Flow Model

**Defining a model glacier**

FlowModel uses a “Glacier” class to construct a model glacier system with some user-defined initial conditions. These include the basic dimensions of the glacier and its weathered crust along with some environmental variables at t=0. Specifically, the user provides the values defined in Table 1. These values are then used by the class constructor to provide eleven attributes to a “glacier” object. These attributes and their derivation are shown in Table 2. Some further explanation may be necessary for the structure of kx, ky and kz and the derivation of H0. The hydraulic conductivity variables kx, ky and kz are each three dimensional arrays containing values for the hydraulic conductivity of each individual grid cell in one specific dimension. We assume that the hydraulic conductivity is bidirectional in that only one value is required to describe the hydrauic conductivity for water flowing in either direction along a particular axis. Therefore, for a given cell there is a value for kx that controls the rate of flow between the cell faces oriented in the latitudinal plane, a value for ky that controls the rate of flow between the cell faces oriented in the longitudinal plane and a value for kz that controls the rate of flow between the top and bottom cell faces. The arrays kx, ky and kz are three dimensional when there is more than one vertical layer considered in the simulation. For H0, the hydraulic head in each dimension in each cell is calculated from the water table elevation. Since we assume the weathering crust to be an unconfined aquifer, we equate water table height with hydraulic head. However, since we are considering multiple vertical layers, we calculate the hydraulic head at multiple points beneath the water table surface. This is achieved by summing the pressure head (Ψ) and the elevation above sea level in each cell, where the pressure head in defined as the elevation difference between the measurement point and the water table surface.

|  |  |  |  |
| --- | --- | --- | --- |
| Symbol | Definition | Derivation | Unit |
| x | N cells in lateral dimension | user-defined | m |
| y | N cells in longitudinal dimension | user-defined | m |
| z | N cells in vertical dimension | user-defined | m |
| Δxy | length of grid cells in xy direction | user-defined | m |
| Δz | length of grid cells in z direction | user-defined | m |
| E | elevation at the lateral midpoint of the glacier. | user-defined | m a.s.l |
| TWC | Thickness of weathering crust (vertical distance between glacier surface and impenetrable ice layer) | user-defined | m |
| TWAT | Water table thickness (proportion of TWC saturated at t=0) | user-defined | % |
| Ccryo | Proportion of glacier surface covered by cryoconite holes at t=0 | user-defined | % |
| melt | melt rate | user-defined | m3/d |
| rain | rainfall | user-defined | m3/d |
| s | surface slope | user-defined | % |
| kxy | hydraulic conductivity between cells in the xy dimension | user-defined | m/d |
| kz | hydraulic conductivity between cells in the z dimension | user-defined | m/d |
| Fedge | extraction rate for water at the glacier edges | user-defined | m3/d |
| Fterm | extraction rate for water at glacier terminus | user-defined | m3/d |
| Locstream | array of cell indexes where supraglacial stream is located | user-defined | n/a |
| Locmoulin | array of cell indexes where moulin is located | user-defined | n/a |
| Fmoulin | extraction rate for water entering moulin | user-defined | m3/d |
| WCconstrain | Boolean to toggle confining the values for the hydraulic head to the upper and lower boundaries of the weathering crust. Hydraulic head above the upper glacier surface is interpreted as surface flow and instantaneously lost. | user-defined | n/a |
| Ss | specific hydrologic storage within each cell | user-defined | % |

*Table 1: User-defined values passed to the Glacier class constructor and used to set the initial conditions for the hydrological model.*

|  |  |  |  |
| --- | --- | --- | --- |
| Symbol | Definition | Derivation | Unit |
| SHP | three-tuple describing the length of the glacier in the z,y,x dimensions | SHP = (lengthx, lengthy, lengthz) | m |
| kx | three dimensional array of hydraulic conductivities between cell faces in the x-dimension | kx = SHP · kxy | m/d |
| ky | three dimensional array of hydraulic conductivities between cell faces in the y-dimension | ky = SHP · kxy | m/d |
| kz | three dimensional array of hydraulic conductivities between cell faces in the z-dimension | kz = SHP · kz | m/d |
| Bound | Boolean array where 1’s represent active cells where hydrologic flow is calculated, 0’s represent inactive cells that provide a model boundary | Hard coded. Interior cells set to 1, 1 cell perimeter set to 0. | n/a |
| Fq | three dimensional array containing the total extraction rate in each cell | Sum of user defined F terms | m3/d |
| H0 | three dimensional array containing the initial hydraulic head in each cell | where h = hydraulic head, Ψ = pressure head (i.e. elevation difference between measurement point and water table, and z = elevation at the measurement point) | m |
| Surfupper | two dimensional array containing the elevation of the upper glacier surface in each cell | E + (idx \* slope) + random noise (10-1m) | m a.s.l |
| Surflower | two dimensional array containing the elevation of the bottom surface of the weathering crust, ie. the boundary with impermeable glacier ice | Surfupper - TWC | m a.s.l |
| Loccryo | Boolean array where 1’s represent holes occupied by cryoconite holes. | Random choice of cells with probability of selection = Ccryo | n/a |
| WaterTable | Elevation of water table surface in each cell |  | m a.s.l |

*Table 2: Attributes of the glacier object derived from the user defined values shown in Table 1.*

**TransientFlowModel**

Once constructed, the glacier object is passed to the TransientFlowModel, which uses a finite difference scheme to simulate the hydrologic flow at a set of timesteps defined by the user (total time and timestep, measured in days). The finite difference model divides the glacier into equally spaced cubes on a Cartesian grid. Properties assigned to each cell are assumed constant within the cells. Flows are computed across the cell faces aligned in the x, y and z directions while hydraulic head and vector components are calculated at the cell centres. The cells positions in the x, y and z dimensions are identified by their indices i, j, k. The model yields a system of linear equations for the water balance of each cell which are solved simultaneously. The number of equations is equal to the number of active cells in the model, where active cells are those where hydraulic head is computed and not a prescribed constant. The water balance at each cell centre is computed as:

Eq. 1

where

Eq 2

where Qinc represents flow into the cell from neighbours and additional source terms and Qextr represents all extraction terms. In this paper, source terms included water generated by ice melting in situ (M), water arriving from rainfall (R) and water flowing arriving from neighbouring cells due to differences in hydraulic head so that Qi is calculated as in Eq 3. In most scenarios we do not consider any extraction terms F.

Eq 3

However, in some special cases we do provide values of F to represent certain physical processes. There is a contant extraction rate at the glacier edges (default 20 m3/d) representing seepage of water off the glacier edges into the surrounding terrain and also at the glacier terminus representing flow of water into the proglacial environment (default 200 m3/d). The two other possible cases are streams and moulins. In our model we provide the option to place a moulin or stream on the glacier surface at user defined locations. The cells included within a moulin always have a hydraulic head equal to zero and an arbitrarily high extraction rate (default 20000 m3/d) indicating that water entering a moulin is instantaneously lost to the subglacial environment. In the case of a stream, the hydraulic head is allowed to vary according to the model calculations but the hydraulic conductivity can be ajusted to arbitrarily large values (default 2000 m3/d) representing low impedence to flow and rapid removal of water via streamflow.

In the finite difference approximation the integral over time in Eq 1 is replaced by the mean flow during the considered time period *dt,* yielding Eq 2.

Eq 4

where V is the volume of the cell calculated as the product of Δx, Δy, Δz. is the net flow into cell i, as defined in Eq 3, with the inflow from neighbours (Qi) calculated using the cell conductances (C) as in Eq. 5.

Eq 5

where j represents the neighbouring cells in the x, y and z directions, and:

where

, , Eq 6

where

, , Eq 7

N is the total number of cells, ϕ is hydraulic head, Qi is the inflow to the cell with positive values representing net inflow and negative values representing net outflow, N is the total number of cells in the glacier, Ci,j is the conductance defined as the change in hydrologic flow across a cell caused by a unit change in hydraulic head (h) and calculated as in Eq2. Ci,j is calculated separately for each dimension x, y, and z. Equation 3 shows the derivation of R from the hydraulic conductivity k and the cell dimensions. Physically, Eq 7 represents the resistance of the cell to hydrologic flow in a direction perpendicular to the cell faces, calculated between the cell centre and the cell faces (hence the scalar 0.5). The conductance is the reciprocal of the flow resistance applied between adjacent cells in each direction.

It is clear from Eq 1 - 5 that the hydraulic head distribution, and therefore flow, between cells in the finite-difference grid is calculated at a series of discrete timesteps from the initial head distribution, hydraulic parameters and external stresses. The hydraulic head at the beginning of the first time step (t=0) is a boundary condition established from the user-defined values describing the glacier topography and initial weathering crust state. The model progresses by calculating the values of the hydraulic heads at the end of the first time step (t=1). The model equations then change because the knowns are updated using the head values at t=1 as the initial values and head at t=2 as the new unknowns. Simultaneously solving the system of equations governing the spatial distribution of hydraulic head iteratively in this way for each time step leads to a time series of hydraulic head distribution from which rates and directions of hydraulic flow can be inferred. At each time step the finite difference equations change and a new set of linear equation exist to be solved simultaneously across the grid. The hydraulic head at the beginning of the time step are knowns in the equation and the hydraulic head at the end of the timestep are unknowns to be calculated. The calculated heads become knowns in the equations to be solved for the next timestep. This is an example of backwards-difference because the time differential os approximated with knowledge of the values of hydraulic head at the end of the timestep, and it was chosen to avoid instabilities and error-propagation associated with central-difference and forward difference approximations as explained in the MODFLOW documentation.

With this finite difference scheme in place we can combine Eqs 4 and 5 and rewrite in finite-difference form as Eq 6:

Eq. 6

This finite-difference balance equation indicates that the sum of all flows across the grid should equal zero over the timestep, representing adherence to conservation of mass. Equation 6 can then be used as the basis for approximating the partial differential equation for groundwater flow. The coefficients (melt, rain ,F, Ss, V and C) are all knowns defined by the user and the hydraulic head at t = t-δt are known either from the initial conditions or from the calculated heads at the previous timestep. While Eq 6 cannot be solved directly because it represents a single equation with multiple unknown head values to solve, writing the equation for each individual active cell in the finite-difference grid yields N equations with N unknowns where each equation has only one unknown value to solve – the head at t=t.

We can simplify this equation by first realising that Equation 5 shows the first term to be equal to Qi and the source and extraction coefficients are knowns that can be grouped together as Qext - positive when entering the cell and negative when leaving. Therefore we rewrite as Eq 7.

Eq 7

We know the value of ht because it is the hydraulic head at the end of the previous timestep, or defined as an initial condition. We do not know the hydrauic head at t+δt. We can therefore rearrange Eq 7 to move all the unknown terms to the left side of the equation and the known terms to the right side of the equation by separating the known and unknown hydraulic heads, producing Equation 8.

Eq 8

Equation 8 can then be solved using a matrix formulation where there is one individual equation of the form of Eq 8 for each individual variable or fixed hydraulic head in the finite difference grid. We obtain Eq 9 by expressing Eq 8 in matrix notation.

Ah = b

where the matrix A contains the coefficients of hydraulic head (i.e. the left side of Eq 8) for all cells in the finite difference grid, h is a vector of hydraulic head values for each cell at the end of the timestep, and b is the right hand side of equation 8. A is referred to as the system matrix, it is square and has shape (Nx\*Ny\*Nz),(Nx\*Ny\*Nz), representing the conductance relation between each cell and **all** other cells. Of course, most of these values will be zeros – the only non-zero elements are the cell’s neighbours. Therefore, the majority of the memory allocation for the matrix A can be saved by using a sparse array. The square matrix contains the conductance relation for each cell with all other cells, meaning there is a row for each cell and a column for each cell, hence the array shape being (Nx\*Ny\*Nz),(Nx\*Ny\*Nz). This means the matrix diagonal is empty, because there is no conductance relation for a cell with itself. However, since there is an element on the matrix diagonal for each cell that is only related to that specific cell, this is where we want to place the total conductance, calculated as the sum of conductance in each direction. We therefore take the row sum and assign it to the matrix diagonal. This is the equivalent of placing a single conductance value into our original finite-difference grid. We also want to add the storage term (second term in Eq 8) to the conductance in each cell, so we can simply add it to the matrix diagonal. We can then compute the right-side of Eq.8 by multiplying matrix A with vector h, which is composed of the hydraulic head ϕ for each cell.

Since there is one element of ϕ for each individual cell in the model grid, the number of elements in ϕ when it is converted into a vector is equal to the number of rows in A. This means we can apply matrix-vector multiplication. The product of matrix A and vector ϕ yields the right side of Eq 8. With known RHS for the previous timestep and known system matrix A, we can now solve the system of linear equations for vector ϕ at the next time step. To do this for our sparse matrix, we use the Scipy tool spsolve from the library scipy.sparse.linalg. This yields values for the unknown heads at the next timstep, which become knowns to generate RHS values for the next time step, which is used to solve the matrix computation again, and this process is repeated until t = tn.

**Albedo-melt-porosity feedback**

At each timestep we have computed flows between adjacent cells in a finite difference grid composed of cubes of known volume. Each cell also has a known porosity, which can be related to specific surface area and density using a linear equation defined by Dadic et al. (2013). Therefore, each cell has a SSA that can be used as an inut variable to a two-stream radiative transfer model. Given meteorological data, we can establish a transient ice-physical model that interfaces with both the flow model and the radiative transfer model.

We know the volume of an individual cell – it is dx\*dy\*dz. We also know the porosity. The volume of each cell occupied by ice is equal to cell volume \* porosity, and the volume of each cell occupied by air (or water if saturated) is equal to cell volume \* 1-porosity. Applying Dadic et al.’s equation enables us to calculate specific surface area and density from the porosity. These can be used to set the initial conditions for a radiative transfer model, given some additional information regarding the illumination conditions. This will provide the albedo of each cell. The albedo can then be used to drive an energy balance model, given prescribed values for several meteorological variables and elevation data as already prescribed. Incoming irradiance can be obtained by running an atmospheric radiative transfer model (SBDART) for a given location and time. The albedo is grabbed from a lookup table. The user can define the proportional coverage of five algal concentrations between 0 – 100000 ppb. The program randomly distributes the appropriate proportions of each algal concentration across the model grid then uses the algal concentration to select a separate LUT where the albedo per unit grain size and density has been calculated with the specific algal concentration in the upper ice layer. By grabbing albedo values from the right LUT at the indexs corresponding to the calculated grain size and density, a 2d array of broadband albedo values is generated which can then be fed to the energy balance model.

The energy balance model then returns the melt in each cell in mm water equivalent per unit time. This is partitioned into a turbulent component and a radiative component. The radiative component causes erosion of ice grains beneath the ice surface. Therefore, the melt generated radiatively gives a value in mm w.e. that can be subtracted from the proportion of the cell occupied by ice, increasing the porosity of the cell. This in turn feds back to the SSA and albedo in the next timestep. However, water accumution also feeds into the SSA and causes the volume occupied by water to increase, reducing the SSA and decreasing the albedo in the next timestep.

The melt generated by turbulent fluxes causes surface lowering – this can be interpreted as a decrease in porosity because the most porous ice at the surface is transformed into liquid water to percolate downwards into the ice column, increasing the SSA. This can be parameterised as a decrease in porosity proportional to the ratio of the volume of melt water and the total volume of the cell. Again, this translates into an increase in SSA and a decrease in albedo at the next timestep. Therefore, the porosity is updated for the next timestep, influencing the subsurface flow regime by feeding into the calculation of the specific storage Ss.

Dadic relation:

SSA = 17.6 – 0.02p

where p = density. Porosity can be calculated as the ratio between the bulk density and the density of pure, unbroken ice (917 kg m-3). i.e. for a porosity of 0.5, bulk density = 917\*0.5 = 458.5 kgm-3. We hold bulk density constant at 650 kg m-3 and allow porosity and SSA to vary.

Since we are considering an unconfined aquifer, the relationship between porosity and storage coefficient is simple:

S= n-Sr

where S is the storage coefficient, n = porosity and Sr is the specific retention. Sr is a measure of the proportion of water retained in pore spaces due to capillarity after an aquifer drains. This is higher for smaller pores. In this model it is a tunabel parameter, we set it to 0.05, which approximates the specific retention of fine gravel (Bear et al. 1973). Therefore, the storage is slightly less than the porosity.

where Ss is storage coefficient, γw is the specific weight of water (9.805 kN/m3 at 0°C), n is the porosity of the medium, βp is the compressibility of the aquifer skeleton (set to 1 x 10-9) and βw is the compressibility of water (set to 4.6 x 10-10 m2/N) .

We also have known values for elevation and can prescribe values for meteorological variables. Therefore, we can use known porosity to calculate the surface albedo, use this to drive an energy balance model that returns radiative and turbulent energy balance driven melt in mm water equivelant per timestep. This can easily be converted into a volume of ice lost, which can be used to update the cell porosity in the next time step. The volume of water added is treated as a source term that adjusts values of FQ at the next timestep. Therefore, the equations in the matrix operation Ax=b are adjusted in each timestep to account for this albedo-melt-porosity feedback.

flow within a timestep varies according to variations in hydraulic head. Therefore the mean hydraulic head is required to integrate hydraulic flow over the timestep by multipling by δt, but it is not clear precisely where in the timestep the hydraulic head is at its mean value.

There is a design decision to make regarding the precise point within each timestep that is considered to be the optimum value for representing flow across that timestep, since it is unlikely that in a real system the mean flow accurately represents the true flow at any instant between the beginning and end of the time step. Therefore, a parameter ε is defined with values between 0-1. If ε=0 the flow at the start of the timestep is taken to be the flow across the timestep, if ε= 1 the flow at the end of the timestep is taken to be the flow across the timestep, and 0 < ε < 1 represents a continuous range between those two extremes. We assume the change in flow between the beginning and end of the timestep is linear.

**VectorComponents**

**ParticleTracker**

**TESTS:**

**Unit Tests:**

**TransientFlowModel:**

alter grid dimensions

alter time steps

alter slope

alter aspect

add moulin

add stream

test range of extractions

test adding rainfall

test constraining head to WC

test effects of varying flow at terminus

**MELT CALCULATIONS**

test changing aspect

test changing inswrad

test algal concentration

test