

# Firn densification model

Cummings, Evan

Davis, Tyler

Brinkerhoff, Douglas

August 28, 2012



# 1 Introduction

The top layer of snow on a glacier or ice sheet increases in density as depth increases; newly accumulated snow builds up and compresses the layers below. *Herron and Langway* [1980] developed a firn densification model based on Arrhenius-type equations with variable rate constants, and found that the densification rate decreased suddenly around  $550 \text{ kg m}^{-3}$ . *Zwally and Li* [2002] expanded upon this model and found an alternate temperature-dependent value for the rate constant. *Arthern et al.* [2010] developed yet another set of equations based from their in situ measurements of Antarctic snow compaction. *Ligtenberg et al.* [2011] modified the *Arthern et al.* [2010] parametrization to better fit areas with a higher average annual temperature.

We have re-created a number of these models and integrated them with an enthalpy-formulation proposed by *Aschwanden et al.* [2012] which accounts for melting of firn layers in percolation zones. The model simulates ice lenses formed from this melting of snow within the column, and work is currently being done to allow the movement of water through the column. The model has been created with the finite-element software package FEniCS; an explanation of its usage and flexibility will be made clear.

# 2 Temperature Solution

We begin with the standard heat-transport equation as explained by *Patterson* [2001]

$$\rho c_i \frac{\partial T}{\partial t} = k_i \frac{\partial^2 T}{\partial z^2} + \left( \frac{dk_i}{dt} - \rho c_i w \right) \frac{\partial T}{\partial z}$$

with heat sources from the deformation of ice omitted,  $\rho$  density,  $c_i$  heat capacity,  $k_i$  thermal conductivity,  $w$  vertical velocity, and  $T$  temperature of firn. To solve the total derivative  $dk_i/dt$  we must apply the chain rule

$$\frac{dk_i}{dz} = \frac{\partial k_i}{\partial \rho} \frac{\partial \rho}{\partial z} + \frac{\partial k_i}{\partial T} \frac{\partial T}{\partial z}.$$

The thermal conductivity of ice is defined by *Arthern et al.*, 1998 as

$$k_i = 2.1 \left( \frac{\rho}{\rho_i} \right)^2,$$

and gives

$$\frac{\partial k_i}{\partial \rho} = 4.2 \frac{\rho}{\rho_i^2}$$

and

$$\frac{\partial k_i}{\partial T} = \frac{4.2}{\rho_i^2} \left( \frac{\partial \rho}{\partial T} \right).$$

*Patterson* [2001] defined  $\rho$  in terms of  $T$  and from this can be derived

$$\frac{\partial \rho}{\partial T} = (5.6 \times 10^{-2}) \exp((-5.7 \times 10^{-3})T).$$

The vertical velocity of ice,  $w$ , is directly proportional to the accumulation,  $\dot{b}$ ,

$$w = -\frac{\dot{b}}{\rho}$$

with  $\dot{b}$  in units of  $\text{kg m}^{-2} \text{ s}^{-1}$ . The densification process is defined with the material derivative

$$\frac{d\rho}{dt} = \frac{\partial \rho}{\partial t} + w \frac{\partial \rho}{\partial z}.$$

*Arthern et al.* [2010] described this derivative differently for density values above and below a critical value,  $\rho_m$ :

$$\frac{d\rho}{dt} = \begin{cases} c_0(\rho_i - \rho), & \rho \leq \rho_m \\ c_1(\rho_i - \rho), & \rho > \rho_m \end{cases}.$$

*Zwally and Li* [2002] defined a single multiplying constant  $c$  with an Arrhenius-type relation

$$c_0 = c_1 = \dot{b}\beta(T) \left( \frac{\rho_i}{\rho_w} \right) K_{0G}(T) \exp\left(-\frac{E(T)}{RT}\right),$$

with  $K_{0G}(T) \exp(-E(T)/(RT)) = 8.36T^{-2.061}$  as described in *Reeh* [2008], and  $\beta(T)$  a smoothing function to match a desired density rate. *Arthern et al* [2010] developed a semi-empirical formula by coupling the rate equations for Nabarro-Herring creep and normal grain-growth:

$$\begin{cases} c_0 = M_0 \dot{b} g^{\frac{k_{c0}}{k_g}} \exp\left(-\frac{E_c}{RT} + \frac{E_g}{RT_{avg}}\right) \\ c_1 = M_1 \dot{b} g^{\frac{k_{c1}}{k_g}} \exp\left(-\frac{E_c}{RT} + \frac{E_g}{RT_{avg}}\right) \end{cases},$$

with the creep coefficients defined as

$$\begin{cases} k_{c0} = 9.2 \times 10^{-9} \text{ m}^3 \text{ s kg}^{-1} \\ k_{c1} = 3.7 \times 10^{-9} \text{ m}^3 \text{ s kg}^{-1} \end{cases}$$

and  $M$  defined in *Ligtenberg et al.* [2011] to better fit with observed densification rates in higher-temperature environments:

$$\begin{cases} M_0 = 2.366 - 0.293 \ln(\dot{b} * 1 \times 10^3) \\ M_1 = 1.435 - 0.151 \ln(\dot{b} * 1 \times 10^3) \end{cases}.$$

Within the same paper a firn surface density expression from data was given:

$$\rho_s = -151.94 + 1.4266(73.6 + 1.06T_s + 0.0669A + 4.77V_a).$$

### 3 Enthalpy Solution

As stated in *Aschwanden et al.* [2012], we take 'enthalpy' to be synonymous with 'internal energy' due to the exclusion of work done with changing volume. The equation used here is the shallow-enthalpy:

$$\rho \frac{\partial H}{\partial t} = \frac{\partial}{\partial z} \left( \left\{ \begin{array}{cc} K_i, & \text{Temperate} \\ K_0, & \text{Cold} \end{array} \right\} \frac{\partial H}{\partial z} \right) + w \rho \frac{\partial H}{\partial z}.$$

Strain heating has been neglected and the advective term  $w \rho \partial H / \partial z$  has been added. The coefficient for temperate ice is

$$K_i = \frac{k_i}{c_i}.$$

The coefficient for cold ice is

$$K_0 = \frac{1}{10} K_i.$$

Temperate firm is defined as firm with  $H > H_s$  and cold firm  $H \leq H_s$ , with

$$H_s = \int_{T_0}^{T_m} c_i(T) dT$$

where  $T_m = 273.15$  K and  $T_0 = 0.0$  K. The enthalpy can be found with a constant heat capacity of  $2009 \text{ J kg}^{-1} \text{ K}^{-1}$  with the linear equation

$$H = \begin{cases} c_i(T - T_0), & \text{where } T \leq T_m \\ c_i(T_w - T_0) + \omega L_f, & \text{where } T > T_m \end{cases}$$

where  $L_f$  is the latent heat of fusion and  $\omega$  represents the water content percentage of firm given by

$$\omega L_f = H - c_i(T_w - T_0).$$

Temperature may be derived from enthalpy easily:

$$T = \frac{H}{c_i}.$$

The density equations remain the same as the temperature solution, and the surface density at time index  $n + 1$  can be described as:

$$\rho_s^{n+1} = \rho_b^{n+1} d_p + \rho_s^n (1 - d_p),$$

where

$$\rho_b^{n+1} = \rho_b^{n-1} + \Delta \omega_s \rho_b^n,$$

$$\Delta \omega_s = \omega_s^n - \omega_s^{n-1},$$

$$d_p = \frac{d_n}{l_s},$$

$$d_n = w_s dt, \text{ and}$$

$l_s$  is the length of the surface node.

If  $T_s \geq T_w$ , the density of surface and column snow with re-freezing simulated is given by

$$\rho_b^n = \begin{cases} \rho_w - \rho_i \text{ kg m}^{-3}, & \Delta \omega \leq 0 \\ \rho_w \text{ kg m}^{-3}, & \Delta \omega > 0 \end{cases},$$

and when  $T_s < T_w$ ,  $\rho_b^n = \rho_{si}$ .

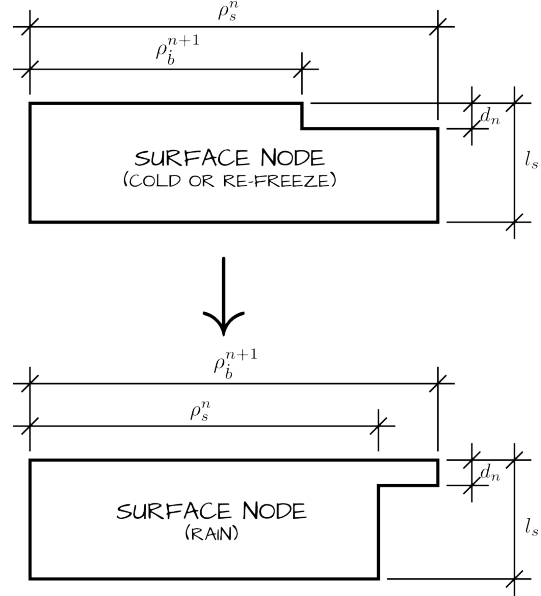


Figure 1: Evolution of surface node density

### 4 Finite Element Method

This section will focus on the enthalpy solution. Solving these equations with FEniCS is done with Galerkin's method and requires finding the weak formulation:

$$0 = \int_{\Omega} \left\{ \begin{array}{c} K_i \\ K_0 \end{array} \right\} \nabla^2 H \psi \, d\Omega + \int_{\Omega} w \rho \nabla H \psi \, d\Omega - \int_{\Omega} \rho \frac{\partial H}{\partial t} \psi \, d\Omega.$$

Here we have integrated over the entire domain,  $\Omega$ , and multiplied by a test function  $\psi$ . After integrating the diffusive term by parts this becomes

$$f_H = - \int_{\Omega} \left\{ \begin{array}{c} K_i \\ K_0 \end{array} \right\} \nabla H \nabla \psi \, d\Omega + \int_{\Omega} w \rho \nabla H \psi \, d\Omega - \int_{\Omega} \rho \frac{\partial H}{\partial t} \psi \, d\Omega.$$

We can discretize the enthalpy time-differential with the second-order accurate backward-difference formula

$$\frac{\partial H}{\partial t} = \frac{H^{k-2} - 4H^{k-1} + 3H^k}{2dt},$$

with superscripts referring to time index. This equation can be represented in FEniCS as:

$$\begin{aligned} f\_H = & \text{rho} * (H\_2 - 4 * H\_1 + 3 * H) / (2 * dt) * \text{psi} * dx \\ & + k / c * K\text{coef} * \text{inner}(\text{grad}(H), \text{grad}(\text{psi})) * dx \\ & + \text{rho} * w * \text{grad}(H) * \text{psi} * dx \end{aligned}$$

The variable `Kcoef` is a coefficient vector which will be updated dynamically depending on the temperature of firm (either 1.0 or 0.1).

The weak form for density is found similarly:

$$f_\rho = \int_{\Omega} \frac{\partial \rho}{\partial t} \phi \, d\Omega + \int_{\Omega} w \nabla \rho \phi \, d\Omega - \int_{\Omega} \frac{d\rho}{dt} \phi \, d\Omega.$$

With the partial-time-differential of density defined identically to the enthalpy equation, this can be represented in FEniCS with the *Arthern et al.* [2010] densification equation as:

$$\begin{aligned} c &= b * g * \text{rhoCoef} / \text{kg} * \\ &\quad \exp(-E_c / (R * T) + E_g / (R * T_{avg})) \\ \text{drhodt} &= c * (\text{rhoi} - \text{rho}) \\ f\_rho &= ((\text{rho}_2 - 4 * \text{rho}_1 + 3 * \text{rho}) / (2 * dt) \\ &\quad - \text{drhodt} + w * \text{grad}(\text{rho})) * \text{phi} * dx \end{aligned}$$

The variable `rhoCoef` is another dynamically-updated-coefficient vector and is either  $k_{c0}$  or  $k_{c1}$  depending upon the density at the node.

We can define the function space for the entire non-linear problem as

$$U = \Omega \times \Omega,$$

with corresponding trial and test functions respectively defined as

$$v, u \subset U.$$

The test functions for each function can now be described as

$$\psi, \phi \subset u.$$

In FEniCS these spaces can be defined by this:

```
mesh = Interval(n, zb, zs)
V = FunctionSpace(mesh, 'Lagrange', 1)
MV = V * V
h = Function(MV)
H, rho = split(h)
dh = TrialFunction(MV)
dH, drho = split(dh)
j = TestFunction(MV)
psi, phi = split(j)
```

The variable `zb` is the z-position of the base of the firm column which does not change and `n` is the number of nodes; the `mesh` variable defines the spacial dimensions of the system to be solved and is here created in one dimension. The mesh may also be created in three dimensions if desired, or made to fit a custom grid.

We define the entire function as

$$f = f_H + f_\rho.$$

Solving this system can be accomplished with *Newton's Method* which requires derivation of the Jacobian:

$$J = \frac{\partial f}{\partial v}.$$

In FEniCS this becomes:

```
f = f_H + f_rho
J = derivative(f, h, dh)
```

## 5 Boundary Conditions

A cyclical enthalpy boundary condition for the surface can be simulated with

$$H_s = c_i(T_s - T_0),$$

$$T_s = T_{avg} + \alpha \sin(\omega t),$$

where  $\alpha$  is the amplitude of temperature variation and  $\omega = 2\pi / \text{spy}$  is the frequency. The surface-density boundary condition can be likewise described as (see figure 1):

$$\rho_s^{n+1} = \rho_b^{n+1} d_p + \rho_s^n (1 - d_p).$$

Both of these can be created with FEniCS by

```
code = 'c*(Tavg + 9.9*sin(omega*t) - T0)'
Hs = Expression(code, c=cp, Tavg=Tavg,
               omega=freq, t=t0, T0=T0)
```

```
code = 'dp*rhon + (1 - dp)*rhoi'
rhoS = Expression(code, rhon=rhosi,
                 rhoi=rhosi, dp=1.0)
```

```
def surface(x, on_boundary):
    return on_boundary and x[0] == zs
```

```
Hbc = DirichletBC(MV.sub(0), Hs, surface)
Dbc = DirichletBC(MV.sub(1), rhoS, surface)
```

Within the time-loop the variables `t`, `rhon`, `rhoi`, and `dp` can be updated as needed.

Now all that is left is to iterate through time and call the `solve` method at each step:

```
solve(f == 0, h, [Hbc, Dbc], J=J)
```

The use of the `solve` function in this way chooses *Newton's Method* by default and minimizes the residual of `f`. The boundary conditions are updated by specifying the list `[Hbc, Dbc]`.

## 6 Model Parameters

Within the time-loop there are a number of parameters which need to be updated. Taking into account conservation of mass, the height  $l$  of each node must be recalculated:

$$l_{new} = l_{ini} \frac{\rho_{ini}}{\rho},$$

where  $\rho_{ini}$  and  $l_{ini}$  are the density and height vectors of the firn column when the system was initialized. With the height of the nodes calculated, the z-positions may be found by iterating through the heights and setting the z vector's corresponding cell equal to the current sum. After successful calculation the FEniCS `mesh` object must have its coordinates refreshed. These tasks may be completed with the following code:

```
lnew      = l*rho_in[index] / firn.rho[index]
zSum      = zb
zTemp     = zeros(n)
for i in range(n)[1:]:
    zTemp[i] = zSum + lnew[i]
    zSum     += lnew[i]
firn.z[index] = zTemp
mesh.coordinates()[:,0] = firn.z
```

The variable `index` is an array of positions corresponding to the correct ordering of the nodes, necessary after mesh refinement.

The height  $s$  at time index  $n$  of the original surface may be calculated as follows:

$$s^n = (z_s - z_b) \frac{s^{n-1} - z_b}{z_s - z_b} + w_s dt.$$

This maintains the relative location of the original surface to the current surface and moves downward proportional to  $w$ . This is accomplished in Python with

```
interp      = interp1d(firn.z[index],
                       firn.w[index],
                       bounds_error=False,
                       fill_value=firn.w[index][0])
zint        = array([firn.origZ])
wOrigZ      = interp(zint)
firn.origZ = (firn.z[index][-1] - zb) *
              (firn.origZ - zb) /
              (zs_0 - zb) + wOrigZ[0] * dt
```

The second index for `firn.z` refers to the surface, `[-1]`, or the base, `[0]`. For all operations it is convenient to store all the state data from the simulation in a class object for ease of access. It was for this purpose the `firn` class was created and contains the signature

```
firn(H, T, rho, omega, w, k, c, z, index)
```

with  $z$  the node z-positions as defined above, `index` the index of re-ordered mesh locations, and all other variables as described earlier in the paper.

The variables for accumulation and surface temperature are the main driving forces in the simulation, and if data from a specific site has been collected may be used in the model by interpolating the data for in increments of  $dt$  and inserting the values into the equations. This may be

accomplished with the `set_local(<numpy array>)` method of the `vector` class, which takes a NumPy array with indexes corresponding to node positions within the `mesh` object as input. If the variable is used in the surface boundary condition, this may be updated within the FEniCS Expression object with the dot operator.

A function has been provided (`set_initial_converge()`) which initializes the density to a previously-derived density. The density may also be initialized to a set of real-world data if desired, and is demonstrated in the temperature equation model, `objModel.py`.

The `plot.py` file contains the class `firn` and the previously undescribed `plot` class. This class uses the plotting package Matplotlib to display the data contained in the `firn` object. The method `plot_all_height()` plot the height history for a group of simulations and is useful for comparing the effects of model parameters.

## 7 Variable Definitions

Many variable are used in this simulation and many do not change. These are defined below:

### Constants :

Var.	Value	Units	Description
$g$	9.81	$\text{m s}^{-2}$	gravitational acceleration
$R$	8.3144621	$\text{J mol}^{-1} \text{K}^{-1}$	gas constant
$spy$	31556926	s	seconds per year
$\rho_i$	917	$\text{kg m}^{-3}$	density of ice
$\rho_w$	1000	$\text{kg m}^{-3}$	density of water
$\rho_m$	550	$\text{kg m}^{-3}$	critical density value
$k_i$	2.1	$\text{W m}^{-1} \text{K}^{-1}$	thermal conductivity of ice
$c_i$	2009	$\text{J kg}^{-1} \text{K}^{-1}$	heat capacity of ice
$L_f$	$3.34 \times 10^5$	$\text{J kg}^{-1}$	latent heat of fusion
$H_s$	$c_i(T_w - T_0)$	$\text{J kg}^{-1}$	Enthalpy of ice at $T_w$
$T_w$	273.15	K	triple point of water
$T_0$	0.0	K	reference temperature
$k_g$	$1.3 \times 10^{-7}$	$\text{m}^2 \text{s}^{-1}$	grain growth coefficient
$E_c$	$60 \times 10^3$	$\text{J mol}^{-1}$	act. energy for water in ice
$E_g$	$42.4 \times 10^3$	$\text{J mol}^{-1}$	act. energy for grain growth

### Model Specific :

Var.	Units	Description
$\rho_{si}$	$\text{kg m}^{-3}$	initial density at surface
$\dot{b}$	$\text{kg m}^{-2} \text{s}^{-1}$	surface accumulation
$A$	$\text{mm a}^{-1}$	surface accumulation
$V_a$	$\text{m s}^{-1}$	mean annual wind speed
$T_{avg}$	K	average annual temperature
$T_s$	K	firm surface temperature
$z_s$	m	surface start z-location
$z_b$	m	firm base z-location
$z_{s0}$	m	previous time-step's surface
$dz$	m	initial z-spacing
$l$	m	vector of node heights
$dt$	s	time-step
$t_0$	s	begin time
$t_f$	s	end-time

## 8 Verification of Program

A converging run of the program was done quite quickly by making  $dt$  equal  $spy$ : this had the effect of producing a steady-state solution. After the density profile converged the density data was saved to a text file, `set_initial-converge()` was called,  $dt$  was set to be  $0.0025 * spy$ , and the simulation was run again. At 10 years  $T_{avg}$  was set to not allow the temperature at the surface to rise above freezing, and thus halt any melting and refreezing. The results are shown in figures 2 and 3.

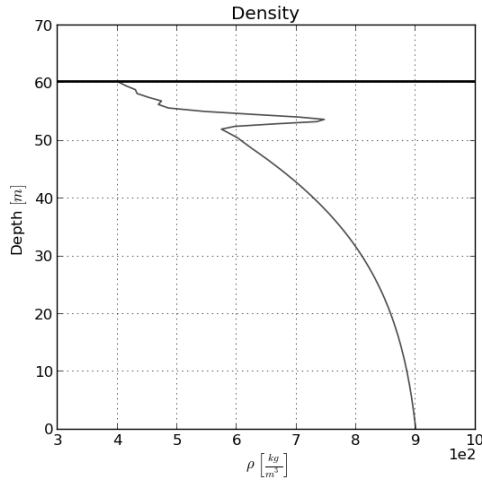


Figure 2: Density profile after 40 years with an ice lens approximately 7 meters below the surface

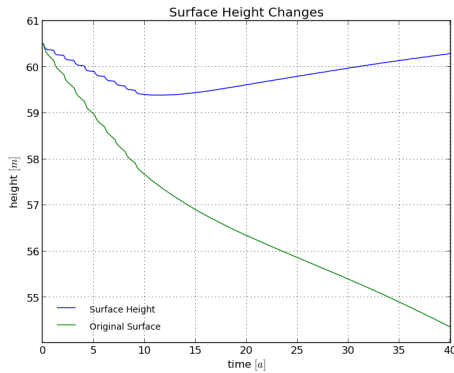


Figure 3: 40-year height history of the column (blue) and original surface (green) resulting in the previous figure. This shows a rapid decrease in height as the lens is formed in the first ten years of simulation. The fluctuations in height show an increase in height with winter temperatures.

Above the ice lens you will see numerical distortion - this is caused by the sudden rise in density where the lens begins. This distortion is a source of inaccuracies and needs to be addressed.

## 9 Interpretation

### 10 Future Developments

At its current state of development the model does not take into account water transport through the firm column, describable with the Darcy flow equations:

$$q = \frac{-k}{\mu} \nabla P, \quad v = \frac{q}{\phi},$$

where  $k$  is permeability,  $\mu$  is the viscosity of water at  $0^\circ$  C,  $\nabla P$  is the pressure gradient vector, and  $\phi$  is porosity.

Waldner *et al.* [2002] introduces this issue and provides references to numerous models which simulate this phenomenon. Coleou *et al.* [1998] supplies an equation for the irreducible water content of snow:

$$S_0 = \frac{0.0057}{1 - \phi} + 0.0017.$$

## References

- [1] Aschwanden, A., Bueler, E., Khroulev, C., and Blatter, H. 2012. An enthalpy formulation for glaciers and ice sheets. *J. Glaciol.* **58**(209), 441-457.
- [2] Paterson, W.S.B. 1994. *The physics of glaciers*, 3rd edn. Elsevier, Oxford
- [3] Arthern, R.J., Vaughan, D.G., Rankin, A.M., Mulvaney, R., and Thomas, E.R. 2010. In situ measurements of Antarctic snow compaction compared with predictions of models. *J. Geophys. Res.*, **115**, F03011, doi:10.1029/2009JF001306.
- [4] Reeh, N. 2008. A nonsteady-state firn-densification model for the percolation zone of a glacier. *J. Geophys. Res.*, **113**, F03023, doi:10.1029/2007JF000746.
- [5] Zwally, H., and J. Li. 2002. Seasonal and interannual variations of firn densification and ice-sheet surface elevation at the Greenland summit. *J. Glaciol.* **48**(161), 199-207.
- [6] Ligtenberg, S.R.M., Helsen, M.M., and van den Broeke, M.R. 2011. An improved semi-empirical model for the densification of Antarctic firn. *The Cryosphere*. **5**, 809-819. doi:10.5194/tc-5-809-2011.
- [7] Coleou, C. and Lesaffre, B. 1998. Irreducible water saturation in snow: experimental results in a cold laboratory. *Ann. Glaciol.*, **26**, 64-68.
- [8] Herron, M. and Langway, C. 1980. Firn densification: an empirical model. *J. Glaciol.* **25**(93), 373-385.
- [9] Waldner, P.A., Schneebeli, M., Schultze - Zimmermann, U., and Flühler, H. 2002. Effect of snow structure on water flow and solute transport. *Hydrol. Process.*, DOI: 10.1002/hyp.1401