et al., 2019). On its own, the calving algorithm is limited in applying a set melt field to the terminus. Future work should focus on coupling the glacier model, with the calving algorithm, with ocean/plume models (Cook et al., 2023).

Future coupling

The most advanced method of calculating frontal melt in Elmer is the coupled hydrology model developed by Samuel Cook (Cook et al., 2020). Cook's model includes a 1D plume model. The Todd (2016) calving algorithm was often used in conjunction with internal 1D plume model or field measurements. Similar to the current situation, there was no standardised method in Elmer to produce front melt for these previous studies. However, recent developments have allowed for the full coupling of a hydrology and plume model with the Todd calving model (Cook et al., 2022). Significant further development would be required to couple this with the new calving algorithm.

Future coupling work in Elmer should focus not just on hydrology but also

fjord circulation (Cook et al., 2023). The lack of coupling of fjord models with glacier models means there are often large uncertainties when applying melt fields to glacier models. Melt profiles are often derived from buoyant plume theory (Slater et al., 2017), but the lack of 3D fjord modelling neglects horizontal flow across the front of the glacier. Coupling should aim to be with a high-resolution fjord model such as MITgcm (Cook et al., 2023). Although computationally expensive, it seems futile to solve the glacier dynamics in detail but neglect the same detail with the fjord model. However, many issues such as congruent time-step sizing would need to be resolved.

7.6 Algorithm efficiency and computational cost

The computational cost of the calving algorithm is relatively low in comparison to the expense of solving the full-Stokes equations. Issues surrounding the algorithm are more often related to robustness rather than computational cost. The development of the parallel remeshing algorithm would further increase computational efficiency and dramatically increase the scalability. Although the computational cost is low relative to the solving of glacier flow, Elmer/Ice is a mature model that has been shown to scale-up well to at least a thousand cores (Gagliardini et al., 2013). Given the serial nature of the remeshing part of the calving algorithm, the algorithm does not scale well. For large simulations, such as the examples in Chapter 3, it has often been run on 128 processors with the majority of the computational time still taken up by the Stokes-equation solver. Simulations like this would benefit from the parallel calving algorithm, with potential gains of up to 10% possible. It is still unclear when the parallel algorithm will be fully operational given the dependency on ParMmg development. Importantly, the Elmer calving model remains computationally light compared to HiDEM and fills the gap between models based on first principles and the widely used 2D SSA-style models.

Bibliography

Alciatore, D. G. and Miranda, R. A winding number and point-in-polygon algorithm, 1995. URL http://www.engr.colostate.edu/$\sim$dga/dga/papers/point_in_polygon.pdf.

Åström, J. A., Riikilä, T. I., Tallinen, T., Zwinger, T., Benn, D., Moore, J. C., and Timonen, J. A particle based simulation model for glacier dynamics. *Cryosphere*, 7(5):1591–1602, 2013.

Aström, J. A., Vallot, D., Schäfer, M., Welty, E. Z., O'Neel, S., Bartholomaus, T. C., Liu, Y., Riikilä, T. I., Zwinger, T., Timonen, J., and Moore, J. C. Termini of calving glaciers as self-organized critical systems. *Nature Geoscience*, 7(12):874–878, 2014.

Balarac, G., Basile, F., Bénard, P., Bordeu, F., Chapelier, J.-B., Cirrottola, L., Caumon, G., Dapogny, C., Frey, P., Froehly, A., Ghigliotti, G., Laraufie, R., Lartigue, G., Legentil, C., Mercier, R., Moureau, V., Nardoni, C., Pertant, S., and Zakari, M. Tetrahedral remeshing in the context of large-scale numerical simulation and high performance computing. *MathematicS In Action*, 11(1):129–164, 2022.

Benn, D. I., Hulton, N. R., and Mottram, R. H. 'Calving laws', 'sliding laws' and the stability of tidewater glaciers. In *Annals of Glaciology*, volume 46, pages 123–130. Cambridge University Press, 2007. doi: 10.3189/172756407782871161. URL https://www.cambridge.org/core.

Benn, D. I., Aström, J., Zwinger, T., Todd, J., Nick, F. M., Cook, S., Hulton, N. R., and Luckman, A. Melt-under-cutting and buoyancy-driven calving from tidewater glaciers: New insights from discrete element and continuum model simulations. *Journal of Glaciology*, 63(240):691–702, 2017a.

Benn, D. I., Cowton, T., Todd, J., and Luckman, A. Glacier Calving in Greenland, 2017b. ISSN 21986061.

Berg, B. and Bassis, J. Crevasse advection increases glacier calving. *Journal of Glaciology*, 68(271):977–986, 2022.

Cook, S. J., Christoffersen, P., Todd, J., Slater, D., and Chauché, N. Coupled modelling of subglacial hydrology and calving-front melting at Store Glacier, West Greenland. *Cryosphere*, 14(3):905–924, 2020.

Cook, S. J., Christoffersen, P., and Todd, J. A fully-coupled 3D model of a large Greenlandic outlet glacier with evolving subglacial hydrology, frontal plume melting and calving. *Journal of Glaciology*, 68(269):486–502, 2022.

Cook, S. J., Christoffersen, P., and Wheel, I. Coupled 3-D full-Stokes modelling of tidewater glaciers. *Annals of Glaciology*, pages 1–4, 2023.

Cowton, T. R., Todd, J. A., and Benn, D. I. Sensitivity of Tidewater Glaciers to Submarine Melting Governed by Plume Locations. *Geophysical Research Letters*, 46(20):11219–11227, 2019.

Dapogny, C., Dobrzynski, C., and Frey, P. Three-dimensional adaptive domain remeshing, implicit domain meshing, and applications to free and moving boundary problems. *Journal of Computational Physics*, 262: 358–378, 2014.

Devine, K. D., Boman, E. G., Riesen, L. A., Catalyurek, U. V., and Chevalier, C. Getting Started with Zoltan : a Short Tutorial. Technical report, 2009. URL `http://www.cs.sandia.gov/Zoltan/ug_html`.

Durand, G., Gagliardini, O., De Fleurian, B., Zwinger, T., and Le Meur, E. Marine ice sheet dynamics: Hysteresis and neutral equilibrium. *Journal of Geophysical Research: Solid Earth*, 114(3), 2009.

Gagliardini, O., Zwinger, T., Gillet-Chaulet, F., Durand, G., Favier, L., De Fleurian, B., Greve, R., Malinen, M., Martín, C., Räback, 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(4):1299–1318, 2013.

Goldberg, D., Holland, D. M., and Schoof, C. Grounding line movement and ice shelf buttressing in marine ice sheets. *Journal of Geophysical Research: Earth Surface*, 114(4):1–23, 2009.

Gudmundsson, G. H. Ice-shelf buttressing and the stability of marine ice sheets. *Cryosphere*, 7(2):647–655, 2013.

Joughin, I., E. Shean, D., E. Smith, B., and Floricioiu, D. A decade of variability on Jakobshavn Isbræ: Ocean temperatures pace speed through influence on mélange rigidity. *Cryosphere*, 14(1):211–227, 2020.

Nick, F. M., Van Der Veen, C. J., Vieli, A., and Benn, D. I. A physically based calving model applied to marine outlet glaciers and implications for the glacier dynamics. *Journal of Glaciology*, 56(199):781–794, 2010.

Nye, J. F. The distribution of stress and velocity in glaciers and ice-sheets. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 239(1216):113–133, 1957.

O'Leary, M. and Christoffersen, P. Calving on tidewater glaciers amplified by submarine frontal melting. *Cryosphere*, 7(1):119–128, 2013.

Osher, S. and Fedkiw, R. P. Level Set Methods: An Overview and Some Recent Results. *Journal of Computational Physics*, 169(2):463–502, 2001.

Scambos, T. A., Bell, R. E., Alley, R. B., Anandakrishnan, S., Bromwich, D. H., Brunt, K., Christianson, K., Creyts, T., Das, S. B., DeConto, R., Dutrieux, P., Fricker, H. A., Holland, D., MacGregor, J., Medley, B., Nicolas, J. P., Pollard, D., Siegfried, M. R., Smith, A. M., Steig, E. J., Trusel, L. D., Vaughan, D. G., and Yager, P. L. How much, how fast?: A science review and outlook for research on the instability of Antarctica's Thwaites Glacier in the 21st century. *Global and Planetary Change*, 153: 16–34, 2017.

Sethian, J. A. *Level set methods and fast marching methods*. Cambridge University Press, 2 edition, 1999.

Slater, D. and Benn, D. A crevasse-depth calving law accounting for submarine melt. In *EGUGA*, 2022. URL `https://meetingorganizer.copernicus.org/EGU22/EGU22-7731.htmlhttps://meetingorganizer.copernicus.org/`

EGU22/EGU22-6767.html%0Ahttps://meetingorganizer.copernicus.org/ISMC2021/ISMC2021-70.html%0Ahttps://meetingorganizer.copernicus.org/EGU21/EGU21-1180.html.

Slater, D., Nienow, P., Sole, A., Cowton, T., Mottram, R., Langen, P., and Mair, D. Spatially distributed runoff at the grounding line of a large Greenlandic tidewater glacier inferred from plume modelling. *Journal of Glaciology*, 63(238):309–323, 2017.

Todd, J. *A 3D full Stokes calving model for Store Glacier, West Greenland*. PhD thesis, University of Cambridge, 2016.

Todd, J., Christoffersen, P., Zwinger, T., Råback, P., Chauché, N., Benn, D., Luckman, A., Ryan, J., Toberg, N., Slater, D., and Hubbard, A. A Full-Stokes 3-D Calving Model Applied to a Large Greenlandic Glacier. *Journal of Geophysical Research: Earth Surface*, 123(3):410–432, 2018.

Todd, J., Christoffersen, P., Zwinger, T., Räback, P., and Benn, D. I. Sensitivity of a calving glacier to ice-ocean interactions under climate change: New insights from a 3-d full-stokes model. *Cryosphere*, 13(6): 1681–1694, 2019.

Van Der Veen, C. J. Fracture mechanics approach to penetration of bottom crevasses on glaciers. *Cold Regions Science and Technology*, 27(3):213–223, 1998.

van Dongen, E. *Monitoring and modelling the calving behaviour of Bowdoin Glacier, Northwest Greenland*. PhD thesis, ETh Zurich, 2021.

Van Dongen, E., Jouvet, G., Walter, A., Todd, J., Zwinger, T., Asaji, I., Sugiyama, S., Walter, F., and Funk, M. Tides modulate crevasse opening prior to a major calving event at Bowdoin Glacier, Northwest Greenland. *Journal of Glaciology*, 66(255):113–123, 2019.

van Dongen, E. C., Åström, J. A., Jouvet, G., Todd, J., Benn, D. I., and Funk, M. Numerical Modeling Shows Increased Fracturing Due to Melt-Undercutting Prior to Major Calving at Bowdoin Glacier. *Frontiers in Earth Science*, 8(July):1–12, 2020.