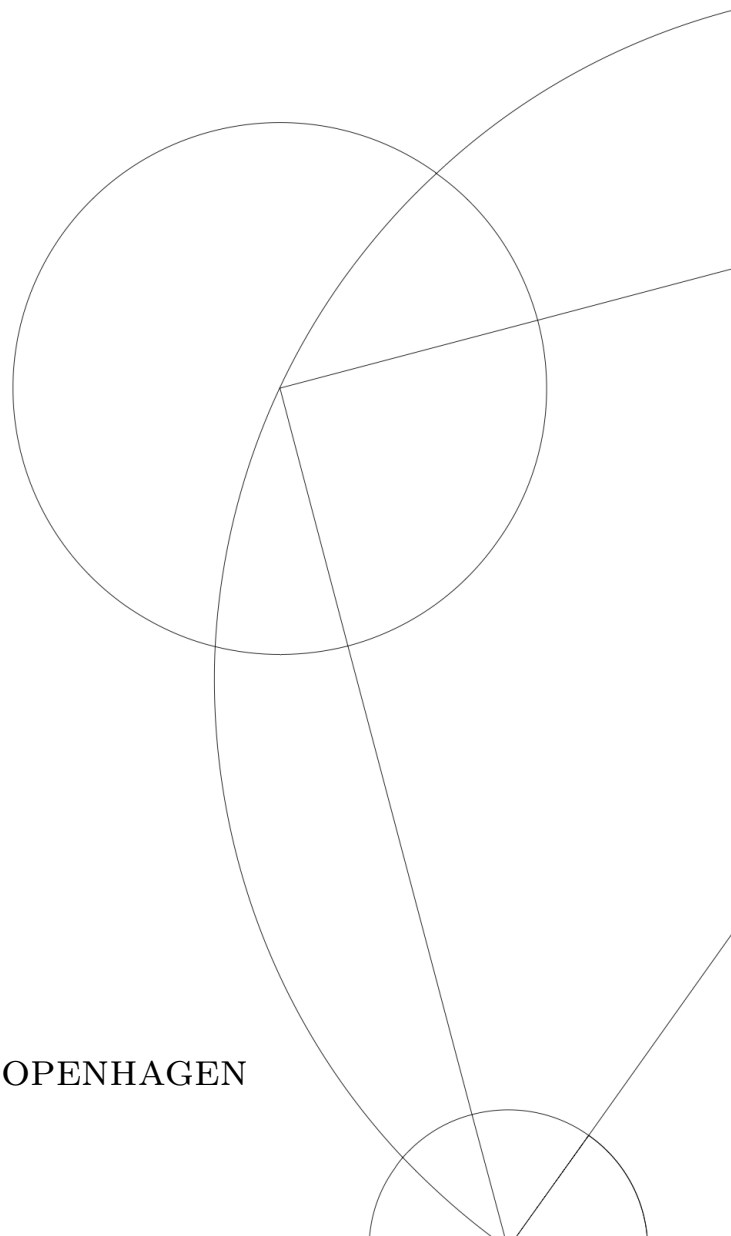
A Marine Outlet Glacier Model

A Model Based on the Perfect Plastic Approximation

Written by *Jeppe Jon Cederholm and Laura Lund Rysager*

Supervised by
Christine Hvidberg



University of Copenhagen

UNIVERSITY OF
COPENHAGEN

Name of Institute:	NBI
Name of Department:	Center of Ice and Climate
Authors:	Jeppe Jon Cederholm and Laura Lund Rysager
Email:	rlx267@alumni.ku.dk and qmd636@alumni.ku.dk

Title and subtitle:	A Marine Outlet Glacier Model - A Model Based on the Perfect Plastic Approximation
Supervisor:	Christine Hvidberg
Handed in:	16.06.2021
Defended:	23.06.2021

Abstract

In this project the perfect plastic model is applied to model six of the largest marine outlet glaciers in Greenland. In this simple model there is one tuneable parameter which is the yield stress, τ_y, that controls the material strength of the ice and determines the profile of the glacier.

First the model has been tuned to match the observed surface elevation profile by varying the value of τ_y. To better match the surface, two separate values of the driving stress have been used for each glacier. The model can evolve in time by varying the inland ice thickness or the calving front position. After determining the optimal value of driving stress the model is used to investigate the time evolution of the glacier under the present climate conditions. In this project the surface mass balance is used as the only climate parameter.

The time evolution of the glaciers is being analyzed to see if the glaciers are retreating, advancing or stable under present conditions. It turns out that all except for one glacier, are retreating. From the stable point, where the glacier is neither retreating or advancing, an estimate of ice mass loss for each glacier is calculated and compared to present day.

Contents

1 Introduction

It is known that the sea level is rising and except for thermal expansion the largest contribution to the sea level rise comes from the ice mass loss in Greenland (*Mottram et al. (2019); Shepherd et al. (2020)*)[1]. Here the greatest mass loss stems from marine outlet glaciers. Therefore, it is important to have a model that can estimate the ice mass loss. The dynamical systems of these glaciers are very complex therefore simple models might be preferred to give an overview and a rough estimate of ice mass loss at the marine outlets. All this is crucial to give an estimation of the sea level rise.

In this project an estimate of the ice mass loss from six of the largest glaciers in Greenland is found. The six glaciers are: The Helheim Glacier, Jakobshavn Isbrae, Upernavik 1, Upernavik 2, Nioghalvfjerdsfjorden, and Zachariae Isstrom. Here Upernavik 1 is the northernmost of the glacier at Upernavik and Upernavik 2 is the glacier terminating just south of the first glacier from Upernavik. These are six of the biggest outlets in Greenland and a significant source of the total drainage of ice from Greenland *Hvidberg (2021)*. These are also glaciers without extensive floating ice tongues.

The perfect plastic approximation is used to make a model of the surface elevation. This model is simple and only requires knowledge of either the surface elevation of one point in the middle of the glacier, or the length from this point to the terminus position. The limitations with this model is that it does not take flow or accumulation/ablation into account and hence assumes a steady state solution. Another model that can be applied to model the profile of glaciers, which is a bit more complex is the Vialov's profile, which comes from Glen's flow law and can be found in *Cuffey and Paterson (2010) (p.388)*. This is based on stresses within the ice, and when combined with the equation for mass continuity $\frac{\partial H}{\partial t} = a - \frac{\partial q_x}{\partial x}$, where $q_i = u_i \cdot H$, gives a model that is not necessary in steady state. This model does not use the terminus position and is therefore best on the interior parts of the ice and not the coastal areas.

The profiles of the glaciers are found and from this the time evolution of each glacier is predicted. This uses Glen's flow law, to make a non-steady state solution, and the current surface mass balance (SMB) (*Ultee and Bassis (2020b)*). The time evolution gives the retreat or advance at every possible terminus position of the glacier with the present surface mass balance as the main parameter. This is applied to find the time evolution of the terminus position and then run the model to see at what terminus position the glacier reaches a stable point given the current SMB. When the steady state is found, the ice mass loss is calculated.

This project contains a section of theory where the perfect plastic model, the height at terminus and the time evolution is derived. Then there is a section concerning the used data. Afterwards there is a section of how the data is implemented in our python scrips. Now the model is tested on ideal cases to see how it responds. Then there is a section where the results from applying the model on real data are presented. This is mostly done by presenting plots and calculated values in tables. Finally, there is the discussion of all the results. The method is discussed, uncertainties and ideas for further development

[1]See figure: https://eng.geus.dk/nature-and-climate/adaptation-to-climate-change/sea-level-rise (*AMAP (2017)*)

of the model is discussed as well.

2 Theory

2.1 The Perfect Plastic Approximation

The perfect plastic approximation is the approximation that internal deformations in ice, due to internal viscosity, happens on much longer timescales than deformations due to an internal collapse of the ice structure. Hence the plastic approximation is that the ice is rigid until the internal stresses in the ice equals a yield stress, τ_y, where the ice collapses as a material without viscosity (*Ultee and Bassis (2016)*).

Furthermore, we will need three more approximations. 1) The glacier is in steady state meaning the surface elevation does not change over time meaning $\frac{\partial H}{\partial t} = 0$. 2) There is no melting or sliding at the bed. 3) The thin film approximation. This means that the range of the glacier is much longer than the height, $\frac{H}{L} \ll 1$. This implies that the stresses are dominated by shear stresses which are balanced by the friction at the bed or walls. This means that $(\tau_{xx}, \tau_{zz}) \ll \tau_{xz}$. These stresses are shown on Fig.1. A consequence of the perfect plastic approximation is that the model mainly includes deformation of ice at the bed, and at the terminus position. Following a flowline, the ice can be seen as a 2D incompressible fluid in that plane. Along this plane a coordinate system is defined: $\hat{x}$ follows the flowline and is perpendicular to the gravitational acceleration, and $\hat{z}$ is parallel to the gravitational acceleration.

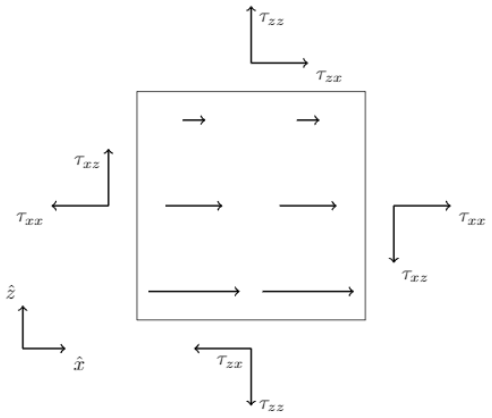


Figure 1: Direction of stresses in the perfect plastic model. Here $\tau_{xz} = \tau_{zx}$ is the stress deviators and are also the ones shown inside the square to show the size of $\frac{\partial \tau_{xz}}{\partial x}$ and $\frac{\partial \tau_{xz}}{\partial z}$.

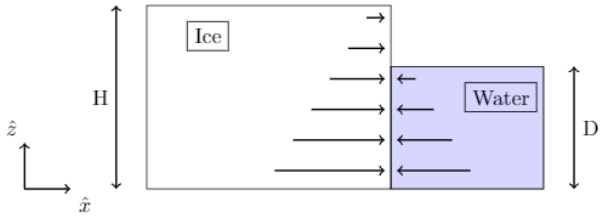


Figure 2: Here the glaciostatic and hydrostatic pressure balances each other.

Using these approximations and Navier-Stokes equations an expression for τ_{xz} can be found. In 2D they simplify to:

$$\rho \frac{Du}{Dt} = -\frac{\partial p}{\partial x} + \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{xz}}{\partial z} + \rho g \sin \alpha$$

$$\rho\frac{Dw}{Dt} = -\frac{\partial p}{\partial z} + \frac{\partial \tau_{xz}}{\partial x} + \frac{\partial \tau_{zz}}{\partial z} - \rho g\cos\alpha$$

Imposing a stress balance such that the acceleration of the ice is zero means that the material derivative becomes zero, hence: $\frac{Du}{Dt} \approx \frac{Dw}{Dt} \approx 0$

Imposing a Cartesian coordinate system where $\hat{z}$ is parallel to $\bar{g}$ results in $\alpha = 0$ such that $\sin\alpha = 0$ and $\cos\alpha = 1$, only taking the stress deviator τ_{xz} into account, and using the fact that $\frac{\partial}{\partial x}\tau_{xz}$ is small (see Fig.1), we get:

$$\frac{\partial p}{\partial x} = \frac{\partial}{\partial z}\tau_{xz} \tag{1}$$

$$\frac{\partial p}{\partial z} = -\rho g \tag{2}$$

Using that the pressure of the ice on top of ice is zero: $\tau_{xz} = p = 0$ at $z = h$, p can be found from Eq.2. Inserting this into Eq.1, τ_{xz} can be found:

$$p = \int_h^z -\rho g\, dz = (h - z)\rho g \tag{3}$$

$$\tau_{xz} = \int_h^z \frac{\partial p}{\partial x}\, dz = \int_h^z \frac{\partial h}{\partial x}\rho g\, dx = -(h - z)\frac{\partial h}{\partial x}\rho g \tag{4}$$

Since we know the glacier is flowing, we know there must be a layer where $\tau_{xz} = \tau_y$. Furthermore, we know this layer must coincide with the bed since we know τ_y is the largest shear stress the ice can withhold and τ_{xz} increases linearly with depth. Hence the change in surface elevation over a distance along the flowline can be found imposing $\tau_{xz} = \tau_y$ at the bed.

$$\frac{\partial h}{\partial x} = \frac{\tau_y}{\rho_i g(h - b)} \tag{5}$$

Since this is a first order differential equation it can be solved numerically with a forward Euler approach using only one boundary condition. This can be done at any point on the glacier. We have chosen to start at the terminus position and calculate the profile from this point. For this to be done we need a theoretical estimation of the height of the ice at the calving front which comes from the following.

2.2 Derivation of H_{term}

Usually the profile derived from Eq.5 results in a glacier that terminates with a continuous steep descent. This is contrary to what is observed at outlet glaciers where they usually terminate either as a floating ice shelf or abruptly as a vertical ice wall. This abrupt termination can be explained, as in *Bassis and Walker (2012)* with the introduction of two constraints on the vertical surface that is the ice front. The conditions are that the main stresses in the vertical plane equals the yield stress, and that the ice is in a force equilibrium. On the ice front the main stress is τ_{xx}, and since we expect calving from the ice, this stress is close to the yield stress. This configuration can be seen in Fig.2.

Here the pressure and push from the ice is balanced by the water pressure. The hydrostatic pressure

from water is $p_w = -\rho_w g z$ and the glaciostatic pressure is $p_i = -\rho_i g(h-z)$. The balance is therefore $p_w = \sigma_{xx} = p_i + \tau_{xx}$. Here σ is the normal stress. This equality is integrated over depth:

$$\int_b^0 (-\rho_w g z)dz = \int_b^h \sigma_{xx} dz = \int_b^h \Big(\tau_{xx} - \rho_i g(h-z)\Big)dz \tag{6}$$

$$\Big[-\frac{1}{2}\rho_w g z^2\Big]_b^0 = \tau_{xx}[z]_b^h - \rho_i g\Big[(hz - \frac{1}{2}z^2)\Big]_b^h = \tau_{xx}(h-b) - \frac{1}{2}\rho_i g(h^2 + b^2 - 2hb) \tag{7}$$

$$-\frac{1}{2}\rho_w g D^2 = \tau_{xx} H - \frac{1}{2}\rho_i g H^2 \tag{8}$$

$$\tau_{xx} = \frac{1}{2}\rho_i g H\left(1 - \left(\frac{\rho_w}{\rho_i}\right)\left(\frac{D}{H}\right)^2\right) = \tau_y \tag{9}$$

Where the water depth, D is the distance from 0 to b and the height of the ice $H = h - b$. Then let τ_{xx} be the yield stress. From this H is isolated, and defined as H_{term}.

$$\tau_y = \frac{1}{2}\rho_i g H\left(1 - \left(\frac{\rho_w}{\rho_i}\right)\left(\frac{D}{H}\right)^2\right) \tag{10}$$

$$0 = -\tau_y H + \frac{1}{2}\rho_i g H^2 - \frac{1}{2}\rho_i g \frac{\rho_w}{\rho_i} D^2 \tag{11}$$

Now the roots for the quadratic function are found.

$$H = \frac{\tau_y}{2\frac{1}{2}\rho_i g} \pm \frac{\sqrt{\tau_y^2 + 4\left(\frac{1}{2}\rho_i g\right)\left(\frac{1}{2}\rho_i g \frac{\rho_w}{\rho_i} D^2\right)}}{2\frac{1}{2}\rho_i g} \tag{12}$$

$$= \frac{\tau_y}{\rho_i g} \pm \frac{\sqrt{\tau_y^2 + \left(\rho_i g\right)^2 \frac{\rho_w}{\rho_i} D^2}}{\rho_i g} \tag{13}$$

$$= \frac{\tau_y}{\rho_i g} \pm \frac{\sqrt{\left(\rho_i g\right)^2\left(\frac{\tau_y^2}{\left(\rho_i g\right)^2} + \frac{\rho_w}{\rho_i} D^2\right)}}{\rho_i g} \tag{14}$$

$$H_{term} = \frac{\tau_y}{\rho_i g} \pm \sqrt{\left(\frac{\tau_y}{\rho_i g}\right)^2 + \frac{\rho_w}{\rho_i} D^2} \tag{15}$$

The solution with minus gives a height equalling zero on land and is negative when the glacier terminates in water therefore only the maximum solution is used.

If the ice column is lighter than the column of water the ice starts to float on the water. Meaning if $H < \frac{\rho_w}{\rho_i} D$ there will be a floating tongue or ice shelf and the model does not take those into account. If this is found the model is terminated and the place, where the tongue starts is made the terminus position. This is only important if the profile is integrated from the interior towards the coastal parts. If the model is integrated from terminus position and inward the terminus height is found and therefore there cannot be any floating tongue.

2.3 Derivation of the Time Evolution, $\frac{dL}{dt}$

The perfect plastic approximation has no time evolution. Therefore, another model is implemented where deformations of the ice is permitted. In this model the only variable that continues from the perfect plastic model is the terminus height which is constrained by the material parameter τ_y. Whereas the perfect plastic model comes from shear and yield stresses this extended model uses Glens' flow law.

The terminus position changes, due to changing surface mass balance. The rate of retreat or advance of the glacier also depends on the bed, but over time the change in SMB is the most important factor.

Below is the derivation of the change in terminus position over time following from *Ultee and Bassis (2020b)*. The terminus position is at $x = L$, and the divide at $x = 0$. H_{term} is again the height of the glacier at the terminus position.

$$\left.\frac{DH}{Dt}\right|_{x=L} = \frac{DH_{term}}{Dt} \tag{16}$$

$$\left[\frac{\partial H}{\partial t} + \frac{dx}{dt}\frac{\partial H}{\partial x}\right]_{x=L} = \frac{\partial H_{term}}{\partial t} + \frac{dL}{dt}\frac{\partial H_{term}}{\partial x} \tag{17}$$

$$\left.\frac{\partial H}{\partial t}\right|_{x=L} + \left.\frac{dL}{dt}\frac{\partial H_{term}}{\partial x}\right|_{x=L} = \frac{dL}{dt}\frac{\partial H_{term}}{\partial x} \tag{18}$$

$$\left.\frac{\partial H}{\partial t}\right|_{x=L} = \left.\frac{dL}{dt}\left(\frac{\partial H_{term}}{\partial x} - \frac{\partial H}{\partial x}\right)\right|_{x=L} \tag{19}$$

In the second line the $\frac{\partial H_{term}}{\partial t}$ vanishes because the terminus height is not time dependent but only depends on the bed which does not changes enough in time to contribute an significant effect.

Now the law of mass continuity is applied. We are only working in 2D because the vertical component can be set to go in the direction of the flowline. Therefore, the last part for the y-direction equals zero and the equation is evaluated at terminus. This makes the left side of Eq.19.

$$\frac{\partial H}{\partial t} = \dot{a} - \frac{\partial q_x}{\partial x} = \dot{a} - H\frac{\partial u}{\partial x} - u\frac{\partial H}{\partial x} \tag{20}$$

Here u is the velocity in the x-direction and $\dot{a}$ the SMB where basal melting and freezing is neglected. This means that the more accumulation of snow the faster the flux has to be if the glacier is to remain in steady state. Now u and $\frac{\partial u}{\partial x}$ have to be found. To find $\frac{\partial u}{\partial x}$ Glen's flow law is applied. H is evaluated at the terminus position and here τ_{xx} must be at the yield stress, which means $\tau_{Ef} = tr(\bar{\bar{\tau}}) = \tau_{xx} = \tau_y$ and the expression becomes:

$$\frac{\partial u}{\partial x} = \dot{\epsilon}_{xx} = A\tau_{Ef}^{n-1}\tau_{xx} = A\tau_y^n \tag{21}$$

Here A is the flow rate parameter from Glen's flow law and n = 3 found from *Cuffey and Paterson (2010)(p.75)*. The right side of Eq.20 is substituted into the left side of Eq.19 and Eq.21 is also substituted into the equation.

$$\dot{a} - H\frac{\partial u}{\partial x} - u\frac{\partial H}{\partial x} = \left.\frac{dL}{dt}\left(\frac{\partial H_{term}}{\partial x} - \frac{\partial H}{\partial x}\right)\right|_{x=L} \tag{22}$$

$$\dot{a} - A\tau_y^n H_{term} - u\frac{\partial H}{\partial x} = \frac{dL}{dt}\left(\frac{\partial H_{term}}{\partial x} - \frac{\partial H}{\partial x}\right)\Big|_{x=L} \tag{23}$$

$$\frac{dL}{dt} = \frac{\dot{a} - A\tau_y^n H_{term} - u\frac{\partial H}{\partial x}}{\frac{\partial H_{term}}{\partial x} - \frac{\partial H}{\partial x}} \tag{24}$$

To find u, Eq.20 is integrated with respect to dx.

$$\int_0^L \frac{\partial H}{\partial t}dx = \int_0^L \dot{a}dx - \int_0^L \frac{\partial q_x}{\partial x}dx \tag{25}$$

$$\int_0^L \frac{\partial H}{\partial t}dx = \int_0^L \dot{a}dx - (HU)|_{x=L} \tag{26}$$

$$(HU)|_{x=L} = \int_0^L \dot{a}dx - \int_0^L \frac{\partial H}{\partial t}dx \tag{27}$$

$$U|_{x=L} = \frac{1}{H_{term}}\left(\int_0^L \dot{a}dx - \int_0^L \frac{\partial H}{\partial t}dx\right) \tag{28}$$

$$= \frac{\dot{\alpha}L}{H_{term}} - \frac{1}{H_{term}}\frac{dL}{dt}\int_0^L \frac{\partial H}{\partial L}dx \tag{29}$$

Here the chain rule is applied: $\frac{\partial H}{\partial t} = \frac{\partial H}{\partial L}\frac{dL}{dt}$, and $\dot{\alpha} = \frac{1}{L}\int_0^L \dot{a}dx$. Now Eq.29 is substituted into Eq.24 and $\frac{dL}{dt}$ is found.

$$\frac{dL}{dt} = \frac{\dot{a} - A\tau_y^n H_{term} - u\frac{\partial H}{\partial x}}{\frac{\partial H_{term}}{\partial x} - \frac{\partial H}{\partial x}} \tag{30}$$

$$= \frac{\dot{a} - A\tau_y^n H_{term} - \left(\frac{\dot{\alpha}L}{H_{term}} - \frac{1}{H_{term}}\frac{dL}{dt}\int_0^L \frac{\partial H}{\partial L}dx\right)\frac{\partial H}{\partial x}}{\frac{\partial H_{term}}{\partial x} - \frac{\partial H}{\partial x}} \tag{31}$$

$$\frac{dL}{dt}\left(\frac{\partial H_{term}}{\partial x} - \frac{\partial H}{\partial x}\right) = \dot{a} - A\tau_y^n H_{term} - \frac{\dot{\alpha}L}{H_{term}}\frac{\partial H}{\partial x} + \frac{1}{H_{term}}\frac{dL}{dt}\int_0^L \frac{\partial H}{\partial L}dx\frac{\partial H}{\partial x} \tag{32}$$

$$\dot{a} - A\tau_y^n H_{term} - \frac{\dot{\alpha}L}{H_{term}}\frac{\partial H}{\partial x} = \frac{dL}{dt}\left(\frac{\partial H_{term}}{\partial x} - \frac{\partial H}{\partial x}\right) - \frac{1}{H_{term}}\frac{dL}{dt}\int_0^L \frac{\partial H}{\partial L}dx\frac{\partial H}{\partial x} \tag{33}$$

$$= \frac{dL}{dt}\left(\frac{\partial H_{term}}{\partial x} - \frac{\partial H}{\partial x} - \frac{1}{H_{term}}\int_0^L \frac{\partial H}{\partial L}dx\frac{\partial H}{\partial x}\right) \tag{34}$$

$$= \frac{dL}{dt}\left(\frac{\partial H_{term}}{\partial x} - \frac{\partial H}{\partial x}\left(1 + \frac{1}{H_{term}}\int_0^L \frac{\partial H}{\partial L}dx\right)\right) \tag{35}$$

And finally the time evolution is found (notice that there is two sign differences compared to the supplementary paper *Ultee and Bassis (2020b)*).

$$\frac{dL}{dt} = \frac{\dot{a} - A\tau_y^n H_{term} - \frac{\dot{\alpha}L}{H_{term}}\frac{\partial H}{\partial x}}{\frac{\partial H_{term}}{\partial x} - \frac{\partial H}{\partial x}\left(1 + \frac{1}{H_{term}}\int_0^L \frac{\partial H}{\partial L}dx\right)} \tag{36}$$

3 Description of Applied Data

The main data of bedrock topography is from bedmachine by *Morlighem et al. (2021)*. These are the data that the MATLAB script uses. The bedmachine has data consist of bed topography and surface heights. The datafile also consist of surface heights from *Howat et al. (2014)*.

The bedmachine is also used to find driving stresses on Greenland. For this the height, the bed and the x- and y- coordinates are used.

To find the rate of retreat from the glaciers another data set of surface mass balance is used. This is data is from HIRHAM by *Langen et al. (2015)*. These data range from 1980 till 2016 and provide an annual mean of SMB in Greenland. They are found from mass balance and climate models. For the SMB data the mean values for the 36 years are used as an indicator of the general trend along the flowlines. We have not made the model variate for each year but made an average value of SMB. This can be seen in the appendix on Fig.19. The surface mass balance is a measurement of accumulation and ablation where ablation is the negative values.

Given the simplicity of the model we would not expect the uncertainties from the data to have a significant influence on the results and hence have used the data as it is without trying to estimate the uncertainties in both surface- or bed elevation.

To detect the flowlines a MATLAB script from *Hvidberg (2021)* is applied. The flowlines are found from where the velocity vector is highest. The velocities comes from *Joughin et al. (2018)*. The MATLAB script furthermore finds the bed geometry, SMB, surface elevation profile, width of flowline, and distance from terminus along the calculated flowline. To find the width of the flowline, two more flowlines are calculated. The width is used for estimating the mass loss. The flowlines that are calculated are found from the terminus and then 10.000 time steps are taken inwards. This makes the flowlines calculated $\sim 250\ km$ long.

From the MATLAB script the flowlines of the six glaciers in Greenland are found. These are: The Helheim Glacier, Jakobshavn Isbrae, Upernavik 1, Upernavik 2, Nioghalvfjerdsfjorden, and Zachariae Isstrom. In Fig.3 the calculated flowline from Upernavik 1 can be seen. The centered flowline is the one used to model the profiles and the upper and lower is used to determine the width of the flowline.

4 Implementation of the Models

In this paper the model is integrated from terminus and inward skipping the floating ice tongue problem. We have tested the model by making the integration both ways and the results are the same. It makes it easier to find the time evolution when the model is integrated from terminus and inward because time evolution changes the terminus position and thereby the height at divide. To make the surface elevation more realistic the step size in the numerical integration has been shortened by interpolating the data points of the bed geometry. From the data the step size was 5 km and with the interpolation the step size is set to 10m. This is done for the bed, distance, surface elevation, width and SMB.

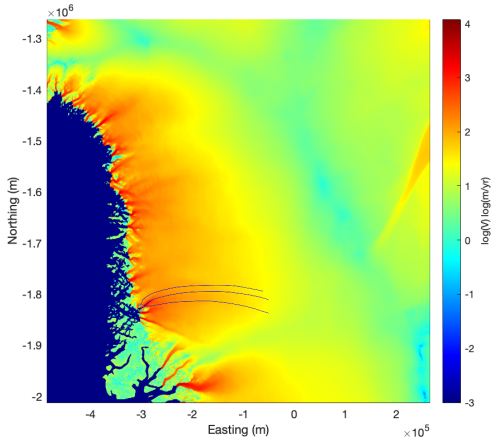


Figure 3: The flowline from the upper glacier of Upernavik. The centerline is blue and is started from terminus. The width of the flowline is plotted. The map is of bed elevation on Greenland.

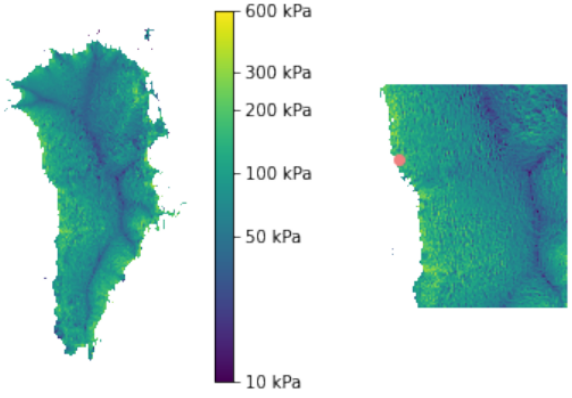


Figure 4: On the left are driving stresses for Greenland and the right figure has been zoomed in for the same part of Greenland where Upernavik 1 is, as the flowline. The red dot is the terminus position of Upernavik. It can be seen that the stresses are highest at the border of Greenland and decreases into a minimum at the divide. There stresses are calculated directly from the surface elevation and slope using Eq.38.

4.1 Numerical Solution

To find the surface elevation and thereby the profile of the glaciers, numerical integration is applied over the bed geometries along the different flowlines. Here Δx is the step size and $h(x_i)$ is the surface elevation at the present position. The integration is started from the terminus taking steps $\Delta x < 0$, where the coordinate system has the initial divide position as x_0.

The model can also be integrated from the divide or some other inward position on the ice and out with steps $\Delta x > 0$. If the model is to run from the divide and out it needs a starting height at the divide. Furthermore, it needs to be cut off and calve when the height equals the terminus height or needs to stop if the height equals the bed or floating ice shelves occurs. This is also avoided starting from terminus (*Ultee and Bassis (2016)*). The equation of a numerical solution:

$$h(x_{i+1}) = \frac{\tau_y}{\rho_i g(h(x_i) - b(x_i))}\Delta x + h(x_i) \tag{37}$$

When the integration is started from terminus position, the terminus height, H_{term} is calculated and used to find the first $h(x_i)$, which is height above sea level or bed from $H(x_{term}) = h(x_{term}) - b(x_{term})$. In this model the yield stresses, τ_y are a list of two different values and ρ_i, g are constant.

As mentioned, floating ice tongues are ignored and five of the glaciers in this paper do not have any floating ice tongues or shelves. At Nioghalvfjerdsfjorden there is some ice further out in the fjord, but this is not part of the profile modelled.

4.2 Yield Stresses

The model that we are using is simple. With only one free parameter which is the yield stress. By letting the yield stress be the only free parameter all the physics can be pulled into this. Then by adding two values of the yield stress it becomes possible for the model to make profiles that are more alike the observed glaciers without losing the simplicity of the model.

Since we equate the yield stress to the driving stress, it is important to see if and how much the driving stresses on Greenland varies, the driving stress is found from the formula in *Cuffey and Paterson (2010)(p.296 l.11)* and is equivalent to Eq.5:

$$\tau_d = -\rho_i g H \frac{dS}{dx} \tag{38}$$

Where $\frac{dS}{dx}$ is the change in surface elevation and H is the surface elevation. This is applied to find the driving stresses in every pixel over Greenland from bedmachine data by *Morlighem et al. (2021)*. This can be seen in Fig.4. The stresses are only found from the part of Greenland that has grounded ice since it is only here the driving stresses are. This calculations is also done along the flowlines. As discussed later, two different yield stresses has been used in the model for each glacier. This differs from *Ultee and Bassis (2016)*, where τ_y is a function of H and D.

4.3 Determine Yield Stress

For determining the best τ_y values we have used two different approaches. The first one relies on a manual estimation of where the τ_y values changes called x_C. Using this position, the best τ_y from the terminus position up until x_C is found, using a iterative process where $\Delta\tau_y$ is added to the original τ_y, until the calculated surface profile matches the observed surface profile best, using a least squares approach as the loss function. Using the τ_y found for the outer part of the profile, the second τ_y for the inner part of the profile is found, using a similar method.

Manually choosing the position of where to change τ, simplifies the optimization problem by effectively reducing it to optimizing three (semi) independent variables. This means finding the two optimal τ_y values is a fast operation and should result in something close to the optimal solution. The downside is that this approach does not scale well with the number of glaciers considered, for six glacier it is okay, but more would be too time demanding. Furthermore, this approach would result in different results given different people could give different estimations for the change position.

The second approach is a full gradient decent approach of the three-dimensional parameter space. While fully automated, this is computationally heavy and likely to find a local minimum different from the optimal solution. It is computationally heavy since it needs to calculate
$(\#\text{ of independent parameters})^3 = 3^3 = 27$ surface profiles for each iteration, and propagating the best parameters for the best profile to the next iteration, only stopping when an optimal parameter set is found. While computationally heavy, with only three parameters it is easily done. The problem with ending up in local minimum, can be solved by using "smart" starting values for the parameters, and also try running the algorithm from different starting parameters positions or with different $\Delta\tau_y$ and

seeing if the result is the same. An overview of this approach can be seen in Fig.5.

Generally, the found stresses from the two different approaches are almost the same, the biggest difference in stress is $\sim 10\ kPa$ which is still in the same order of magnitude $\sim 4\%$. The positions found where the yield stresses changes are varying with $\sim 10\ km$ making it 4% of the length.

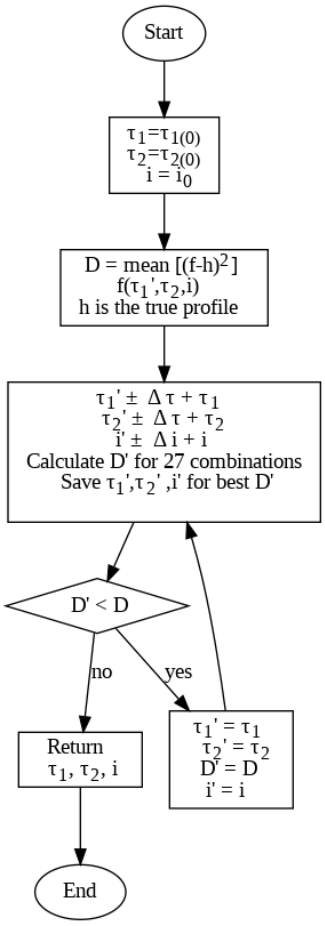


Figure 5: Flowchart of the way the two values for τ is found.

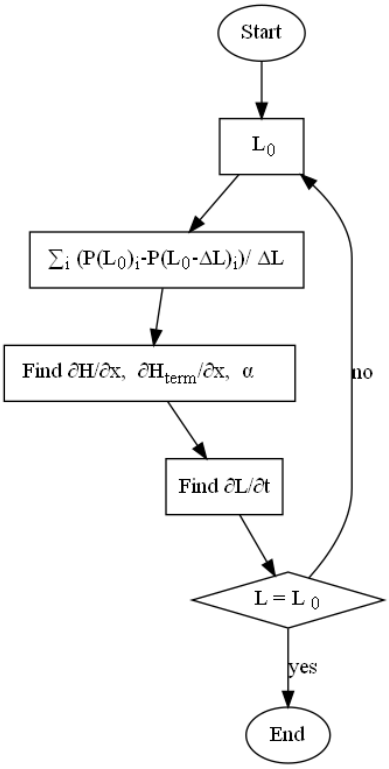


Figure 6: Flowchart of how the time evolution and stable point is found. Here P is the function that finds the profile from an initial condition, here terminus position.

4.4 Time Evolution, $\frac{dL}{dt}$

The flow rate parameter, A, is temperature dependent. The temperature is set to be $-10\ ^\circ C$ throughout the ice and $A = 3.5 \cdot 10^{-25} \cdot 356 \cdot 3600 \cdot 24 Pa^{-3}\ yr^{-1}$ as from *Cuffey and Paterson (2010)(p.75)*. The densities are also set to be constants, $\rho_i = 917\ \frac{kg}{m^3}$ and $\rho_w = 1027\ \frac{kg}{m^3}$.

To implement the time evolution Eq.36 has been made into discrete steps as it can be seen in Fig.6. Here P is the function that finds the profile from the terminus position. It is important that the model is run from terminus and inward or else some of the signs will be wrong.

To make the uncertainties of the heights smaller we take n points (here $n = 10$) to both side of the actual position of L and make an average of the height. This makes $\frac{dH}{dx}$ and $\frac{dh_{term}}{dx}$ more accurate.

To find an estimation of mass loss the time evolution is used. From this a function is made that

takes the terminus position and finds the time evolution, then takes the new position and repeat. This is done with time steps set to 0.5 yr. Then the profile from the new terminus position is calculated. If the retreat in the current position exceeds the expected values greatly and hence is probably an outlier coming from the numerical method the retreat from a random nearby point is calculated and used. This continues till the terminus position is stable.

The difference in height in the profile is found and multiplied with the width of the marine outlets and the length of dx to give an estimate of the ice mass loss.

5 Testing the Models

5.1 Profiles for Idealized Beds

To test how the model for determining the profile works, it is tested on idealized beds which are relevant to the bed geometries of glaciers. This can be seen in Fig.7. The first plot is of a linear downward sloping bed and the other is with a Gaussian bump on it. These tests are inspired by Ultee and Bassis. The τ_y value is set to 100 kPa, which normally is found to be around 50-300 kPa from *Ultee and Bassis (2016)*. The height at starting position is found from H_{term} which depends on τ_y and the depth of water. On Fig.7d is a plot of the height of the glacier if the model is being integrated till divide from different terminus positions. Here the divide is at a determined position and the terminus position is varied. There are two different beds made to see the impact a bump has on the profile. The bed has the same slope, the Gaussian bump is centred around 170 km from the divide.

To see the profiles at terminus, see Fig.20 in appendix. Here is a glacier that terminates in water or on land both on a linear sloping bed.

5.2 Yield Stresses on Different Beds

Different τ_y values gives different slopes on the profile. This can be seen Fig.8a. From the flowlines the driving stresses are calculated, in the same way as for the entire Greenland, and plotted as the oscillating line in Fig.8b. Next to this are the two different yield stresses found as described previously. The calculated driving stresses that are oscillating are found from every 2 km on the flowline of Upernavik. This can be seen on Fig.8b and the two different ways of estimating the stresses gives almost the same values. The driving stresses are found and it can be seen on Fig.4 that they match the others found and are of the order $\sim 10^5$ Pa. They are varying with the greatest values at the edge of the ice cap and smallest at the ice divide.

In this paper two different τ_y values are found from each glacier to better fit the profile. This can be seen on Fig.8c) and d). A low τ_y fits well with a flat surface and a higher τ_y fits with steeper slopes at the surfaces. These are the yield stresses plotted in Fig.8b. As it can be seen, the profile made from two values of yield stresses matches the observed profile better than one value of yield stress would.

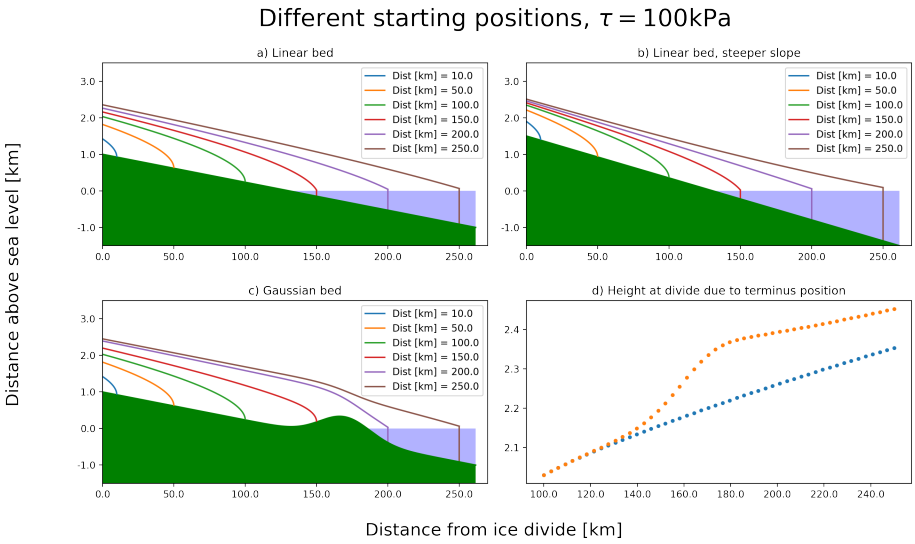


Figure 7: A cross section of three different idealized beds are shown. The green is bed, blue is water, and the different lines are profiles due to different termini positions. a+b) Shows two linear beds with profiles determined from the different termini positions. The second has a steeper slope than first. c) Linear bed with a Gaussian bump and also profiles determined from terminus position. On figure d) The height at divide due to terminus position. The height at divide is calculated from the terminus position. It can also be seen that shorter terminus length gives smaller ice cap at the divide. Here the blue is from a linear bed and orange is from a bed with a Gaussian bump.

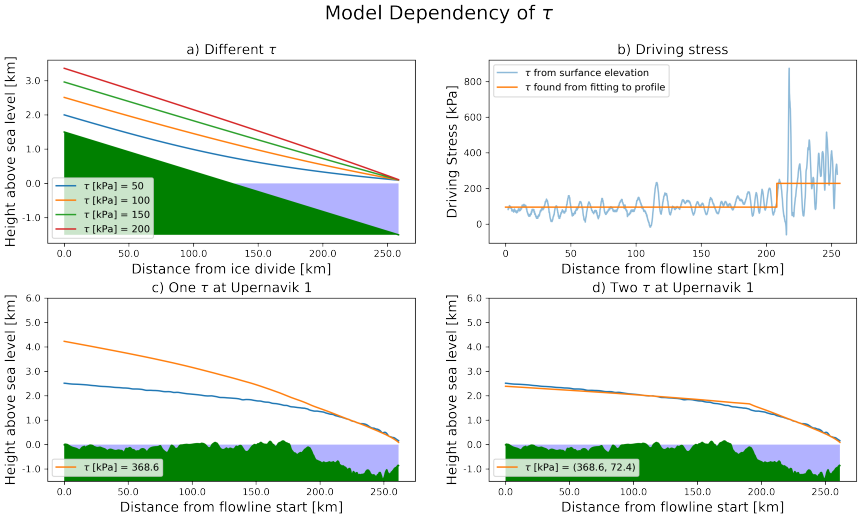


Figure 8: a) The profiles from four different yield stresses. b) Driving stresses calculated from the surface elevation and yield stresses from fitting to the profile. The fitted (orange) follows the trend from the calculated (blue). c+d) Shows the observed data and modelled profile fitted by one or two constant value for yield stress. All from Upernavik 1.

5.3 Time Evolution on a Linear Bed

The time evolution has also been tested on a linear bed with a SMB of 0, meaning no accumulation or ablation, and the same yield stress for the hole flowline. This is done to test how the model responds to a

very simple bed geometry and SMB. This can be seen on Fig.9. The green is the slope and blue is water and these are just to give intuition of what the trends are at the two different regions of termination in water and on land. Therefore the y-axis belongs to the time evolution (orange dots).

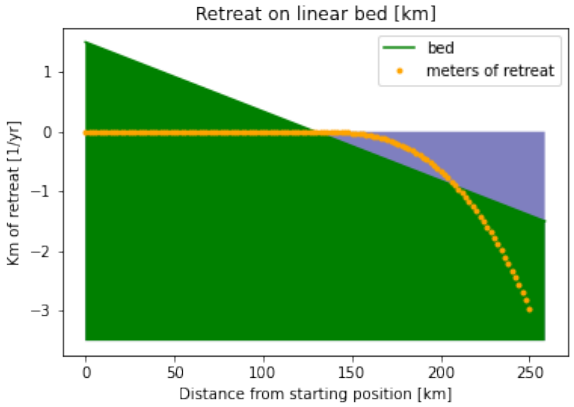


Figure 9: This figure shows the time evolution on a linear bed. The background of bed and water is just to make it easier to see where the glacier is stable and where it is retreating. The retreat is steady until the glacier terminates in water.

With finding the time evolution on a linear bed, the accumulation is also set to be zero. This simplifies Eq.36 of the time evolution to be:

$$\frac{dL}{dt} = -\frac{A\tau_y^n H_{term}}{\frac{\partial H_{term}}{\partial x} - \frac{\partial H}{\partial x}\left(1 + \frac{1}{H_{term}}\int_0^L \frac{\partial H}{\partial L}dx\right)} \tag{39}$$

As it can be seen on Fig.9 the glacier rapidly retreats when it terminates in water. When the glacier terminates on land the retreat is steady while in water it scales as $-x^2$ with zero accumulation. This can be seen from looking at the individual terms in Eq.39: Plots of all the terms are in appendix in Fig.21.

On land: H_{term} is positive and constant, hence $\frac{\partial H_{term}}{\partial x}$ is zero. $\frac{\partial H}{\partial x}$ evaluated at the terminus position is constant with respect to a change in L and negative and the integral is $\int_0^L \frac{\partial H}{\partial L}dx \sim \sqrt{x}$. Making the terms: $\frac{dL}{dt} \sim -\frac{k}{\frac{1}{k}(1+k\sqrt{x})}$. Hence as x grows: $\frac{dL}{dt} \sim -\frac{1}{\sqrt{x}}$.

On water: H_{term} is positive and since the bed has a linear slope then $H_{term} \sim x$ when it terminates in water and hence $\frac{\partial H_{term}}{\partial x}$ is a constant. $\frac{\partial H}{\partial x}$ now scales as $\frac{1}{x}$ The integral scales now linearly with x. Making the terms: $\frac{dL}{dt} \sim -\frac{x}{k+\frac{1}{x}(1+\frac{1}{x}kx)} = -\frac{x}{k+\frac{1}{x}}$. Hence as x grows: $\frac{dL}{dt} \sim -x^2$.

6 Results

Now results for the data of the glaciers are presented.

6.1 Profiles and Driving Stresses for the Glaciers

The perfect plastic approximation has been used on the six different glaciers and from this different τ_y values are found to match the observed profile. On Fig.10 profiles with the best τ_y values are shown.

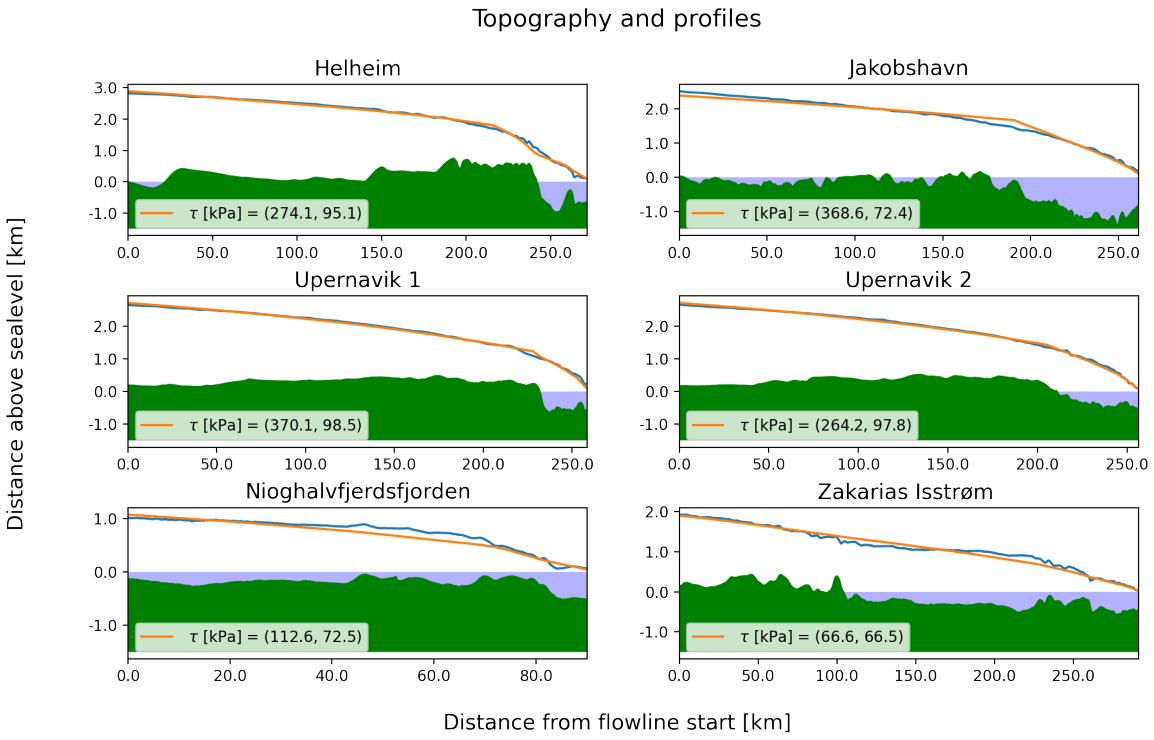


Figure 10: Profiles for the different glaciers. The blue line is the observed surface elevation and orange are the modelled profile. The distance is along the flowlines and are measured form the position of where the flowline stopped, not at the divide. The profiles are represented with the best fitting two τ_y values.

There are for each glacier two different values, for the outer and inner part of the glacier. In Table 1 the value from each glacier are shown. Here x_C is how far from the terminus the best fitting value for yield stresses changes. For all the glaciers the greatest value of τ_y is found in the outer part of the glacier, close to terminus. This is except for the glacier at Zachariae Isstrom where the yield stresses are almost identical. When the flowlines are found they start, as mentioned earlier, from terminus and continues inward. On the figures of the different profiles from the glacier; $x = 0$ is where the flowline ends and x increases towards terminus. This is done to easier compare our model with the model from *Ultee and Bassis (2016)*. The vertical line on the plot indicates when the value of the yield stress changes.

	Helheim	Jakobshavn	Upernavik 1	Upernavik 2	Nioghalvfjerdsfjorden	Zachariae Isstrom
x_C [km]	56.0	70.7	30.4	51.1	18.6	4.6
τ_1 [kPa]	274.1	368.6	370.1	264.2	112.6	66.6
τ_2 [kPa]	95.1	72.4	98.5	97.8	72.5	66.5

Table 1: Different τ_y Values for the Glaciers. x_C is the position where the yield stress value changes.

To see what effect x_C has, the width of the flowline, the SMB and the found yield stresses are

plotted. In Fig.11 this can be seen for Upernavik 1. The light vertical blue line indicates the position of changing yield stresses. The same figure has been made for Jakobshavn on Fig.22 and a similar for all the other glaciers and can be seen in appendix Fig.23.

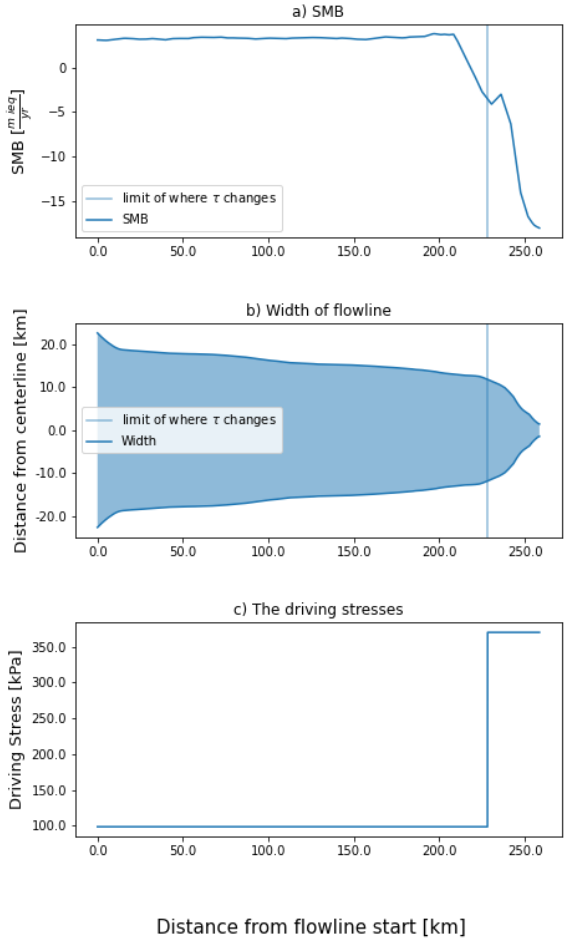


Figure 11: This is the width, SMB and found yield stresses for Upernavik 1. Here the vertical line is the limit of where we have let the yield stress value change.

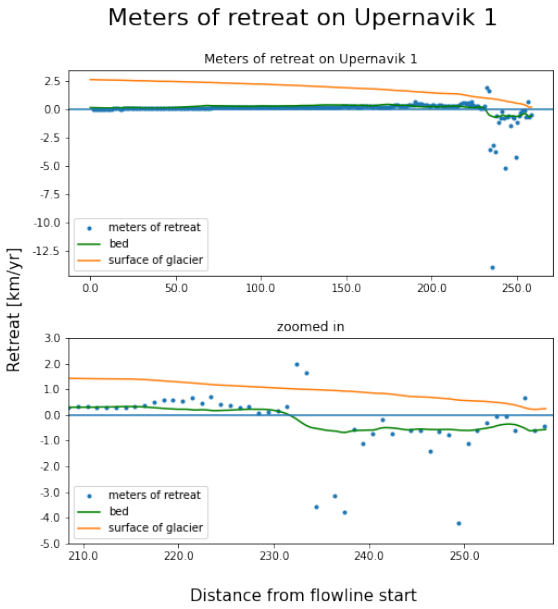


Figure 12: This figure shows the time evolution from Upernavik. The dots indicates the retreat/advance the glacier will make if terminus is at this position. It can be seen that in the outer part the evolution is negative meaning that the glacier is retreating. In the inner part it becomes positive meaning that it will advance. The stable point will be where $\frac{dL}{dt}$ changes sign. The upper plot is of the entire glacier and the lower is from the outer part. This is the range where the retreat happens.

6.2 Time Evolution

The time evolution of Upernavik 1 is shown on Fig.12. When the value is negative, the glacier is retreating. The time evolution has been calculated for every 2 km starting from terminus.

6.3 Estimation of Mass Loss

The iterations can be seen on Fig.13a which is from Upernavik 1. This shows the retreat in each time step, and also which length it will stabilize at. The two different profiles can be seen on Fig 13b. And the retreat off the glacier can be seen. For the other glaciers the same plots are made on Fig: 14, 15,

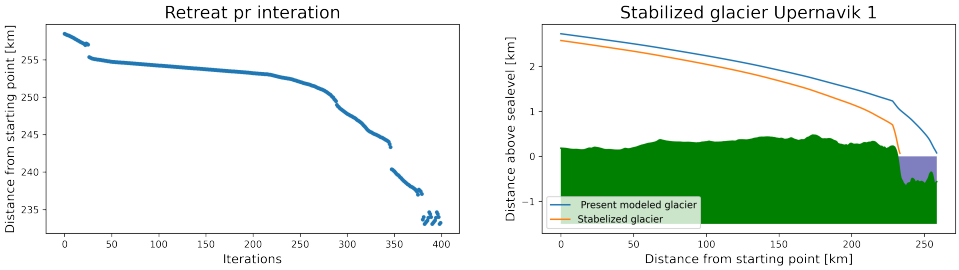


Figure 13: Left: Here the new terminus positions are found, and the stable point is where function is oscillating around. The iterations are found from every 0.1 yr. Right: This figure shows the present profile (blue) and the profile of the glacier when it has stabilized (orange). It can be seen that it stabilizes closer to the divide, since the time evolution is negative. The height at divide is almost the same since the retreat is small relative to the length of the glacier.

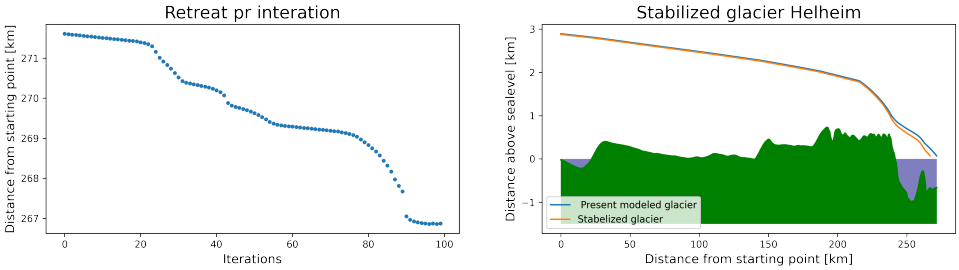


Figure 14: Here are the stable point from Helheim. This glacier stabilizes faster than the others and only retreats ~ 5 km. It can be seen that this glacier stabilizes right before the bed has a steep slope from a mountain under water.

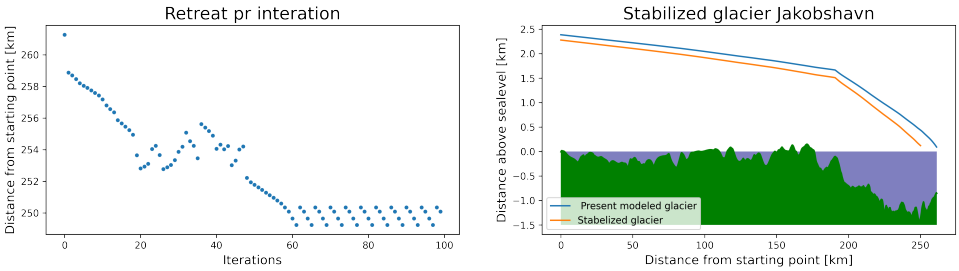


Figure 15: This is the time evolution from Jakobshavn. From this glacier there is both some advance and retreat. But it ends up stabilizing ~ 10 km inward. The slope of this bed is also fluctuating. This fits with the pattern of evolving that is seen.

16, 17, and 18.

In Table 2 are the estimated mass loss from the six glaciers. The estimated mass loss if from when they have retreated to a stable position from the present position. The mass loss is given in volume of ice and found from the difference of ice in the profile and the width of the flowlines.

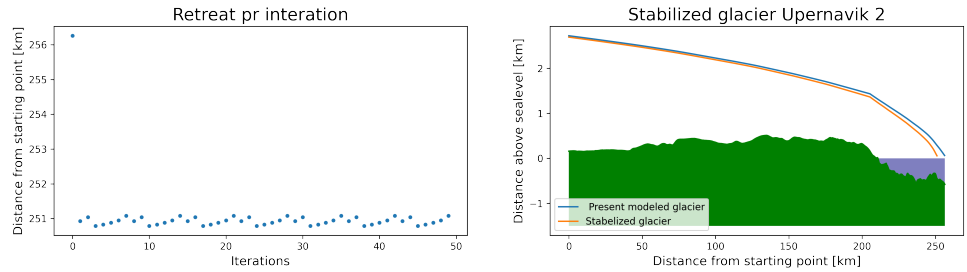


Figure 16: This retreat is from Upernavik 2. Here the first point is very unstable and the glacier fast retreats to a stable position. Again, this stable position is right next to a steep slope. These are holding back the glacier for further retreat. The big retreat and there after stable terminus position indicates some uncertainty on this retreat. But since the stable point is right next to a steep slope the overall retreat looks plausible.

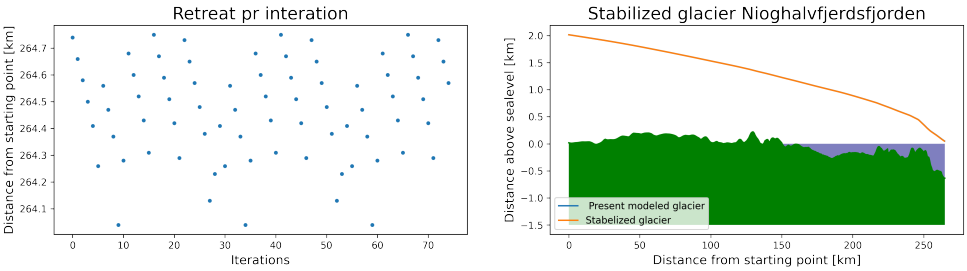


Figure 17: This is from the glacier at Nioghalvfjerdsfjorden. Since the retreat oscillates in a 500 m range, the glacier seems stable at the current position.

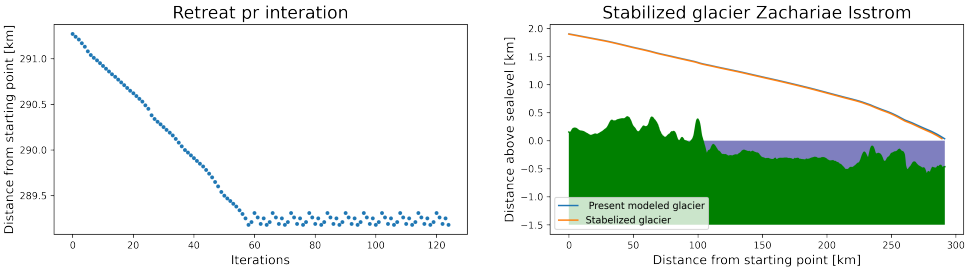


Figure 18: Here is the retreat from Zachariae Isstrom. This glacier retreats ~ 3 km. This is done with a slope that looks linear. This looks like the glacier at present is almost at it is stable point.

	Helheim	Jakobshavn	Upernavik 1	Upernavik 2	Nioghalvfjerdsfjorden	Zachariae Isstrom
Mass loss $[km^3]$	522 ± 1	3500 ± 200	2140 ± 50	233 ± 9	Undefined	111 ± 5

Table 2: Estimated Mass Loss for the Glaciers.

7 Discussion

7.1 Idealized Flowlines

To see how the model will act on real glacier some idealistic beds are made. These have realistic slopes and lengths.

The steepest slope of the linear beds has a slope 1.5 times as great as the flatter slope. By looking at the idealized flowlines in Fig.7 it can be seen that the steeper the slope on the bed, the smaller the ice cap will be at the divide. This is because the height at the bed is in the denominator in Eq.5 and can be seen from the profiles at $x = 0$. On the plot the lines indicating the profiles lies closer together. This is also what intuition tells: the steeper the slope the less ice will be able to stabilize there.

By looking at the bed with the Gaussian bump it can be seen, from the two profiles terminating farthest out, that the bump holds back the ice. These have almost the same height at the divide. This must be because the hill provides a pressure point of resistive stress on the ice.

To investigate this further a plot showing divide height calculated from the terminus position is made. In Fig.7d a plot is made of the different terminus positions both with a linear bed and a bed with the same slope, but a Gaussian bump. Here the bump is located 170 km from terminus, as on Fig.7c. On the y-axis is the height of the ice cap at the divide. Here it can be seen that the bump holds back the ice if the terminus position is further out than the bump. With a terminus position right behind the bump, the height at divide varies a lot due to the local downward sloping bed. Therefore, retreat will be most rapidly behind the bump and almost stable further out on the bump.

7.2 Tau

From Fig.4 it can be seen, that the driving stress resulting from the slope of the ice surface, generally can be divided into three segments when following a flowline from the ice divide to the edge of the glacier. At the ice divide driving stress is generally less than 50 kPa, between the divide and the coast it is fairly stable around 100 kPa, and at the coast the driving stress increases again to 200 kPa or indeed larger. Since the thin film approximation equates the driving stress to the basal shear stress, and the perfect plastic approximation equate the basal shear stress to a yield stress, this implies that if both approximations are fairly accurate, we should at least use two different yield stresses in the calculation of the profile. Since the flowlines are not run till the ice divide only two values representing the outer parts are appropriate.

Since driving stresses are not the main stress at the divide, we would not expect the perfect plastic approximation to be accurate. This is because the perfect plastic model uses the bed geometry but near the divide the profile is not only dependent on local variables as bed geometry and the slope of the surface, but also on the outer parts of the glacier. Also, internal deformation in the ice is also crucial at the divide to determine the profile there, which is assumed to be zero in the approximation. This makes the map in Fig.4 show driving stresses that are to low at the the divide and too high at coastal regions. Therefore, the perfect plastic approximation is more accurate at terminus, because here the deformation within the ice is small compared to calving but still not perfect, where at the divide the

internal deformations are a greater factor of deformation, making the perfect plastic model less precise here. This means that the calculated driving stresses are most accurate between the divide and terminus where there is no calving and less deformation.

We have, as described, made the yield stresses two different constant. This is verified by the two plots of driving stresses calculated from the data and seen in Fig.4. These two plots shows the driving stresses calculated from the surface elevation. The first is of all of Greenland and the second follows the flowline from Upernavik 1. The first plot shows that the driving stresses are highest on the edge and that it decreases till a minimum at the divide. To compare with the driving stresses we use the other plot seen in Fig.8b. The flowline is from terminus and inward and stops before the divide is reached. Here it shows that the simple two-term τ_y follows the trend of the oscillating pattern. This trend is seen in the other glaciers as well. As can be seen on the figure, the driving stress is not decreasing as much on the figure of the flowline as on the figure of entire Greenland.

An other way of finding the yield stresses, which was used in *Bassis and Walker (2012)*, could be to make a function that was linearly increasing from divide and out. This is trying to accommodate for a high τ_y inland due to hard bedrock, and a lower τ_y at the terminus due to softer sediments and water pressure. In the paper there is a discussion of which value of τ_y to use. There is both a constant value and a value which varies as a function of height of ice and water depth, due to the change of friction at the bed.

$$\tau_y = \tau_0 + \mu\big(\rho_i g(h-b) - \rho_w g D\big) \tag{40}$$

But this does not catch the x_C position either and therefore they have shortened the flowline they have looked at. This would not fit well either due to the x_C position and since we find the lowest values of τ_y at the inner parts and not at the coast.

As can be seen on Fig.8 the two different values for yield stresses Upernavik 1 is 370.1 kPa and 98.5 kPa. This means that the biggest are ~ 4 times as big as the smallest one. If we look at Fig.11 it can be seen that where the yield stress values changes is also where the width of the flowline gets smaller. This looks like the outlet is going in a valley where stress from the walls should also be considered. This indicates that the resisting stress could accommodate a higher driving stress and therefore a higher yield stress should be used in the model. At the same time this position is also where the SMB changes. But also the topography changes. Further out the glacier terminates in water, and further in the glacier terminates on land. Thereby it is difficult to conclude which of the three conditions that determine the driving stress, but it is probable that all three have an impact.

The trend here is the same that can be seen from the other glaciers. This can be seen on Fig.23 in appendix. Here the variables are normalized to make all the variables visible in the same plot.

As it can be seen on Fig.10 the τ_y values on the inner parts of the glaciers differs from each other with no more than ~ 30 kPa, whereas the outer τ_y values varies greatly. This indicates that bed geometry and thereby that the basal shear stresses are similar. The bed geometries on the coastal parts of Greenland differs more because here the glaciers are passing through a fjord, where the walls contribute

to the total resisting stress. This difference is accommodated in the f' factor introduced in *Cuffey and Paterson (2010)* p.296 and p.341-342: $\tau_b = f'\tau_d$. Usually over a flat bed, without walls $f' \sim 1$, if the side walls and half the width of the flowline is of equal size $f' \sim 0.5$.

While introducing several τ_y yields markedly better fits, and perhaps incorporate the underlying geometry of the geology better than a single τ_y, it also introduces larger uncertainties in the found values. When optimizing for one τ_y it is reasonable to expect the found value to be the best for the problem, with a three-dimensional parameter space it is not unlikely to expect several solutions to yield equal or at least similar values, with similar loss. Furthermore, while the calculation of one profile is not computational heavy, a thorough parameter search for each glacier is not favourable. Finding viable solutions via a gradient decent method, using "good" starting parameters close to the expected values, does consistently yield good and repeatable results close to values found by eye. Naturally both τ_y depend on where the shift from one to the other occur, and hence using a suitable starting limit is crucial. Since we expect $\tau_1 \sim 200\ kPa$ and $\tau_2 \sim 100\ kPa$, these can be used to find the optimal position for the change in τ_y for these values with a simple one dimensional parameter search, and then propagate this result to the three dimensional parameter search with this as the starting parameter.

7.3 Uncertainties

Since the whole model of the profile only has τ_y as a direct model parameter all uncertainties are pooled into this variable. Hence it is difficult to precisely determine the actual uncertainties and from where these uncertainties originate. The first and probably foremost contributor of uncertainties comes from the simplicity of the model itself. With the perfect plastic model, we try to summarise the incredible complex systems marine glaciers consist of into one first order differential equation and while reproducing the general shape, it do not reproduce local fast changing features that is observed in Greenland. The second likely origin of uncertainties is from the actual data used and while no data ever is without flaws, these flaws are dominated by the uncertainties coming from the simplicity of the model. For a comparison a change in τ_y by 1 kPa yields a difference of ten meters furthest from the terminus position.

For the model of time evolution of the terminus position one main uncertainty lies in the choices regarding the SMB along the flowline. While we know the SMB changes continuously, we have chosen to use the mean value of the 36 years of data. While this might not represent the present conditions and probably is an unwise metric to base predictions of future ice mass loss on, it does provide the general trend in SMB the glacier has been exposed as can be seen in Fig.19, a perhaps greater uncertainty lies in the discretization of the data and the model. While integrating the profile is not sensitive to the discrete problem the time evolution mainly consist of derivatives which are much more sensitive to this discretization and sometimes yield results that seems to contradict physical intuition. A simple possible solution to this would be to filter and smoothen the data, rather than using a linear interpolation between the data points. The reason we have not done this, is that the time evolution mostly produces the expected results, and the outliers arising from numerical problems that fairly easily can be sorted out during the evaluation of the retreat of the glacier.

In evaluating the ice mass loss a large uncertainty comes from the fact that we assume a constant density of the ice, and thereby completely ignores the firn layer. Where there is accumulation of snow, there will be a layer of firn which has lower density than ice. This is only on the inner part of the ice cap and not at the edge. Normally this is accounted for by making the height gradually smaller from the inner part and out. This is not done in this paper because the height is used to find the driving stresses and thereby the profile and the time evolution. The profiles from different terminus positions are then subtracted to find an estimate of the ice loss and here the errors should cancel out.

Furthermore, the data is sensitive to where the starting point of the glacier is set to. This results in quite large variation in flowlines. Which propagates into large uncertainties in the width of the profile and results in uncertainties on the yield stresses and where they changes.

7.4 Comparing Results with Other Studies

From the paper by *Larsen et al. (2016)* the terminus height is found to be 1 km for Upernavik 1, and the bed to intersect sea level at 45 km from the terminus. In our model the terminus height is 550 m above bed, and the bed and sea level intersect at 34 km from terminus. The difference in these values can be due to the fact that her observations and data has a higher resolution or the fact that two slightly different flowlines are calculated.

Ultee has also found yield stresses for glaciers. In her work it is only the outer part of the glacier the values are from, and she does not have yield stresses for all six glaciers. But she has from Jakobshavn $= 150\ kPa$, Upernavik $= 145\ kPa$ and Helheim $= 215\ kPa$. All our values are higher but at the same order of magnitude. This can be caused by her cutting off the inner part of the flowline an other place than where we let our model take an lower yield stress. If she has shortened the flowline further in than us her profile will have started to flatten. This means that the yield stress would have to be lower to fit the profile. Here from *Ultee and Bassis (2020a)* (Supplementary Table 1).

To compare our model with other work it can be seen in *Nick et al. (2013)* that a model that includes up to five tuneable parameters gets similar results for retreat at the glacier in Jakobshavn. The issue with this model compared to ours is that it is difficult to consistently optimize functions with more degrees of freedom than the three parameters that is applied in this model.

7.5 Ice Mass Loss

The estimation of drained water comes from the retreat of the glaciers. All the glaciers except for Nioghalvfjerdsfjorden are found to retreat which matches the present observations.

From Upernavik 1 in Fig.13 there is also retreat until the first steep slope. This is far from the present position, therefore much water will be drained from here. To make the model more stable the step size of time has been varied but the same stable point has been found both from small time steps and greater.

For Helheim, see Fig.14, it can be seen that the glacier stabilizes quite fast. It stabilizes right before a steep slope. This stable point is only 5 km from the present position therefore this glacier will not drain as much water as some of the others.

For the actual glaciers the retreat is rapid when it terminates in water. The glacier is stable or advancing a bit when grounding line is on land. The reason it is not always stable here is because the differing accumulation rate are non-zero in contrast to the case with idealized beds.

The estimation from Jakobshavn is not as smooth retreating as the Helheim glacier. This can be seen on Fig.15. The glacier also stabilizes at the same spot given different time steps.

From the other glacier at Upernavik 2 there is a fast retreat as can be seen on Fig.16. This looks like the glacier is very unstable at the present position. And after the first retreat the glacier stabilizes. This is again at a spot where the slope of the bed is great.

The glacier at Nioghalvfjerdsfjorden has no pattern in the retreat and advance. Therefore, there is no estimate of the retreat of this glacier, this might be caused by it already being in a stable equilibrium. This can be seen on Fig.17.

For the last glacier at Zachariae Isstrom there is a steady retreat. This can be seen on Fig.18. The retreat in each time step is small but steady till the glacier stabilises 2 km from the present terminus position.

To find the estimated mass loss the mass of the present glacier and the glacier at the first stable point is found and subtracted from each other. This gives the mass losses in Table 2. There is mass loss at all the glaciers except for Nioghalvfjerdsfjorden. This is because at this glacier the time evolution is both retreating and advancing, which indicates that it already is in a stable position, and therefore it does not make sense to find the total ice mass loss for this glacier.

Some of the mass losses are quite big such as the one from Jakobshavn where it is estimated that the mass loss is 3500 km^3. This is because the width of the flowline is ~ 100 km and if the profiles get smaller all the way in there will be a massive mass loss. This can be seen on Fig.22 in appendix.

The estimated mass loss is only for a subsection of the flowline. This means that the estimate is too low since the profile all the way in to divide will get smaller when the terminus position retreats.

A disadvantage of this model is that it cannot evolve the glaciers to advance further out than the present terminus position. This is due to the amount of data that is imported from MATLAB when finding the flowlines. If the glaciers were to advance more bed data from bedmachine *Morlighem et al. (2021)* should have been imported to our python scripts. Luckily none of the glaciers are advancing and for now this is not a problem.

7.6 Outlook

If we were to make the model even more precise climate forcing could be introduced. Different parameters could be included such as sea level rise or melt at terminus from intersection of water and ice. The surface mass balance is the net accumulation and ablation. To make the model more realistic melting at terminus by the water could be taking into account. To this the HIRHAM data could be used even more. This data are annual means of SMB, and with this it is possible to tune the model, so the glacier will be in steady state for the years of non varying SMB. This is done by taking into account that the sea melts away the ice at the ice-ocean interactions when glaciers are terminating in water. This could be done by making a constant melt rate pr. meter of water and ice touching each other. This is added

to the water depth at the terminus. Then this is added to the accumulation rate in the time evolution. All this would make the estimate of ice mass loss more precise.

8 Conclusion

In this project we have tried to explore the use of the perfect plastic approximation in relation to marine outlet glaciers. While simple, the model emulates the physical observations well with only one tuneable parameter τ_y, and with the addition of a second τ_y the range for which the model functions well increase remarkably. The simplicity of the model results of it being computationally easy to use, and therefore scales well with the number of glaciers added. The time evolution used to find the ice mass loss is on the other hand less stable but results in reasonable estimations of the retreat and ice mass loss. Overall it is a good model to give an overview of the dynamics of the marine outlet glaciers and the estimated ice mass loss.

References

AMAP. Snow, Water, Ice and Permafrost in the Arctic (SWIPA). *Arctic Monitoring and Assessment Programme (AMAP), Oslo, Norway. xiv + 269 pp*, 2017.

J. N. Bassis and C. C. Walker. Upper and lower limits on the stability of calving glaciers from the yield strength envelope of ice. *PROCEEDINGS OF THE ROYAL SOCIETY A-MATHEMATICAL PHYSICAL AND ENGINEERING SCIENCES*, 468(2140):913–931, APR 8 2012. ISSN 1364-5021. doi: 10.1098/rspa.2011.0422.

Cuffey and Paterson. *The Physics of glaciers*. Butterworth-Heinemann, Academic Press, 2010.

I. M. Howat, A. Negrete, and B. E. Smith. The Greenland Ice Mapping Project (GIMP) land classification and surface elevation data sets. *CRYOSPHERE*, 8(4):1509–1518, 2014. ISSN 1994-0416. doi: 10.5194/tc-8-1509-2014.

Christine S. Hvidberg. Supervision from Christine. , 2021.

Ian Joughin, Ben E. Smith, and Ian M. Howat. A complete map of Greenland ice velocity derived from satellite data collected over 20 years. *JOURNAL OF GLACIOLOGY*, 64(243):1–11, FEB 2018. ISSN 0022-1430. doi: 10.1017/jog.2017.73.

P. L. Langen, R. H. Mottram, J. H. Christensen, F. Boberg, C. B. Rodehacke, M. Stendel, D. van As, A. P. Ahlstrom, J. Mortensen, S. Rysgaard, D. Petersen, K. H. Svendsen, G. Adalgeirsdottir, and J. Cappelen. Quantifying Energy and Mass Fluxes Controlling Godthabsfjord Freshwater Input in a 5-km Simulation (1991-2012). *JOURNAL OF CLIMATE*, 28(9):3694–3713, MAY 1 2015. ISSN 0894-8755. doi: 10.1175/JCLI-D-14-00271.1.

Signe Hillerup Larsen, Shfaqat Abbas Khan, Andreas Peter Ahlstrom, Christine Schott Hvidberg, Michael John Willis, and Signe Bech Andersen. Increased mass loss and asynchronous behavior of marine-terminating outlet glaciers at Upernavik IsstrOm, NW Greenland. *JOURNAL OF GEOPHYSICAL RESEARCH-EARTH SURFACE*, 121(2):241–256, FEB 2016. ISSN 2169-9003. doi: 10.1002/2015JF003507.

Morlighem et al. IceBridge BedMachine Greenland, Version 3. *Boulder, Colorado USA. NASA National Snow and Ice Data Center Distributed Active Archive Center*, 2021. doi: 10.5067/VLJ5YXKCNGXO.

Ruth Mottram, Sebastian B. Simonsen, Synne Hoyer Svendsen, Valentina R. Barletta, Louise Sandberg Sorensen, Thomas Nagler, Jan Wuite, Andreas Groh, Martin Horwath, Job Rosier, Anne Solgaard, Christine S. Hvidberg, and Rene Forsberg. An Integrated View of Greenland Ice Sheet Mass Changes Based on Models and Satellite Observations. *REMOTE SENSING*, 11(12), JUN 2 2019. doi: 10.3390/rs11121407.

Faezeh M. Nick, Andreas Vieli, Morten Langer Andersen, Ian Joughin, Antony Payne, Tamsin L. Edwards, Frank Pattyn, and Roderik S. W. van de Wal. Future sea-level rise from Greenland's main outlet glaciers in a warming climate. *NATURE*, 497(7448):235–238, MAY 9 2013. ISSN 0028-0836. doi: 10.1038/nature12068.

Shepherd et al. Mass balance of the Greenland Ice Sheet from 1992 to 2018. *NATURE*, 579(7798): 233+, MAR 2020. ISSN 0028-0836. doi: 10.1038/s41586-019-1855-2.

Lizz Ultee and Jeremy Bassis. The future is Nye: an extension of the perfect plastic approximation to tidewater glaciers. *JOURNAL OF GLACIOLOGY*, 62(236):1143–1152, 2016. ISSN 0022-1430. doi: 10.1017/jog.2016.108.

Lizz Ultee and Jeremy N. Bassis. SERMeQ Model Produces a Realistic Upper Bound on Calving Retreat for 155 Greenland Outlet Glaciers. *GEOPHYSICAL RESEARCH LETTERS*, 47(21), NOV 16 2020a. ISSN 0094-8276. doi: 10.1029/2020GL090213.

Lizz Ultee and Jeremy N. Bassis. Supporting information for ¨SERMeQ¨. *GEOPHYSICAL RESEARCH LETTERS*, 2020b.

Appendix

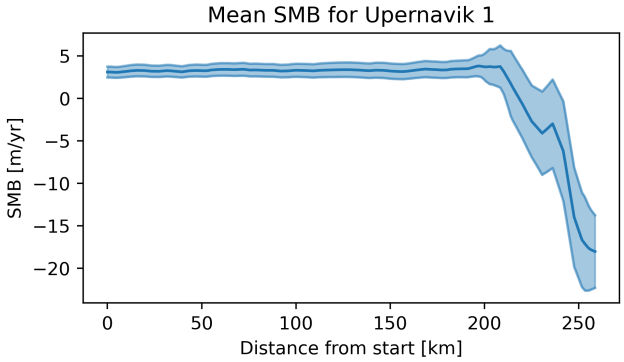


Figure 19: Mean SMB values with uncertainties along the Upernavik 1 flowline

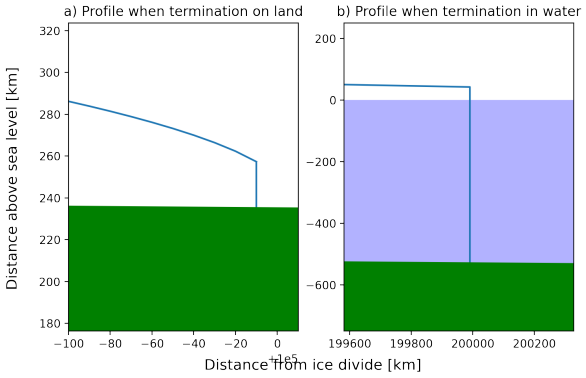


Figure 20: Here are profile from glaciers terminating in water and on land zoomed in om the terminus region. Both are modeled on a linear bed.

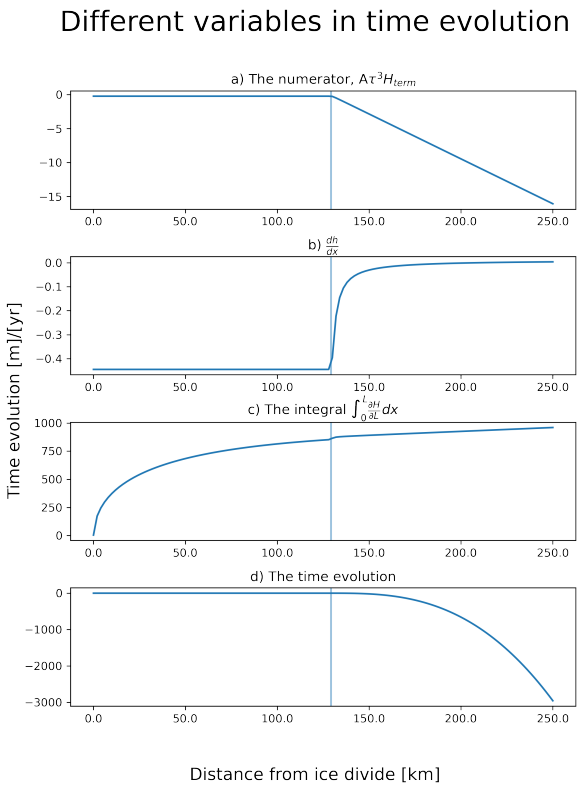


Figure 21: Here is the terms from the time evolution on a linear bed with no accumulation. The thee upper plots are of the individual terms and the last is of the overall trend.

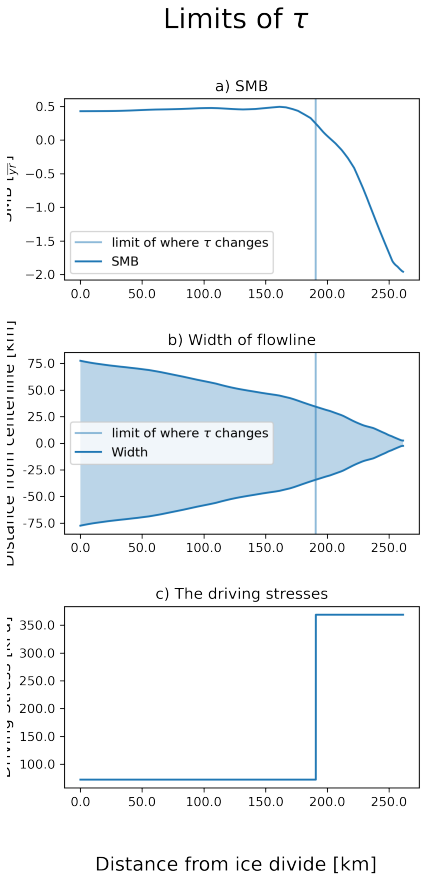


Figure 22: Here are the SMB, width and estimated τ. Here the changing τ values looks like it depends on the varying SMB

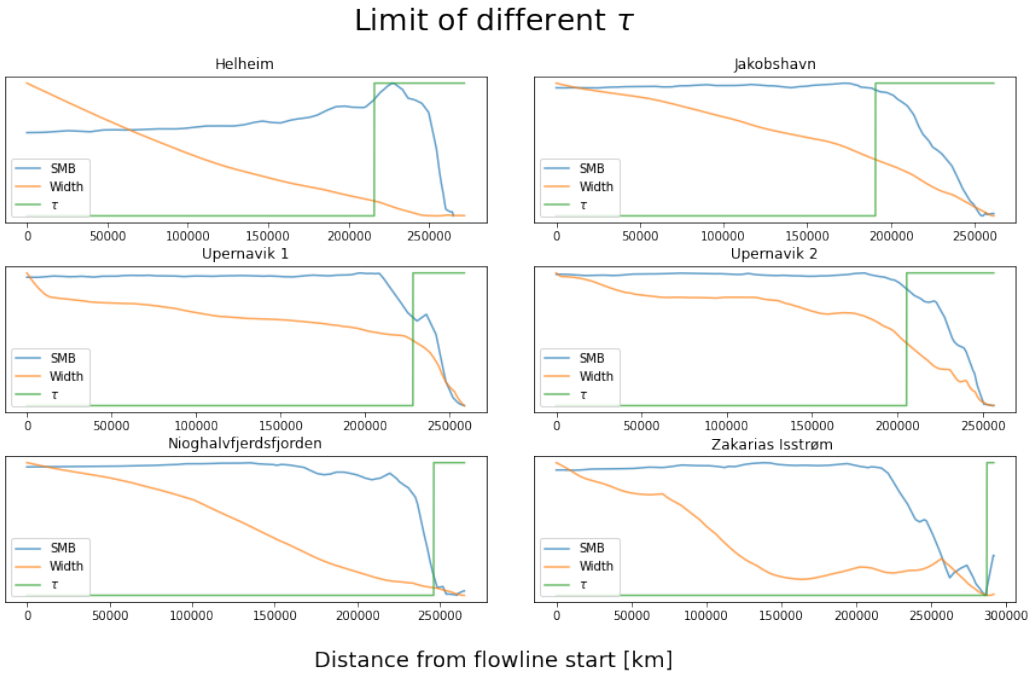


Figure 23: Here the SMB, τ and width of the six glaciers found. All the variables have been normalized to make all parameters visible in the same plot. As it can be seen on the plot from Upernavik the trend is overall that the best driving stress value changes with the width and SMB.

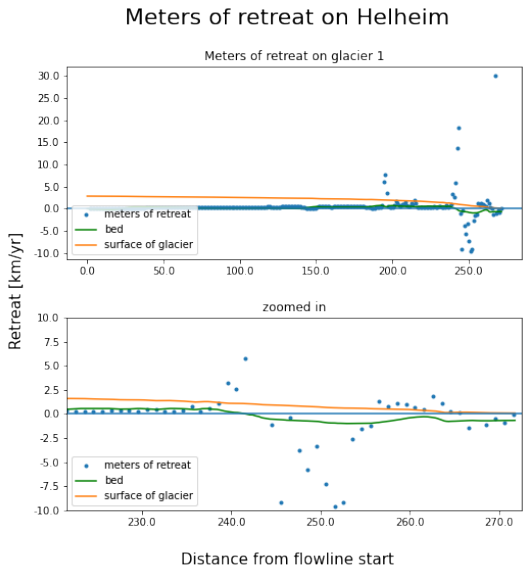


Figure 24: Here is the time evolution from Helheim.

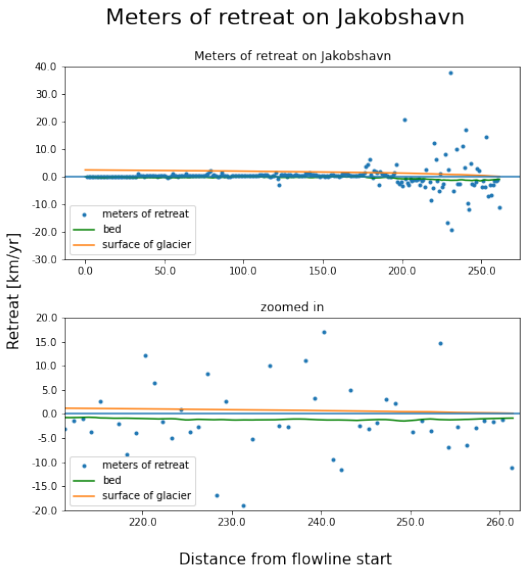


Figure 25: Here is the time evolution from Jakobshavn.

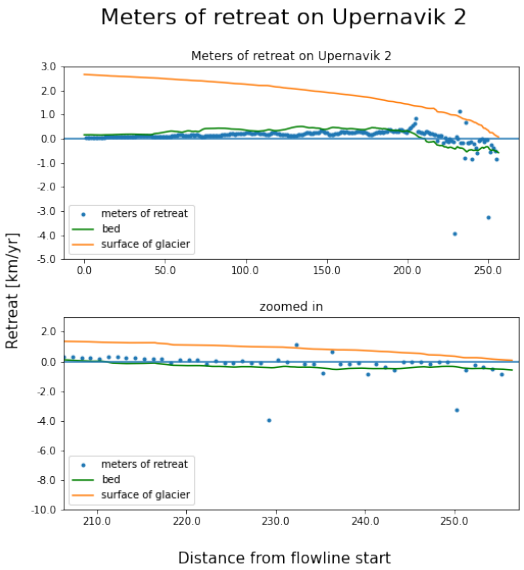


Figure 26: From Upernavik 2

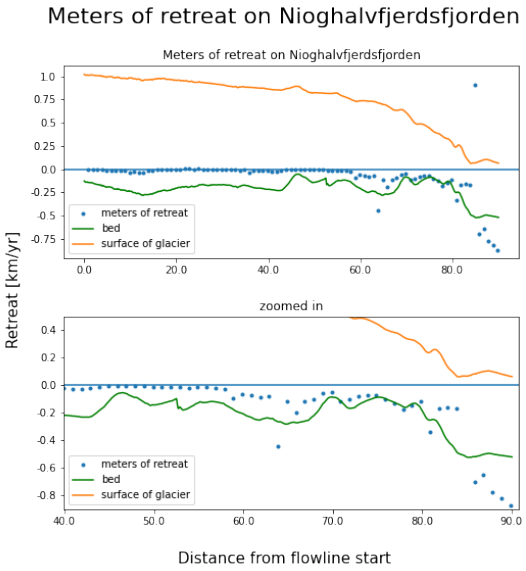


Figure 27: From Nioghalvfjerdsfjorden. Here all most all of the points makes retreat.

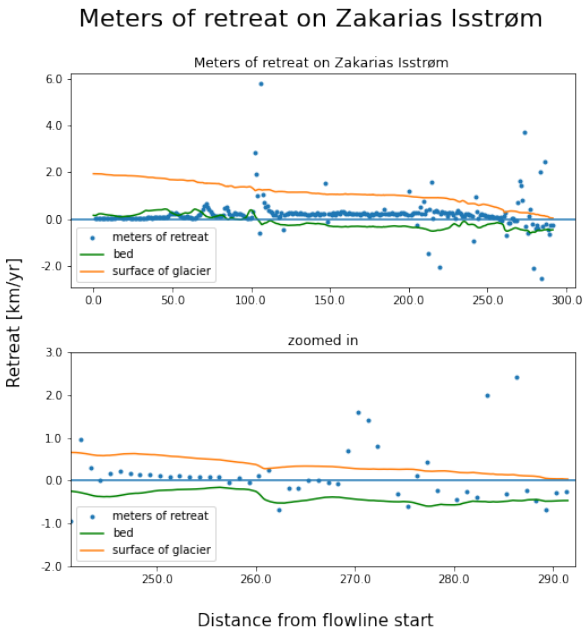


Figure 28: From Zachariae Isstrom

Python scripts

Functions needed to find the profile, and time evolution:

```
#============================
# Data
# Input:
# Output:
# => Array: Ty,
# => Int: n
# => List: data,functions
#============================
def Init(Name = 0):
    if Name == 0:
        Name = int(input("# Galcier: "))
    dx = 10
    TY = [  [ 274100, 95100, 56000  ],[ 370100, 98500, 30400  ],[ 264200, 97800, 51100  ],[ 368600, 72400, 70700  ],[ 112600, 72500, 18600 ],[ 66600, 66500, 4600  ]  ]
    n,data,f = Format_Data(Filename = 'FLflowlinehighres{}.mat'.format(Name),dx = dx)
    TY = TY[Name-1]
    TY = Ty(TY[0] ,TY[1] , TY[2] , n )
    return TY,n,data,f

#============================
# Data
# Input:
# => String: Filename
# => int: dx
# Output:
# => Int: n
# => List: data,functions
#============================
def Format_Data(Filename,dx = 10):
    Data = scipy.io.loadmat(Filename)
    x = Data['FLdist'].flatten()
    x = x[-1]-x
    b = Data['FLbed'].flatten()
    h = Data['FLsurf'].flatten()
    a = Data['FLsmb'].flatten()
    w = Data['FLwidth'].flatten()

    x = x[::-1]
    b = b[::-1]
    h = h[::-1]
    a = a[::-1]
    w = w[::-1]

    fb = interpolate.interp1d(x,b)
    fh = interpolate.interp1d(x,h)
    fa = interpolate.interp1d(np.delete(x,np.where(np.isnan(a))) , np.delete(a,np.where(np.isnan(a))) , fill_value = 'extrapolate' )
    fw = interpolate.interp1d(x,w)

    x = np.arange(x[0],x[-1],dx)
    n = len(x)
    b = fb(x)
    h = fh(x)
    a = fa(x)
    w = fw(x)
    return n,[x,b,h,a,w],[fb,fh,fa]

#============================
# Calculate ice profile
# Input:
# => Array: x,b,ty
# => Float: h0
# Returns:
# => Array: h
#============================
def profile(X,B,h0,Ty):
    x = X[::-1]
    b = B[::-1]
    ty = Ty[::-1]
    h = np.zeros(len(x))
    h[0] = h0
    for i in range(len(x)-1):
        i += 1
        if h[i-1]<b[i-1]:
            print("Error: Hit bottom")
            break
        else:
            dx = x[i] - x[i-1]
            h[i] = -ty[i-1] / (rho*g*(h[i-1]-b[i-1]) ) * dx + h[i-1]
    return h[::-1]

#============================
# Calculate terminus height
# Input:
# => Float: b,ty
# => (Array: b,ty)
# Returns:
# => Float: h_term
# => (Array: h_term)
#============================
def H_term(b,ty):
    if type(b) is np.ndarray:
        D = np.zeros(len(b))
        D[b<0] = -b[b<0]
    else:
        D = 0
        if b<0:
```

```
            D = -b
    h_term =  ty/(rho*g) + np.sqrt((ty/(rho*g))**2 + D**2*rho_w/rho)
    return h_term

#============================
# Calculate dl/dt in the i'th point
# Input:
# => Array: ty, b, x, a
# => Int: i
# => Float: forcing
# Returns:
# => Float: dl/dt,Numerator,Denominator
#============================
def dldt(ty,x,b,a,i,forcing):
    dx = x[1]-x[0]
    n =10
    A = 3.5*10**(-25)*365*24*3600
    d = 0
    aa = 0
    if b[i]<0:
        aa = -forcing
        d = -b[i]*aa

    alp = np.sum(dx * a[:i]) + d

    h = profile(x[:i], b[:i] , H_term(b[i],ty[i]) + b[i],ty[:i])
    h_p = profile(x[:i],b[:i],H_term(b[i],ty[i]) + b[i],ty[:i])
    h_m = profile(x[:i-2*n],b[:i-2*n], H_term(b[i-2*n],ty[i-2*n])+ b[i-2*n],ty[:i-2*n])

    dhdx = ((h[-1]-b[i]) - ( np.mean(h[-(n*2+1):-1]) -np.mean(b[i-n*2:i])))/(x[i]-np.mean(x[i-n*2:i] ))

    dhydx = (- np.mean(  H_term(b[i-n:i], ty[i-n:i]) )+ np.mean( H_term(b[i:i+n],ty[i:i+n]) ) )/(-np.mean(x[i-n:i]) +  np.mean(x[i:i+n] ))

    dhdl = np.sum( ( ( -h_m +h_p[:-2*n] ) /(2*n*dx) )*dx)

    aa =  a[i] + aa - A*ty[i]**3*H_term(b[i],ty[i]) - alp * dhdx/H_term(b[i],ty[i] )
    bb = dhydx - dhdx*(1+dhdl/H_term(b[i],ty[i]))
    DLDT = aa/bb
    return DLDT,aa , bb

#============================
# Calculate ty
# Input:
# => Float: ty0, ty1
# => Int: i,n,dx
# Returns:
# => Array: ty
#============================
def Ty(ty0,ty1,i,n,dx = 10):
    i = int(i/dx)
    ty = np.ones(n)*ty1
    ty[n-i:] = ty0
    return ty
```

Another of those:

```
# -*- coding: utf-8 -*-
"""
Functions for the profile and the time evolution

"""

import matplotlib.pyplot as plt
import scipy.io as sio
from scipy import interpolate
import numpy as np

###Constants########
rho_i = 917
rho_w = 1027
g = 9.82
A = 3.5*10**(-25)*356*3600*24
dx =  10


###### Import of data ###########
#res = int(input('number?')) #Ask in the other program which glacier we are looking at

def Data(res): # It is only in here tau-values and limit-values have to be changed

    if res == 1: #Helheim
        data = sio.loadmat('FLflowlinehighres1',appendmat=True)
        ty = [274100,95100]
        grense = 5600
    elif res == 2: #Upernavik 1
        data = sio.loadmat('FLflowlinehighres2',appendmat=True)
        ty = [370100,98500]
        grense = 3040
    elif res ==3: #Upernavik 2
```

```
        data = sio.loadmat('FLflowlinehighres3',appendmat=True)
        ty = [264200,97800]
        grense = 5110
    elif res == 4: # Jakobshavn
        data = sio.loadmat('FLflowlinehighres4',appendmat=True)
        ty = [368600,72400]
        grense = 7070
    elif res == 5: #nioghalvfjerds
        data = sio.loadmat('FLflowlinehighres5',appendmat=True) # in range [100:3333]
        ty = [112600,72500]
        grense = 1860
    elif res == 6: # Zakarias
        data = sio.loadmat('FLflowlinehighres6',appendmat=True)
        ty = [66.6*10**3,66.5*10**3]
        grense = 460

    bed_dat = data['FLbed'].flatten()# make data one-dimensional
    surf = data['FLsurf'].flatten()
    dist = data['FLdist'].flatten()
    smb = data['FLsmb'].flatten()
    width = data['FLwidth'].flatten()

    x = dist[-1]-dist
    x = x[::-1]
    surf  = surf[::-1]
    smb = smb[::-1]
    width = width[::-1]
    bed_dat = bed_dat[::-1]


    #Making funktions with linear interpolation.
    # To make smaller steps
    #Removes 'nan' and extrapolates
    funk_bed = interpolate.interp1d(x, bed_dat)
    funk_surf = interpolate.interp1d(x, surf)
    funk_smb = interpolate.interp1d(np.delete(x,np.where(np.isnan(smb))),np.delete(smb,np.where(np.isnan(smb))),fill_value = 'extrapolate')
    funk_width = interpolate.interp1d(x, width)

    x = np.arange(x[0],x[-1],dx)
    b = funk_bed(x)
    surf_dat = funk_surf(x)
    a = funk_smb(x)
    bredde = funk_width(x)
    n = len(b)

    return x,b,surf_dat,a,bredde,ty,grense,n


########### Definitions of functions for making the profile ###########

def h_term(b,ty): #finding terminus height, tau has two values
    D = 0 # Sealevel is 0
    if b<0: #if terminating in water
        D = -b
    h_term = ty/(rho_i*g)+np.sqrt((ty/(rho_i*g))**2+D**2*rho_w/rho_i)
    return h_term

def h0(ty, x_start): # x_start is the x position to calculate starting height
    h0 = h_term(x_start, ty)+ x_start
    return h0

############## The profile ###########
def hojde(h_start,b, x, ty, gr , n ): # calculating profile.
    x = x[::-1] #From divide and out
    b = b[::-1]
    N = len(b)
    h = np.zeros(N)
    h[0]=h_start

    for i in range(N-1):
        if x[i] <(n-gr)*dx: #x-axis reversed, therefore new 'gr', to make tau
                                #change at the right spot
            Tau = ty[1]
        else:
            Tau =ty[0]
        i += 1

        if h[i-1]<=b[i-1]:
            h[i] = b[i]
            print('bottom') # To see if the glacier reach the bed.
            break

        else:
```

```
            h[i] = -Tau/(rho_i*g*(h[i-1]-b[i-1]))*(x[i]-x[i-1]) + h[i-1]
    return h[::-1]

######## The time evolution ####################

def dldt(b,x,i,ty,a,gr,n, forcing):
    D = 0
    nx= 10 # the amount of step to each side. To make average -> more precise.
    a1 = 0
    if b[i]<0: # If forcing, then the SMB would be smaller,
        a1 = -forcing # when terminating in water
        D = -b[i]*a1# this is not included.

    alpha = np.sum(dx*a[:i]) + D # D = 0 when no forcing

    Ty = np.ones(n)*(ty[1]) # Here tau is a list
    Ty[int((x[-1]-gr*dx)/dx):] = (ty[0])

    #if x[i]<(len(b)-gr)*dx: #For when tau should be a number and not a list
     #   Ty = ty[1]
    #else:
     #   Ty = ty[0] #ret til 0 bu[i-2]


    profil = hojde(h0(int(Ty[i]), b[i]),b[:i],x[:i],ty,gr,n)
    dhy = (-h_term(np.mean(b[i-(nx):i]), int(Ty[i]))+
           h_term(np.mean(b[i:i+nx]),int(Ty[i])))

    dhydx = dhy/(x[i]-np.mean(x[i-2*nx:i]))
    dhdx = (+profil[-1]-b[i]-np.mean(profil[-(2*nx+1):-1])
            +np.mean(b[i-(2*nx):i]))/(x[i]-np.mean(x[i-2*nx:i]))

    H_term = h_term(b[i],Ty[i])

    dh = -hojde(h0(Ty[i],b[i-2]),b[:i-2],x[:i-2],ty,gr,n)\
         +hojde(h0(Ty[i],b[i]),b[:i],x[:i],ty,gr,n)[:-2]


    dl = dx*2
    dhdl = dh/dl*dx
    inte = np.sum(dhdl)

    aa = (a[i]+a1-A*Ty[i]**3*H_term-(alpha/H_term)*dhdx)
    bb = (dhydx-dhdx*(1+inte/H_term))
    dldt = (aa/bb)
    return dldt,aa,bb, dhdx


def l_udvikling(N,b,x,ty,a,gr,n, forcing = 0):
    i = np.zeros(N)
    i[0] = n - 10

    dx = x[1]-x[0]
    for j in range(N-1):
        j += 1
        DLDT = dldt(b,x,int(i[j-1]),ty,a,gr,n, forcing)[0]*0.5
        print(DLDT)
        if DLDT<10 and DLDT>5:
            DLDT = 10
        elif DLDT>-10 and DLDT<-5:
            DLDT = -10
        elif DLDT <-10000 and DLDT>10000:
            DLDT = 200*(np.random.rand()-0.5)
        i[j] = int(i[j-1] + DLDT/dx)
        if i[j]>n or i[j]<0:
            print('Error')
            break

    i = i.astype(int)
    plt.figure()
    plt.plot(x[i])
    return i
```

Script that finds the optimal values for τ_1, τ_2 and i:

```
import numpy as np
import matplotlib.pyplot as plt
from  Functions_v0 import *
plt.close('all')
plt.ion()

ty,n,data,f =Init()
x,b,h,a,w = data[0],data[1],data[2],data[3],data[4]
Name  = int(input("Again: "))

def Best(Ty0,Ty1,II,d_i,d_ty):
    diff = 10**100

```

```
    I,J,K = 1,1,1
    while I != 0 and J != 0  and K != 0:
        ty0 = Ty0
        ty1 = Ty1
        ii = II
        I,J,K = 0,0,0
        for i in range(3):
            i -= 1
            for j in range(3):
                j -= 1
                for k in range(3):
                    k -= 1

                    T0_t = ty0 + i*d_ty
                    T1_t = ty1 + j*d_ty
                    ii_t = int(ii + k*d_i
                        )

                    ty = Ty(T0_t,T1_t,
                        ii_t,n)
                    h0 = H_term(b[-1],ty
                        [-1]) + b[-1]
                    h_forward = profile(x
                        ,b,h0,ty)
                    Diff = np.mean((
                        h_forward-h)**2)
                    if Diff < diff:
                        diff = Diff
                        I = i
                        J = j
                        K = k
        Ty0 = I*d_ty + ty0
        Ty1 = J*d_ty + ty1
        II = K * d_i + ii
        print('|')
    return Ty0,Ty1,II, int(diff)

def BestI():
    diff = 10**100
    Ty0 = 10000
    d_ty = 10000
    I = 1
    while I != 0:
        ty0 = Ty0
        I = 0
        for i in range(2):
            T0_t = ty0 + i*d_ty
            ty = Ty(200000,100000,T0_t,n)
            h0 = H_term(b[-1],ty[-1]) + b
                [-1]
            h_forward = profile(x,b,h0,ty
                )
            Diff = np.mean((h_forward-h)
                **2)
            if Diff < diff:
                diff = Diff
                I = i
        Ty0 = I*d_ty + ty0
        print('|')
    return Ty0

Ty0 = 250000
Test = np.array([269,389,292,411,114,71])
    *1000
Ty0 = Test[Name-1]
Ty1 = 100000
II  = 50000
II = BestI()
d_ty = 10000
d_i = 10000
print(Ty0,Ty1,II)

for i in range(3):
    Ty0,Ty1,II,diff  =  Best(Ty0,Ty1,II,
        d_i,d_ty)
    print(Ty0,Ty1,II,diff)
    d_ty = int(d_ty/10)
    d_i = int(d_i/10)

ty = Ty(Ty0,Ty1,II,n)
h0 = H_term(b[0],ty[0]) + b[0]
h_forward = profile(x,b,h0,ty)

plt.plot(b)
plt.plot(h)
plt.plot(h_forward)

T(t)
```

And

```
# -*- coding: utf-8 -*-
"""
For finding tau values:

"""
import numpy as np
import scipy.io as sio
from Functions import*

rho_i = 917
rho_w = 1027
g = 9.82
A = 3.5*10**(-25)*356*3600*24


glacier = (input('number?'))
x,b,surf,a,bredde,ty,gr,n = Data(int(
    glacier))
# Takes data for the glacier, here is
    guessed yield stresses and limits.
# this scripts take guessed yield
    stresses and optimizes them.
```

```

def ty_i(grense): #Finding tau at terminus,
    tyy = 100*10**3 # starts from 50 kPa
    tyy2 = 100*10**3 # A guess for the inner tau
    for i in range(600): # have to be able to reach 300 kPa
        diff1 = (hojde(h0(tyy,b[0]), b,x, ty = [tyy,tyy2], gr=gr,n=n) -surf)[:grense]
        diff2 = (hojde(h0((tyy+1*10**3),b[0]), b,x, ty = [tyy+1*10**3,tyy2],gr=gr,n=n)-surf)[:grense]
        fejl1 = np.sqrt(np.sum(diff1**2))
        fejl2 = np.sqrt(np.sum(diff2**2))
        if fejl2 < fejl1:
            tyy += 1*10**3 #adds 1 kPa to tau
        else:
            break
    return tyy

fejl = np.sqrt(np.sum((hojde(h0(ty[0],b[0]), b,x, ty = ty,gr = gr,n= n) -surf)**2))
tyi = ty_i(gr)#finds best tau for a chosen limit.


def ty_f(grense):
    ty1 = tyi
    tyy2 = 50*10**3 # starts from 50 kPa
    ty_2=np.empty([])
    for i in range(600):
        diff1 = (hojde(h0(tyi,b[0]), b,x, ty = [tyi,tyy2], gr= gr, n= n) -surf)[grense:]
        diff2 = (hojde(h0(tyi,b[0]), b,x, ty = [tyi,tyy2+1*10**3],gr= gr,n=n)-surf)[grense:]
        fejl1 = np.sqrt(np.sum(diff1**2))
        fejl2 = np.sqrt(np.sum(diff2**2))
        if fejl2 < fejl1:
            tyy2 += 1*10**3
        else:
            ty_2 = np.append(ty_2,tyy2)
            break
    return ty_2[1]
tyf = ty_f(gr)#finds best tau for a chosen limit on the inner parts.

ty = [tyi,tyf] #new tau values
```

Script used to estimate retreat, and to find mass loss:

```
import numpy as np
import matplotlib.pyplot as plt
from  Functions_v0 import *
plt.close('all')
plt.ion()


ty,n,data,f =Init()
x,b,h,a,w = data[0],data[1],data[2],data[3],data[4]

#=============
T()
#=============
N = 150
i = np.zeros(N)
i[0] =   n - 700 # Normalt 10

dx = x[1]-x[0]
for j in range(N-1):
    j += 1
    DLDT = dldt(ty,x,b,a,int(i[j-1]-1), forcing = 0 )[0]
    DLDT += dldt(ty,x,b,a,int(i[j-1]+0), forcing = 0 )[0]
    DLDT += dldt(ty,x,b,a,int(i[j-1]+1), forcing = 0 )[0]
    DLDT = DLDT/3*0.1

    print(j,DLDT)
    if DLDT<10 and DLDT>0:
        DLDT = 10
    elif DLDT>-10 and DLDT<0:
        DLDT = -10
    elif DLDT > 1000 or DLDT < -1000:
        DLDT = 200*(np.random.rand()-0.5)

    i[j] = int(i[j-1] + DLDT/dx)
    if i[j]>n or i[j]<0:
        print('Error')
        break

i = i.astype(int)
plt.figure()
plt.plot(x[i],'.')

#=============
T(t)
```

```
#==============

start = i[0]
end = i[-1]

dx = x[1]-x[0]

Max = np.max(i[-30:])
Min = np.min(i[-30:])

if Max == Min:
    Max += 1
    Min -= 1

def MassChange(start,end):
    Mass_Start =np.sum( (profile(x[:start],b[:start],H_term(b[start],ty[start])+b[start],ty[:start])   -b[:start]) * w[:start] * dx)
    Mass_End =np.sum( (profile(x[:end],b[:end],H_term(b[end],ty[end])+b[end],ty[:end])   -b[:end]) * w[:end] * dx)

    DM = Mass_End - Mass_Start
    DM = DM*10**(-9)
    print(DM)
    return DM

Min =MassChange(start,Min)
Max =MassChange(start,Max)
Stop= MassChange(start,end)

Mean = (Max+Min)/2

std = np.sqrt((Mean-Max)**2 + (Mean-Min)**2 )

print(Mean, "pm" ,std)
```

Python script that calculates the driving stress for the whole of Greenland:

```
import numpy as np
import matplotlib.pyplot as plt
import netCDF4 as nc
import matplotlib.colors as colors

plt.ion()
plt.close('all')

ds = nc.Dataset("BedMachineGreenland-2017-09-20.nc")

h = ds["surface"][:]
mask = ds["mask"][:]  #(0 = ocean, 1 = ice-free land, 2 = grounded ice, 3 = floating ice, 4 = non-Greenland land)
b = ds["bed"][:]
H = ds["thickness"][:]

rho = 917
g = 9.82

shift = 10

dx = np.roll(h,shift,axis =0)- np.roll(h,-shift,axis = 0)
dx = dx.astype(int)/(150*2*shift)
dy = np.roll(h,shift,axis =1)- np.roll(h,-shift,axis = 1)
dy = dy.astype(int)/(150*2*shift)

alpha = dx**2 + dy**2
alpha = np.sqrt(alpha)

tau = rho*g*H*alpha
tau[mask != 2] = 0

plt.imshow(tau)
plt.colorbar()

plt.show()
```