The Glacier Energy and Mass Balance model (GEMB)

Table of Contents

Model Documentation

Created by: Alex S. Gardner, University of Alberta

Date last modified: Dec 2009

Description:

This program calculates a detailed 1-D surface glacier mass balance and includes detailed representation of subsurface process. Key features:

- melt water percolation and refreeze
- pore water retention
- dynamic albedo with long-term memory
- subsurface temperature diffusion
- subsurface penetration of shortwave radiation

Reference:

Many aspects of the general model structure and specific parameterizations are taken from:

Bassford, R. P., 2002: Geophysical and numerical modelling investigations of the ice caps on Severnaya Zemlya, Bristol Glaciology Centre, School of Geographical Sciences, University of Bristol 220.

Bougamont, M. and J. L. Bamber, 2005: A surface mass balance model for the Greenland Ice Sheet. Journal of Geophysical Research-Earth Surface, 110, doi:10.1029/2003JD004451.

Greuell, W.; Konzelmann, T., 1994. Numerical Modeling of the Energy-Balance and the Englacial Temperature of the Greenland Ice-Sheet - Calculations for the Eth-Camp Location (west Greenland, 1155M

Asl). Global and Planetary Change. 9(1994)

Model variables:

- a = albedo [fraction] % dz = grid cell size [m] % EC = evaporation (-) & condensation (+)
- m = mass [kg m^-^3]
- M = melt mm w.e. [kg m^-^2]
- z = grid cell depth below surface [m]

Run initialization files

```
%clear all                    % clear workspace
%close all                    % close all open fligures
tic                           % start timer
run modelSettings             % load initial model paramaterizations
```

Load in meteorological data

load meteorological data derived from an automatic weather station (AWS):

- dateN : date/time [UTC]
- Ta: 2m air temperature [°C]
- V: wind speed [m s^-^1]
- dswrf: downward shortwave radiation flux [W m^-^2]
- dlwrf: downward longwave radiation flux [W m^-^2]
- RH: relative humidity [%]
- P: precipitation [mm w.e. m^-^2]
- eAir: screen level vapor pressure [Pa]
- dt: time step of met data [s]
- elev: surface elevation [m a.s.l.]

```
[dateN, Ta0, V0, dswrf0, dlwrf0, RH0, P0, eAir0, dt, elev] = ...
    loadMetData(S.cldFrac, S.site, S.dataFreq, S.saveFreq);
```

Generate model grid

```
dz = gridInitialize(S.zTop, S.dzTop, S.zMax, S.zY);
```

Warning: initial top grid cell length (dzTop) is < 0.05 m

Initialize model variables

```
% --------------- INPORTANT NOTE ABOUT GRAIN PROPERTIES -------------------
% initial grain properties must be chosen carefully since snow with a
% density that exceeds 400 kg m-3 will no longer undergo metamorphosis and
% therefore if grain properties are set inappropriately they will be
% carried all through the model run.
%
% !!!! grainGrowth model needs to be fixed to allow evolution of snow !!!!!
% !!!!  grains for densities > 400 kg m-3                              !!!!!
% -------------------------------------------------------------------------

% initialize profile variables
m = length(dz);
Z = zeros(m,1);         % create zeros matrix
d = Z + 910;            % density to that of ice [kg m-3]
re = Z + 2.5;           % set grain size to old snow [mm]
gdn = Z;                % set grain dentricity to old snow
gsp = Z;                % set grain sphericity to old snow
M = 0;                  % melt [kg m-2]
EC = 0;                 % surface evaporation (-) condensation (+) [kg m-2]
W = Z;                  % set water content to zero [kg m-2]
a = Z + S.aSnow;        % set albedo equal to fresh snow [fraction]

% set initial grid cell temperature to the annual mean temperature [K]
T = Z + S.meanT;

% fixed lower temperatuer bounday condition - T is fixed
T_bottom = T(end);

% initialize output structure

% single level time series
n = length(dateN);
I = S.saveFreq:S.saveFreq:n;            % save index
Z = zeros(size(I));                     % create zeros matrix
O.date = dateN(I)';
O.Ta = (Ta0(I) - 273.15)';
O.P = P0(I)';

O.M = Z; O.elev = Z; O.R = Z; O.netSW = Z; O.netLW = Z; O.shf = Z;
O.lhf = Z; O.a1 = Z; O.lwUp = Z; O.netQ = Z; O.re1 = Z; O.d1 = Z;

% multi level time series
X = 500;                            % number of addtional vertical levels
Z = zeros(length(d)+X,length(I));   % create zeros matrix

O.d = Z; O.T = Z; O.W = Z; O.a = Z; O.dz = Z; O.depth = Z; O.re = Z;
O.gdn = Z; O.gsp = Z;

% initialize output counter
out = 0;
record = 1;
```

Start year loop for model spin up

```
for year = 1:S.spinUp + 1
```

```
% determine initial mass [kg]
initMass = sum (dz .* d) + sum(W);

% initialize cumulative variables
sumR = 0; sumM = 0; sumEC = 0; sumP = 0; sumMassAdd = 0; sumW = 0;
```

Start loop for data frequency

```
% specify the date range over which the mass balance is to be calculated
for date = 1:length(dateN)
    % extract daily data
    dlwrf = dlwrf0(date);   % downward longwave radiation flux [W m-2]
    dswf = dswrf0(date);    % downward shortwave radiation flux [W m-2]
    Ta = Ta0(date);         % screen level air temperature [K]
    P = P0(date);           % precipitation [mm w.e. m-2] == [kg m-2]
    V = V0(date);           % wind speed [m s-1]
    eAir = eAir0 (date);    % screen level vapor pressure [Pa]

    % albedo calculations contained in switch to minimize passing of
    % variables to albedo function
    switch S.aIdx
        case {1,2}
            % snow grain metamorphism
            [re, gdn, gsp]  = ...
                grainGrowth(T, dz, d, W, re, gdn, gsp, dt);

            % calculate snow, firn and ice albedo
            a(1) = albedo(S.aIdx, re(1), [], [], [], [], [], [], ...
                [], [], [], [], [], [], []);

        case 3

            % calculate snow, firn and ice albedo
            a(1) = albedo(S.aIdx, re(1), d(1), S.cldFrac, S.aIce, ...
                S.aSnow, [], [], [], [], [], [], [], [], []);

        case 4
            % calculate snow, firn and ice albedo
            a = albedo(S.aIdx, [], [], [], S.aIce, S.aSnow, a, T, ...
                W, P, EC, S.t0wet, S.t0dry, S.K, dt);
    end

    % determine distribution of absorbed sw radiation with depth
    swf = shortwave(S.swIdx, S.aIdx, dswf, a(1), d, dz, re);

    % calculate new temperature-depth  profile
    [T EC] =  thermo(T, dz, d, swf, dlwrf, Ta, V, eAir, W(1), dt, ...
        elev, S.Vz, S.Tz);

    % change in thickness of top cell due to evaporation/condensation
    % assuming same density as top cell
    % NEED TO FIX THIS INCASE ALL OR MORE OF CELL EVAPORATES
    dz(1) = dz(1) + EC / d(1);

    % add snow/rain to top grid cell adjusting cell depth, temperature
    % and density

    [T, dz, d, W, a, re, gdn, gsp] = accumulation(Ta, T, dz, d, ...
         P, W, S.dzMin, a, S.aSnow, re, gdn, gsp);

    % calculate water production, M [kg m-2] resulting from snow/ice
    % temperature exceeding 273.15 deg K (> 0 deg C), runoff R [kg m-2]
```

```
% and resulting changes in density

 [M R d T dz W mAdd a re gdn gsp] = melt(T, d, dz, W, a, ...
     S.dzMin, S.zMax, S.zMin, re, gdn, gsp);

% allow non-melt densification
[d, dz] = densification(d, T, dz, S.meanSnow, dt);

% calculate upward longwave radiation flux [W m-2]
% not used in energy balance
% CALCULATED FOR EVERY SUB-TIME STEP IN THERMO EQUATIONS
ulwrf = 5.67E-8 * T(1)^4;

% calculate net shortwave and longwave [W m-2]
netSW = sum(swf);
netLW = dlwrf - ulwrf;

% calculate turbulent heat fluxes [W m-2]
[shf lhf dayEC] = turbulentFlux(Ta, T(1), V, eAir, d(1), W(1), ...
    elev, S.Vz, S.Tz);

% sum component mass changes [kg m-2]
sumMassAdd = mAdd + sumMassAdd;
sumM = M + sumM;
sumR = R + sumR;
sumW = sum(W);
sumP = P +  sumP;
sumEC = sumEC + EC;           % evap (-)/cond(+)

% calculate total system mass
sumMass = sum(dz .* d);
dMass = sumMass + sumR + sumW - sumP - sumEC - initMass - ...
    sumMassAdd;
dMass = round(dMass * 100)/100;

% check mass conservation
if dMass ~= 0
    error('total system mass not conserved in MB function')
end

% check bottom grid cell T is unchanged
if T(end)~=T_bottom
    warning('T(end) ~= T_bottom')
end

if year == S.spinUp + 1
    if rem(date, S.saveFreq) == 0
```

Store model output in strucutre format

```
        r = date/S.saveFreq;
        O.M(r) = M*(86400/dt);       O.R(r) = R*(86400/dt);
        O.netSW(r) = netSW;          O.netLW(r) = netLW;
        O.shf(r) = shf;              O.lhf(r) = lhf;
        O.a1(r) = a(1);              O.re1(r) = re(1);
        O.lwUp(r) = ulwrf;           O.d1(r) = d(1);
        O.netQ(r) = netSW + ...
            netLW + shf + lhf;

        o = (size(d,1) - 1);

        O.re(end-o:end,r) = re;      O.d(end-o:end,r) = d;
        O.T(end-o:end,r) = T;        O.W(end-o:end,r) = W;
```

```
            O.a(end-o:end,r) = a;       O.dz(end-o:end,r) = dz;
            O.gdn(end-o:end,r) = gdn;   O.gsp(end-o:end,r) = gsp;
            O.depth(end-o:end,r) = ...
                flipud(cumsum(flipud(dz)));

            % will need to correct elevation for mass added/subtracted from base
            O.elev(r) = sum(dz);


            end
        end
    end
    % display year completed and time to screen
    disp(['year: ' num2str(year) ' time: ' num2str(toc) 's' ' melt: ' num2str(sumM
```

year: 1 time: 14.6902s melt: 252.1235mm

year: 2 time: 28.4277s melt: 438.2385mm

year: 3 time: 42.8283s melt: 448.5413mm

year: 4 time: 59.1105s melt: 449.7817mm

year: 5 time: 78.7161s melt: 453.5097mm

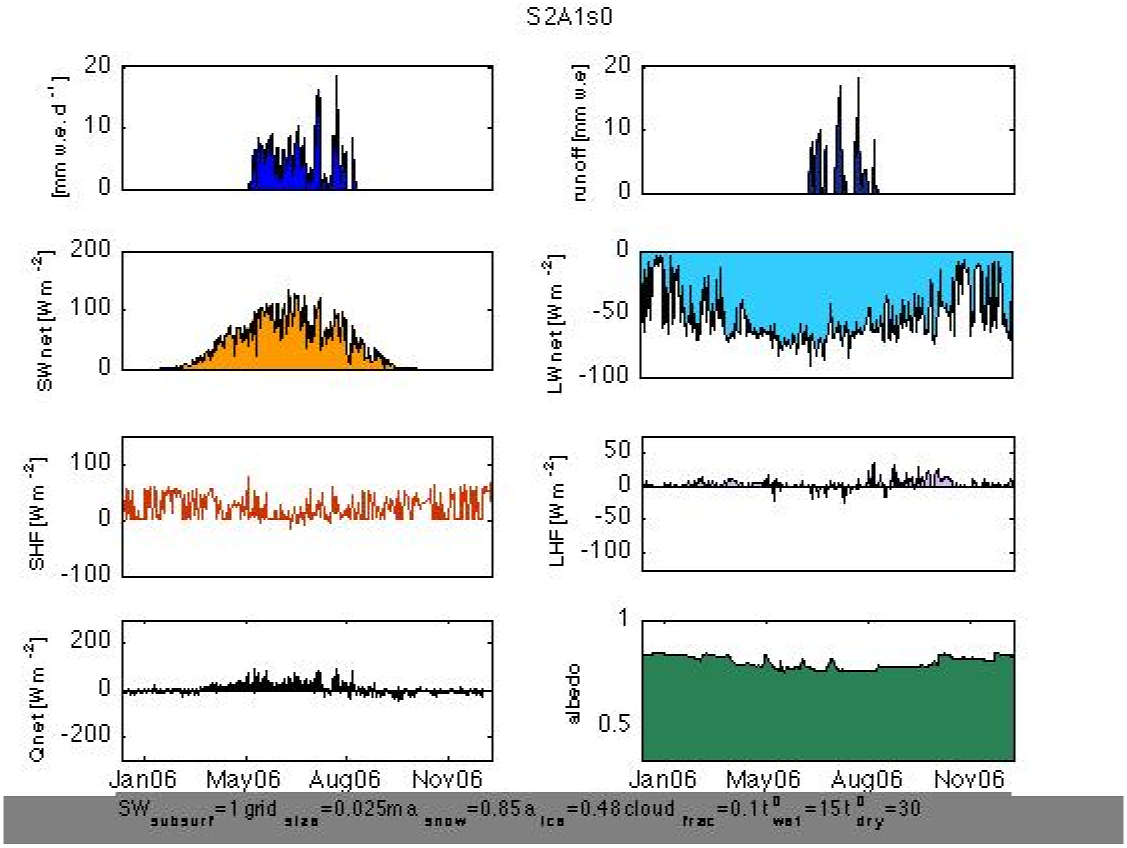
```
end
```

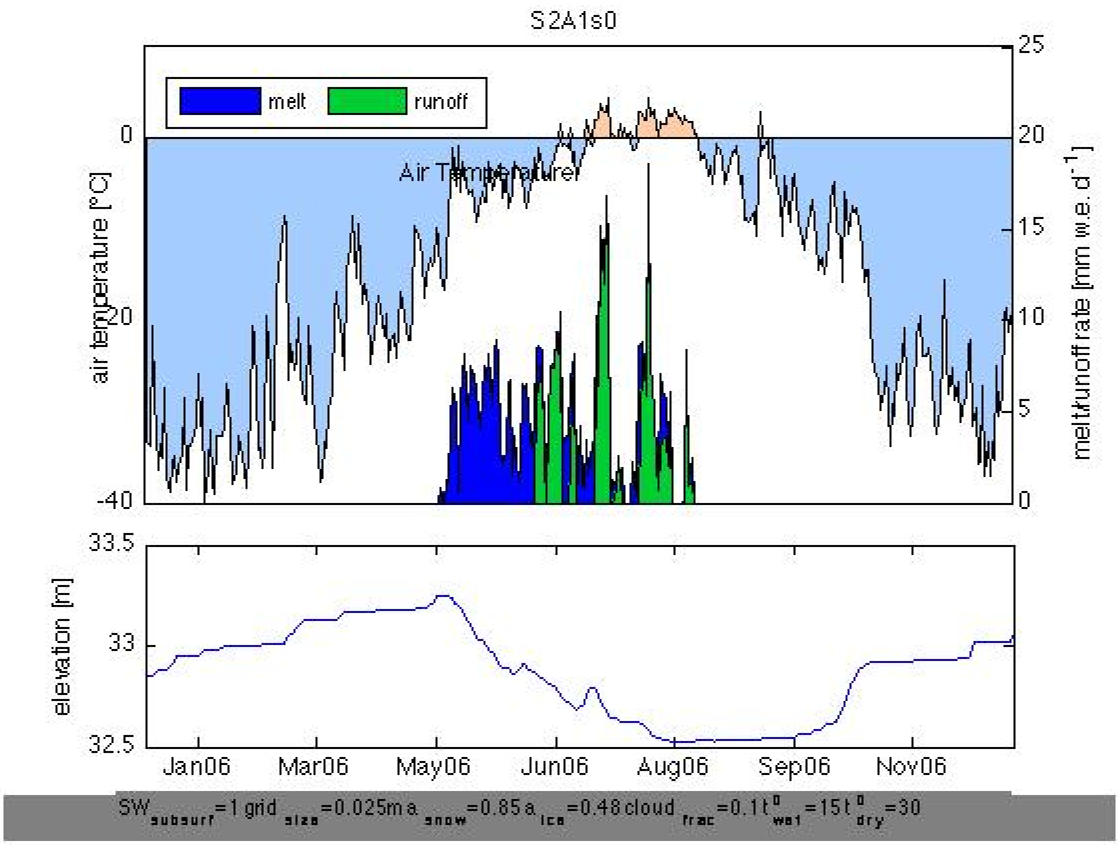
Save model output and model run settings

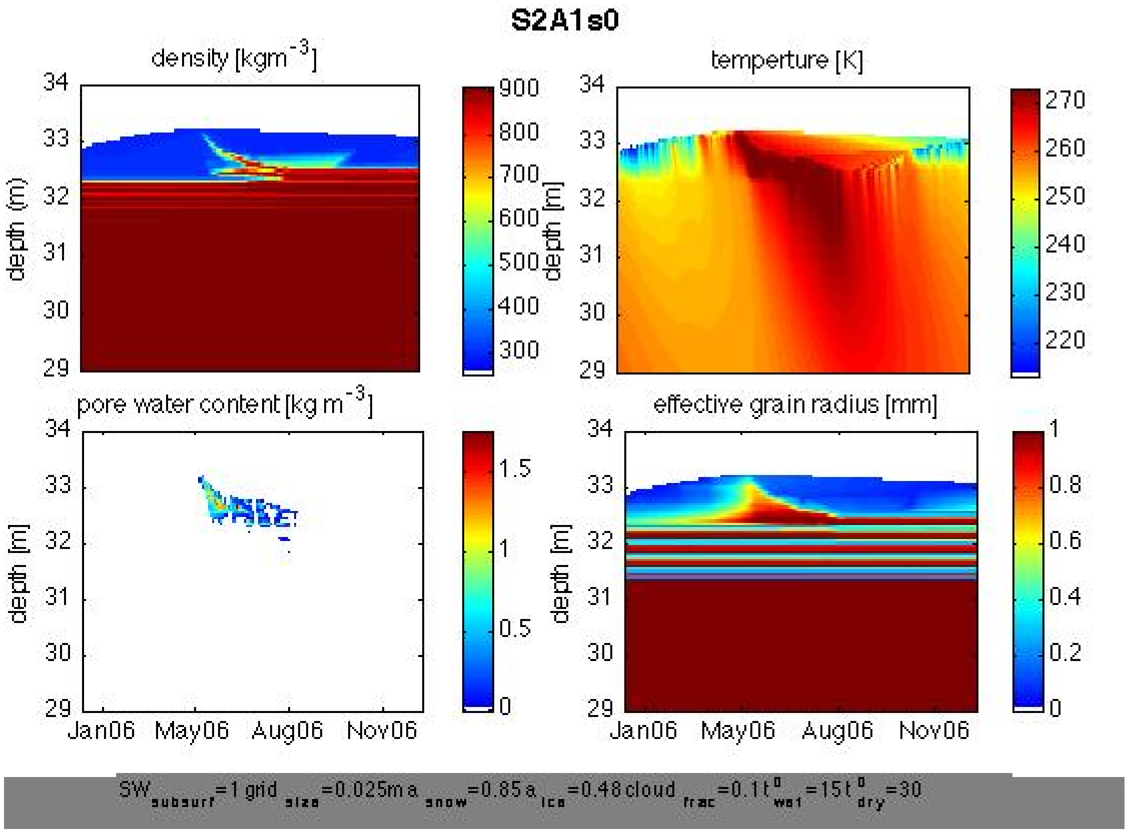
```
save(fullfile('..','Output',runID), 'O', 'S')
```

Plot model output

```
addpath(fullfile('..','Figures'))
plotOutput(runID)
```


S2A1s0
[mm w.e. d -1]
runoff [mm w.e]
SWnet [W m -2]
LWnet [W m -2]
SHF [W m -2]
LHF [W m -2]
Qnet [W m -2]
albedo
Jan06 May06 Aug06 Nov06
SW subsurf=1 grid size=0.025m a snow=0.85 a ice=0.48 cloud frac=0.1 t0 wet=15 t0 dry=30


S2A1s0
melt
runoff
Air Temperature
air temperature [°C]
melt/runoff rate [mm w.e. d-1]
elevation [m]
Jan06 Mar06 May06 Jun06 Aug06 Sep06 Nov06
SW subsurf=1 grid size=0.025m a snow=0.85 a ice=0.48 cloud frac=0.1 t0 wet=15 t0 dry=30



Published with MATLAB® 7.9