PISM's climate forcing components

The PISM Authors

Contents

Support by email: `help@pism-docs.org`.

Please see the *PISM User's Manual* for the full list of authors.

Manual date June 30, 2015. Based on PISM revision `stable v0.7.1-2-g79b8840`.

Get development branch source code: `git clone -b dev git@github.com:pism/pism.git pism-dev`

1 Introduction

PISM has a well-defined separation of climate forcing from ice dynamics. This manual is about the climate forcing interface.

By contrast, most options documented in the PISM User's Manual [1] control the ice dynamics part. Section 3.4 of the User's Manual does, however, give an overview of PISM's surface (atmosphere) and ocean (sub-shelf) interfaces. At these interfaces the surface mass and energy balances are determined and/or passed to the ice dynamics code.

To get started with climate forcing usage we need to introduce some language to describe parts of PISM. In this manual a *component* is a piece of PISM code, usually a C++ class. A combination of components (or, in some cases, one component) makes up a "model" — an implementation of a physical/mathematical description of a system.

PISM's climate forcing code has two kinds of components.

- Ones that can be used as "stand-alone" models, such as the implementation of the PDD scheme (section 5.4). These are *model components*.
- Ones implementing "corrections" of various kinds, such as lapse rate corrections (sections 5.7 and 6.10) or ice-core derived offsets (sections 6.6 and 7.5, for example). These are called *modifier components* or *modifiers*.

Model components and modifiers can be chained as shown in Figure 1. For example,

```
-ocean constant,delta_SL -ocean_delta_SL_file delta_SL.nc
```

combines the component providing constant (both in space and time) ocean boundary conditions with a modifier that applies scalar sea level ("SL") offsets. This combination one of the many ocean models that can be chosen using components as building blocks.

Section 3 gives examples of combining components to choose models. Before that we address how PISM handles model time (Section 2).

Summary of the main idea in using this manual:

> Setting up PISM's climate interface *requires* selecting one surface and one ocean component. The surface component may use an atmosphere component also; see Figure 1. Command-line options `-atmosphere`, `-surface` and `-ocean` each take a comma-separated list of keywords as an argument; the first keyword *has* to correspond to a model component, the rest can be "modifier" components. Any of these options can be omitted to use the default atmosphere, surface or ocean model components, but one has to explicitly choose a model component to use a modifier. Model components and modifiers are chained as in Figure 1.

---

[1] PDF for latest stable release at `http://www.pism-docs.org/wiki/lib/exe/fetch.php?media=manual.pdf`.

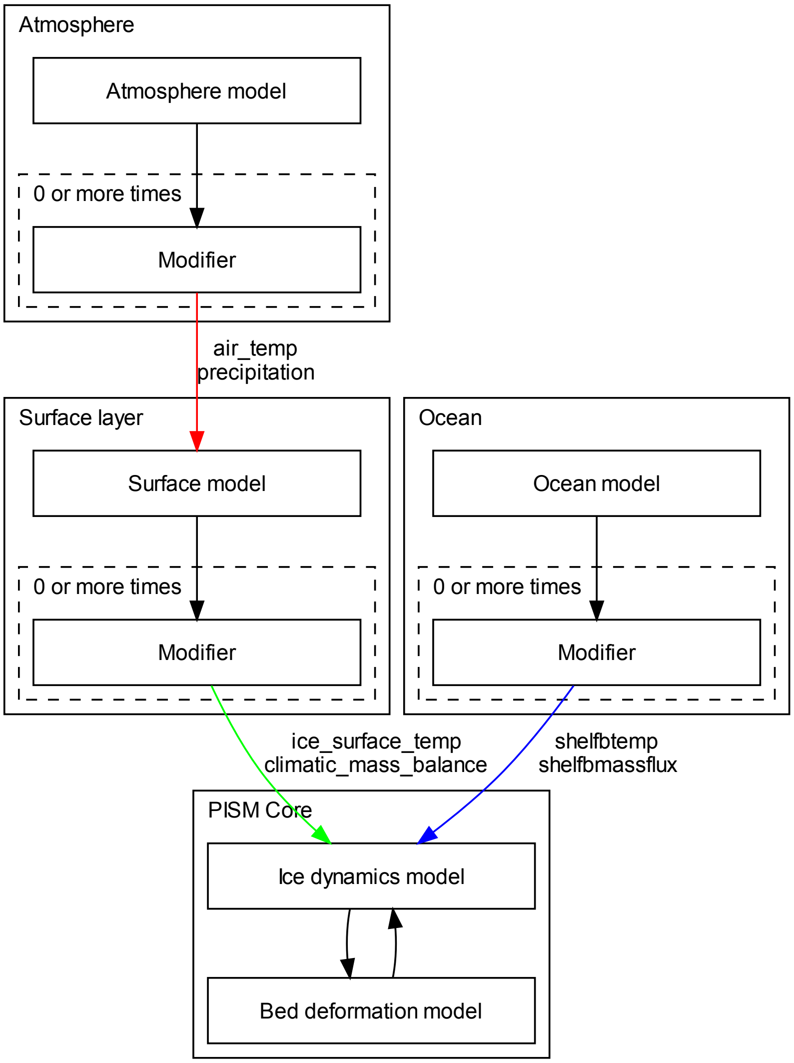


Figure 1: PISM climate input data flow. Colored arrows match colored interfaces in Figure 15 in the User's Manual. An atmosphere component is only needed for some surface models.

2 Managing model time

Most of PISM only needs to know how long the current time step is. The climate forcing (reporting) code, on the other hand, uses time in a precise manner to provide (and report) the correct values at the right time. For example: the February mass balance should be used for 28 days (except during leap years) and not $365/12 = 30.4167$ days.

2.1 Periodic climate data

All components reading time-dependent forcing data from files can interpret it as "periodic". The length of the period (in years) is specified using a `-..._period` option. For example, to prescribe a periodic climate which has the same values each year but which includes inter-annual variations, using the `-surface given` option, set:

```
-surface given -surface_given_period 1 -surface_given_file forcing.nc
```

Each component has a unique command-line option prefix for a `-..._period` option. Please refer to corresponding sections for allowed prefixes.

If forcing data has the period other than one year it is also necessary to specify the "starting time" using the `-..._reference_year` option.

For example, to use a 20 year long climate record as periodic climate starting at the beginning of the model year 10, do

```
-surface given -surface_given_period 20 -surface_given_file forcing.nc \
-surface_given_reference_year 10
```

Note that the reference year is given in *model years*, not calendar years.

The `time` variable in a forcing file that is to be used as periodic should start at 0. (In other words, time in a file with periodic forcing data is *time since the beginning of a period.*) Please see section 5.4 of the User's Manual for a discussion of time units appropriate in forcing files.

2.2 Using time bounds in forcing data

PISM interprets climate forcing data as piecewise-constant in time. A forcing file is required to contain time bounds corresponding to each record.

PISM follows the CF (Climate and Forecasting) meta-data conventions. The `ncdump -h` output from a conforming file would look similar to:

```
netcdf forcing {
dimensions:
        time = UNLIMITED ; // (214 currently)
        nv = 2 ;
variables:
        double time(time) ;
                time:units = "seconds since 2000-1-1" ;
```

```
                time:axis = "T" ;
                time:bounds = "time_bounds" ;
                time:calendar = "gregorian" ;
                time:long_name = "time" ;
        double nv(nv) ;
        double time_bounds(time, nv) ;
```

The `time_bounds` variable stores the starting and the ending time for each interval in the forcing. This variable is assumed to have the same units as the `time` variable it is associated with, which is why its arguments are not set in this example.

Please see the CF Conventions document for details.

3 Examples and corresponding options

This section gives a very brief overview of some coupling options. Please see sections referenced below for more information.

3.1 One way coupling to a climate model

One-way coupling of PISM to a climate model can be achieved by reading a NetCDF file with time- and space-dependent climate data produced by a climate model.

There are two cases:

- coupling to a climate model that includes surface (firn, snow) processes
- coupling to a climate model providing near-surface air temperature and precipitation

3.1.1 Reading ice surface temperature and mass balance

This is the simplest case. It is often the preferred case, for example when the climate model in use has high quality surface mass and energy sub-models which are then preferred to the highly simplified (e.g. temperature index) surface models in PISM.

Variable names:	`climatic_mass_balance`, `ice_surface_temp`
Options:	`-surface given -surface_given_file forcing.nc`
See also	5.2

3.1.2 Reading air temperature and precipitation

As mentioned above, if a climate model provides near-surface air temperature and precipitation, these data need to be converted into top-of-the-ice temperature and climatic mass balance.

One way to do that is by using a temperature index (PDD) model component included in PISM. This component has adjustable parameters; default values come from [12].

Variable names:	`precipitation`, `air_temp`
Options:	`-atmosphere given -atmosphere_given_file forcing.nc` `-surface pdd`
See also	`-atmosphere given`: 6.1, `-surface pdd`: 5.4

If melt is negligible `-surface pdd` should be replaced with `-surface simple` (see section 5.1).

3.2 Using climate anomalies

Prognostic modeling experiments frequently use time- and space-dependent air temperature and precipitation anomalies.

Variable names:	`precipitation_anomaly`, `air_temp_anomaly`
Options:	`-atmosphere given,anomaly -atmosphere_anomaly_file anomalies.nc` `-surface simple`
See also	`-atmosphere given`: 6.1, `anomaly`: 6.11, `-surface simple`: 5.1

The `simple` surface model component re-interprets precipitation as climatic mass balance, which is useful in cases when there is no melt (Antarctic simulations is an example).

Simulations of the Greenland ice sheet typically use `-surface pdd` instead of `-surface simple`.

3.3 SeaRISE-Greenland

The SeaRISE-Greenland setup uses a parameterized near-surface air temperature [3] and a constant-in-time precipitation field read from an input (`-i`) file. A temperature-index (PDD) scheme is used to compute the climatic mass balance.

Variable names:	`precipitation`, `lat`, `lon`
Options:	`-atmosphere searise_greenland -surface pdd`
See also	`-atmosphere searise_greenland`: 6.3, `-surface pdd`: 5.4

The air temperature parameterization is a function of latitude (`lat`), longitude (`lon`) and surface elevation (dynamically updated by PISM).

3.4 SeaRISE-Greenland paleo-climate run

The air temperature parameterization in the previous section is appropriate for present day modeling. PISM includes some mechanisms allowing for corrections taking into account differences between present and past climates. In particular, one can use ice-core derived scalar air temperature offsets [9], precipitation adjustments [7], and sea level offsets from SPECMAP [8].

Variable names:	`precipitation`, `delta_T`, `delta_SL`, `lat`, `lon`

Options:	`-atmosphere searise_greenland,delta_T`
	`-atmosphere_delta_T_file delta_T.nc -surface pdd`
	`-ocean constant,delta_SL -ocean_delta_SL_file delta_SL.nc`

See also	`-atmosphere searise_greenland`: 6.3, `-surface pdd`: 5.4 `-ocean constant`: 7.1 `delta_SL`: 7.5

Note that the temperature offsets are applied to *air* temperatures at the *atmosphere level*. This ensures that ΔT influences the PDD computation.

3.5 Antarctic paleo-climate runs

Variable names:	`climatic_mass_balance`, `air_temp`, `delta_T`, `delta_SL`
Options:	`-surface given,delta_T -surface_delta_T_file delta_T.nc` `-ocean constant,delta_SL -ocean_delta_SL_file delta_SL.nc`
See also	`-surface given`: 5.2, `delta_T`: 5.6, `-ocean constant`: 7.1, `delta_SL`: 7.5

4 Checking if forcing data is used correctly

It is very important to ensure that selected forcing options produce the result you expect: we find that the ice sheet response is very sensitive to provided climate forcing, especially in short-scale simulations.

This section describes how to use PISM to inspect climate forcing.

4.1 Visualizing climate inputs, without ice dynamics

Recall that internally in PISM there is a separation of climate inputs from ice dynamics (see User's Manual, section 3.4). This makes it possible to turn "off" the ice dynamics code to visualize the climate mass balance and temperature boundary conditions produced using a combination of options and input files. This is helpful during the process of creating PISM-readable data files, and modeling with such.

To do this, use the option `-test_climate_models` (which is equivalent to `-stress_balance none` and `-energy none`) together with PISM's reporting capabilities (`-extra_file`, `-extra_times`, `-extra_vars`).

Turning "off" ice dynamics saves computational time while allowing one to use the same options as in an actual modeling run. Note that `-test_climate_models` does *not* disable geometry updates, so one can check if surface elevation feedbacks modeled using lapse rates (and similar) work correctly. Please use the `-no_mass` command-line option to fix ice geometry. (This may be necessary if the mass balance rate data would result in extreme ice sheet growth that is not balanced by ice flow in this setup.

As an example, set up an ice sheet state file and check if climate data is read in correctly:

```
$ mpiexec -n 2 pisms -eisII A -y 1000 -o state.nc
$ pismr -i state.nc -surface given -extra_times 0.0:0.1:2.5 \
         -extra_file movie.nc -extra_vars climatic_mass_balance,ice_surface_temp \
         -ys 0 -ye 2.5
```

Using `pisms` merely generates demonstration climate data, using EISMINT II choices [11]. The next run extracts the surface mass balance `climatic_mass_balance` and surface temperature `ice_surface_temp` from `state.nc`. It then does nothing interesting, exactly because a constant climate is used. Viewing `movie.nc` we see these same values as from `state.nc`, in variables `climatic_mass_balance`, `ice_surface_temp`, reported back to us as the time- and space-dependent climate at times `ys:dt:ye`. It is a boring "movie."

A more interesting example uses a positive degree-day scheme (section 5.4). The positive degree-day scheme uses a variable called `precipitation`, and a calculation of melting, to get the surface mass balance `climatic_mass_balance`.

Assuming that `g20km_pre100.nc` was created as described in the User's Manual, section 2, running

```
$ pismr -test_climate_models -no_mass -i g20km_pre100.nc \
        -atmosphere searise_greenland -surface pdd \
        -ys 0 -ye 1 -extra_times 0:1week:1 \
        -extra_file foo.nc \
        -extra_vars climatic_mass_balance,ice_surface_temp,air_temp_snapshot,precipitation
```

produces `foo.nc`. Viewing in with `ncview` shows an annual cycle in the variable `air_temp` and a noticeable decrease in the surface mass balance during summer months (see variable `climatic_mass_balance`). Note that `ice_surface_temp` is constant in time: this is the temperature *at the ice surface but below firn* and it does not include seasonal variations [5].

4.2 Low-resolution test runs

Sometimes a run like the one above is still too costly. In this case it might be helpful to replace it with a similar run on a coarser grid, with or without the `-test_climate_models` option. (Testing climate inputs usually means checking if the timing of modeled events is right, and high spatial resolution is not essential.)

The command

```
$ pismr -i g20km_pre100.nc -bootstrap -Mx 51 -My 101 -Mz 11 \
        -atmosphere searise_greenland \
        -surface pdd -ys 0 -ye 2.5 \
        -extra_file foo.nc -extra_times 0:0.1:2.5 \
        -extra_vars climatic_mass_balance,air_temp_snapshot,smelt,srunoff,saccum
        -ts_file ts.nc -ts_times 0:0.1:2.5 \
        -o bar.nc
```

will produce `foo.nc` containing a "movie" very similar to the one created by the previous run, but including the full influence of ice dynamics.

In addition to `foo.nc`, the latter command will produce `ts.nc` containing scalar time-series. The variable `surface_ice_flux` (the *total over the ice-covered area* of the surface mass flux) can be used to detect if climate forcing is applied at the right time.

4.2.1 Visualizing the climate inputs in the Greenland case

Assuming that `g20km_pre100.nc` was produced by the run described in section 2), one can run the following to check if the PDD model in PISM (see section 5.4) is "reasonable":

```
$ pismr -i g20km_pre100.nc -atmosphere searise_greenland,paleo_precip \
          -surface pdd -atmosphere_paleo_precip_file pism_dT.nc \
```

```
          -extra_times 0:1week:3 -ys 0 -ye 3 \
          -extra_file pddmovie.nc -o_order zyx \
          -extra_vars climatic_mass_balance,air_temp_snapshot
```

This produces the file `pddmovie.nc` with several variables: `climatic_mass_balance` (instantaneous net accumulation (ablation) rate), `air_temp_snapshot` (instantaneous near-surface air temperature), `precipitation` (mean annual ice-equivalent precipitation rate) and some others.

The variable `precipitation` does not evolve over time because it is part of the SeaRISE-Greenland data and is read in from the input file.

The other two variables were used to create figure 2, which shows the time-series of the accumulation rate (top graph) and the air temperature (bottom graph) with the map view of the surface elevation on the left.

Here are two things to notice:

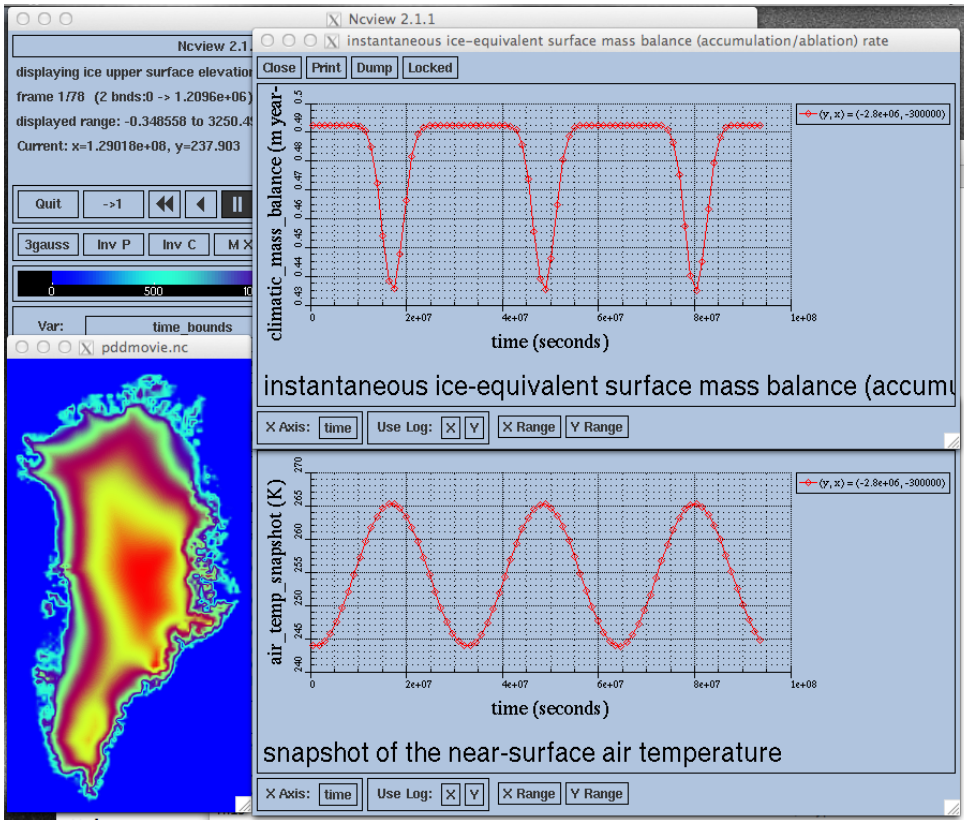
1. The summer peak day is in the right place. The default for this value is July 15 (day 196, at approximately $196/365 \simeq 0.54$ year). (If it is important, the peak day can be changed using the `snow_temp_july_day` configuration parameter).
2. Lows of the surface mass balance rate `climatic_mass_balance` correspond to positive degree-days in the given period, because of highs of the air temperature. Recall the air temperature graph does not show random daily variations. Even though it has the maximum of about 266 Kelvin, the parameterized instantaneous air temperature can be above freezing. A positive value for positive degree-days is expected [2].

We can also test the surface temperature forcing code with the following command.

```
$ pismr -i g20km_pre100.nc -surface simple \
          -atmosphere searise_greenland,delta_T \
          -atmosphere_delta_T_file pism_dT.nc \
          -extra_times 100 -ys -125e3 -ye 0 \
          -extra_vars ice_surface_temp \
          -extra_file dT_movie.nc -o_order zyx \
          -test_climate_models -no_mass
```

The output `dT_movie.nc` and `pism_dT.nc` were used to create figure 3.

This figure shows the GRIP temperature offsets and the time-series of the temperature at the ice surface at a point in southern Greenland (bottom graph), confirming that the temperature offsets are used correctly.



```
displaying ice upper surface elevatio
frame 1/78  (2 bnds:0 -> 1.2096e+06)
displayed range: -0.348558 to 3250.4
Current: x=1.29018e+08, y=237.903
```

Figure 2: Time series of the surface mass balance rate and near-surface air temperature.

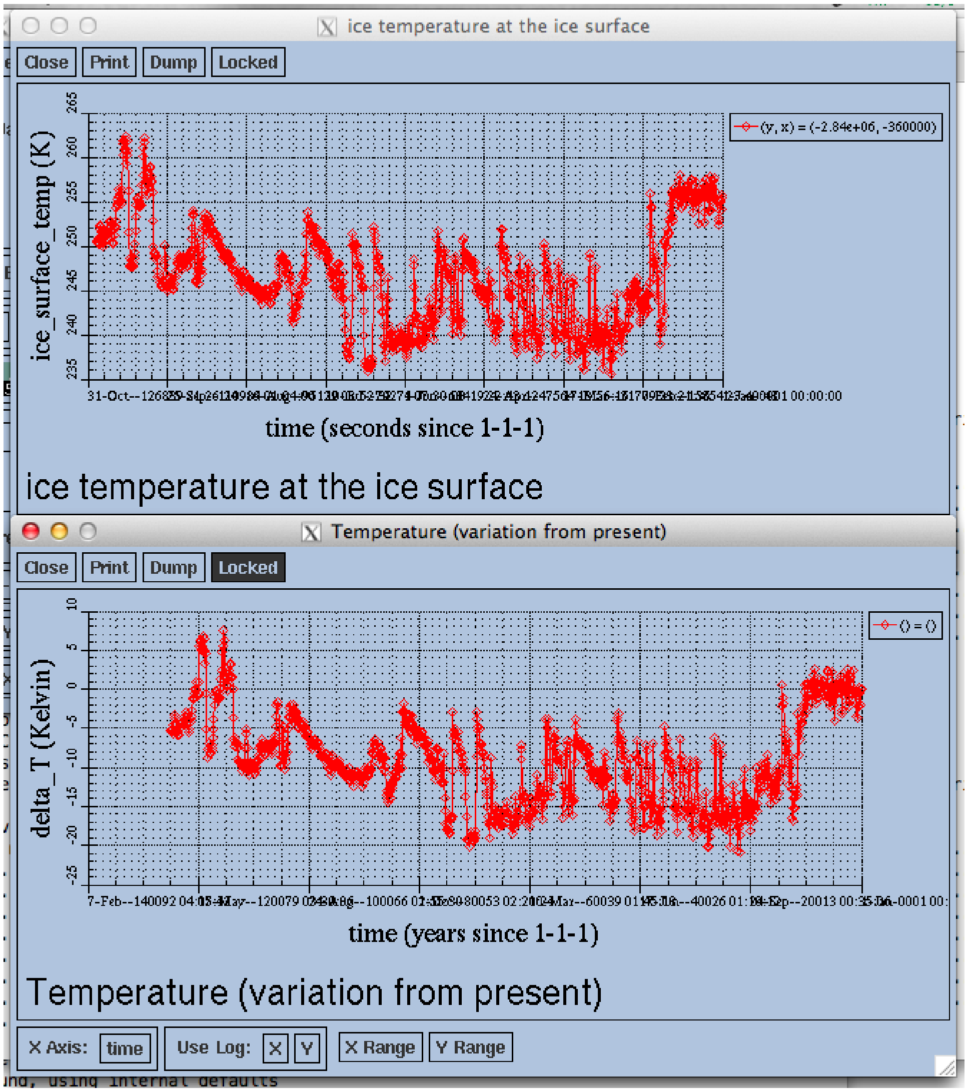


Figure 3: Time series of the surface temperature compared to GRIP temperature offsets

5 Surface mass and energy process model components

5.1 The “invisible” model

Command-line option:	`-surface simple`
NetCDF variables:	none
C++ class:	`PSSimple`

This is the simplest “surface model” available in PISM, enabled using `-surface simple`. Its job is to re-interpret precipitation as climatic mass balance, and to re-interprets mean annual near-surface (2m) air temperature as the temperature of the ice at the depth at which firn processes cease to change the temperature of the ice. (I.e. the temperature *below* the firn.) This implies that there is no melt. Though primitive, this model component may be desired in cold environments (e.g. East Antarctic ice sheet) in which melt is negligible and heat from firn processes is ignored.

5.2 Reading top-surface boundary conditions from a file

Command-line option:	`-surface given`
NetCDF variables:	`ice_surface_temp`, `climatic_mass_balance` [kg/m^2/s]
C++ class:	`PSGivenClimate`

This is the default choice.

This model component was created to force PISM with sampled (possibly periodic) climate data by reading ice upper surface boundary conditions from a file. These fields are provided directly to the ice dynamics code (see User’s Manual, table 3).

PISM will stop if variables `ice_surface_temp` (ice temperature at the ice surface but below firn) and `climatic_mass_balance` (top surface mass flux into the ice) are not present in the input file.

Command-line options:

- `-surface_given_file` *filename* prescribes an input file
- `-surface_given_period` *years* makes PISM interpret data in `-surface_given_file` as periodic. See section 2.1.
- `-surface_given_reference_year` sets the reference model year; see section 2.1.

A file `foo.nc` used with `-surface given -surface_given_file foo.nc` should contain several records. If this file contains one record (i.e. fields corresponding to one time value only), provided forcing data is interpreted as time-independent. The `time` variable should describe what model time these records correspond to; see section 2 for details.

For example, to use monthly records and period of 1 year, create a file (say, “`foo.nc`”) with 12 records. The `time` variable may contain 0, 1, 2, 3, . . . , 11 and have the units of “month” (you can use other units, too). Then, run

```
$ pismr -surface given -surface_given_file foo.nc -surface_given_period 1
```

Notes:

- This surface model *ignores* the atmosphere model selection made using the `-atmosphere` option.
- PISM can handle files with virtually any number of records: it will read and store in memory at most `climate_forcing_buffer_size` records at any given time (default: 60, or 5 years' worth of monthly fields).
- when preparing a file for use with this model, it is best to use the `t,x,y` variable storage order: files using this order can be read in faster than ones using the `t,y,x` order, for reasons explained in the User's Manual, section 9.1.1.

  To change the storage order in a NetCDF file, use `ncpdq`:

  ```
  $ ncpdq -a t,x,y input.nc output.nc
  ```

  will copy data from `input.nc` into `output.nc`, changing the storage order to `t,x,y` at the same time.

5.3 Elevation-dependent temperature and mass balance

Command-line option:	`-surface elevation`
NetCDF variables:	none
C++ class:	`PSElevation`

This surface model component parameterizes the ice surface temperature T_h = `ice_surface_temp` and the mass balance m = `climatic_mass_balance` as *piecewise-linear* functions of surface elevation h.

The option `-ice_surface_temp` *list of 4 numbers* determines the surface temperature using the 4 parameters T_{min}, T_{max}, h_{min}, h_{max}. Let

$$\frac{\mathrm{d}T}{\mathrm{d}h} = (T_{\text{max}} - T_{\text{min}})/(h_{\text{max}} - h_{\text{min}}) \tag{1}$$

be the temperature gradient. Then

$$T(x,y) = \begin{cases} T_{\text{min}}, & h(x,y) \le h_{\text{min}}, \\ T_{\text{min}} + \frac{\mathrm{d}T}{\mathrm{d}h}\,(h(x,y) - h_{\text{min}}), & h_{\text{min}} < h(x,y) < h_{\text{max}}, \\ T_{\text{max}}, & h_{\text{max}} \le h(x,y). \end{cases} \tag{2}$$

The option `-climatic_mass_balance` *list of 5 numbers* determines the surface mass balance using the 5 parameters m_{min}, m_{max}, h_{min}, h_{ELA}, h_{max}. Let

$$\frac{\mathrm{d}m_{\text{abl}}}{\mathrm{d}h} = -m_{\text{min}}/(h_{\text{max}} - h_{\text{min}}) \tag{3}$$

and

$$\frac{\mathrm{d}m_{\mathrm{acl}}}{\mathrm{d}h} = m_{\max}/(h_{\max} - h_{\min}) \tag{4}$$

be the mass balance gradient in the ablation and in the accumulation area, respectively. Then

$$m(x,y) = \begin{cases} m_{\min}, & h(x,y) \le h_{\min}, \\ \frac{\mathrm{d}m_{\mathrm{abl}}}{\mathrm{d}h}\left(h(x,y) - h_{\mathrm{ELA}}\right), & h_{\min} < h(x,y) < h_{\max}, \\ \frac{\mathrm{d}m_{\mathrm{acl}}}{\mathrm{d}h}\left(h(x,y) - h_{\mathrm{ELA}}\right), & h_{\min} < h(x,y) < h_{\max}, \\ m_{\max}, & h_{\max} \le h(x,y). \end{cases} \tag{5}$$

The option `-climatic_mass_balance_limits` *list of 2 numbers* limits the mass balance below $h_{\min}$ to ${m^*}_{\min}$ and above $h_{\max}$ to ${m^*}_{\max}$, thus

$$m(x,y) = \begin{cases} m^*_{\min}, & h(x,y) \le h_{\min}, \\ \frac{\mathrm{d}m_{\mathrm{abl}}}{\mathrm{d}h}\left(h(x,y) - h_{\mathrm{ELA}}\right), & h_{\min} < h(x,y) < h_{\max}, \\ \frac{\mathrm{d}m_{\mathrm{acl}}}{\mathrm{d}h}\left(h(x,y) - h_{\mathrm{ELA}}\right), & h_{\min} < h(x,y) < h_{\max}, \\ m^*_{\max}, & h_{\max} \le h(x,y). \end{cases} \tag{6}$$

Note: this surface model *ignores* the atmosphere model selection made using the `-atmosphere` option.

5.4 Temperature-index (positive degree-day) scheme

Command-line option:	`-surface pdd`
NetCDF variables:	`air_temp_sd`, `snow_depth`
C++ class:	`PSTemperatureIndex`

The default PDD model used by PISM, turned on by option `-surface pdd`, is based on [2] and EISMINT-Greenland intercomparison (see [12]).

Our model computes the solid (snow) precipitation rate using the air temperature threshold with a linear transition. All precipitation during periods with air temperatures above `air_temp_all_precip_as_rain` (default of $2°C$) is interpreted as rain; all precipitation during periods with air temperatures below `air_temp_all_precip_as_snow` (default of $0°C$) is interpreted as snow.

For long-term simulations, a PDD model generally uses an idealized seasonal temperature cycle. “White noise” is added to this cycle to simulate additional daily variability associated to the vagaries of weather. This additional random variation is quite significant, as the seasonal cycle may never reach the melting point but that point may be reached with some probability, in the presence of the daily variability, and thus melt may occur. Concretely, a normally-distributed, mean zero random temperature increment is added to the seasonal cycle. There is no assumed spatial correlation of daily variability. The standard deviation of the daily variability is controlled by command-line options:

- `-pdd_sd_file` *filename*, which prescribes an input file.
- `-pdd_sd_period` *years*, which interprets its data as periodic; see section 2.1.

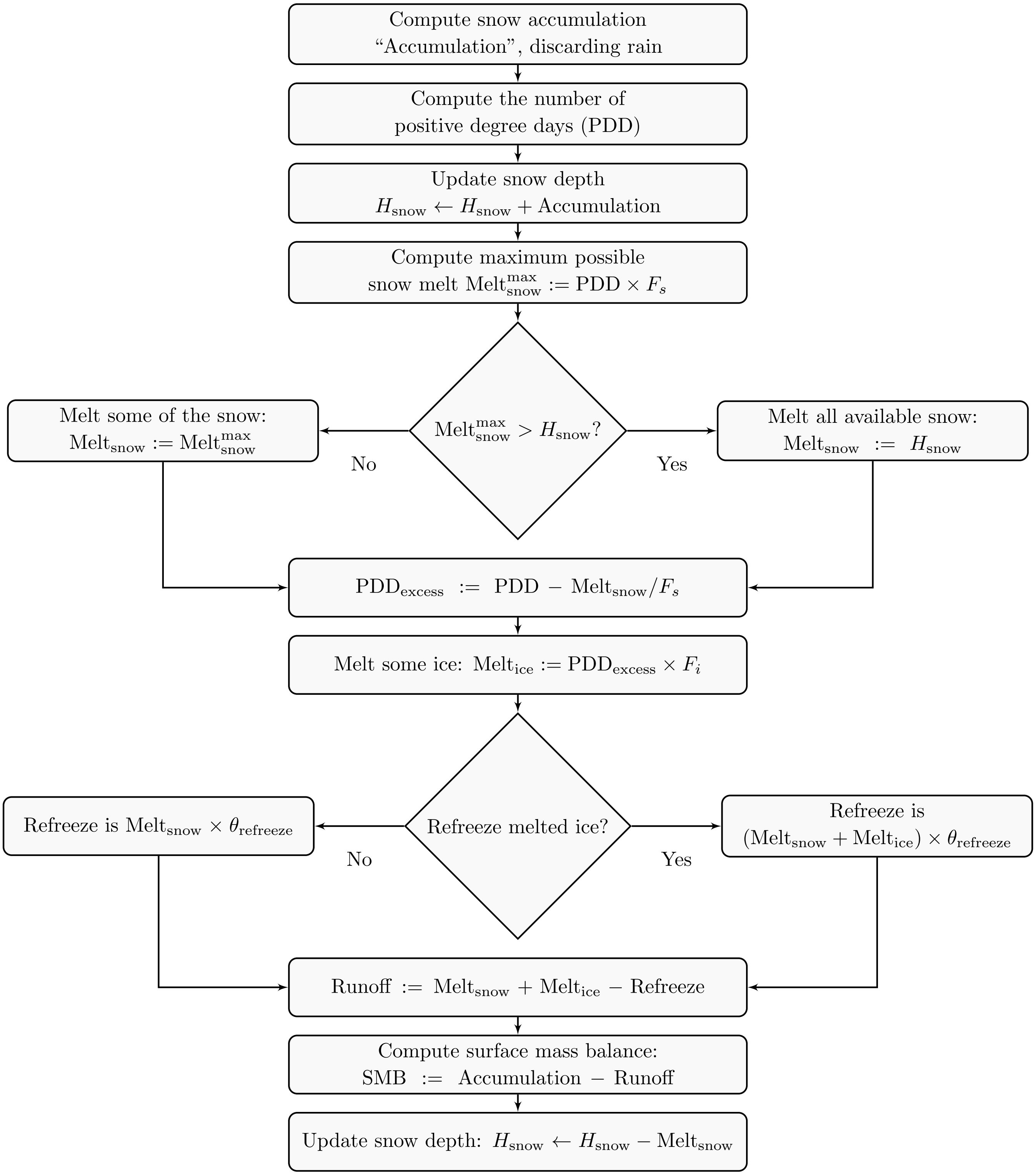


Figure 4: PISM's positive degree day model. F_s and F_i are PDD factors for snow and ice, respectively; θ_{refreeze} is the refreeze fraction.

- `-pdd_sd_reference_year`, which sets the reference model year; see section 2.1.

A file `foo.nc` used with `-surface pdd -pdd_sd_file foo.nc` should contain standard deviation of near-surface air temperature in variable `air_temp_sd`, and the corresponding time coordinate in variable `time`. If `-pdd_sd_file` is not set, PISM uses a constant value for standard deviation, which is set by the `pdd_std_dev` configuration parameter. The default value is 5.0 degrees [12]. However, this approach is not recommended as it induces significant errors in modeled surface mass balance in both ice-covered and ice-free regions [13, 14].

Over ice-covered grid cells, daily variability can also be parameterized as a linear function of near-surface air temperature $\sigma = a * T + b$ using the `pdd_std_dev_use_param` configuration flag, and the corresponding parameters `pdd_std_dev_param_a` and `pdd_std_dev_param_b`. This parametrization replaces prescribed standard deviation values over glacierized grid cells as defined by the `mask` variable (see `mask_icefree_thickness_standard`). Default values for the slope a and intercept b were derived from the ERA-40 reanalysis over the Greenland ice sheet [15].

The number of positive degree days is computed as the magnitude of the temperature excursion above $0\,^{\circ}$C multiplied by the duration (in days) when it is above zero.

In PISM there are two methods for computing the number of positive degree days. The first computes only the expected value, by the method described in [2]. This is the default when a PDD is chosen (i.e. option `-surface pdd`). The second is a Monte Carlo simulation of the white noise itself, chosen by adding the option `-pdd_rand`. This Monte Carlo simulation adds the same daily variation at every point, though the seasonal cycle is (generally) location dependent. If repeatable randomness is desired use `-pdd_rand_repeatable` instead of `-pdd_rand`.

By default, the computation summarized in Figure 4 is performed every week. (This frequency is controlled by the `pdd_max_evals_per_year` parameter.) To compute mass balance during each week-long time-step, PISM keeps track of the current snow depth (using units of ice-equivalent thickness). This is necessary to determine if melt should be computed using the degree day factor for snow (`pdd_factor_snow`) or the corresponding factor for ice (`pdd_factor_ice`).

A fraction of the melt controlled by the configuration parameter `pdd_refreeze` (θ_{refreeze} in Figure 4, default: 0.6) refreezes. The user can select whether melted ice should be allowed to refreeze using the `pdd_refreeze_ice_melt` configuration flag.

Since PISM does not have a principled firn model, the snow depth is set to zero at the beginning of the balance year. See `pdd_balance_year_start_day`. Default is 274, corresponding to October 1$^{\text{st}}$.

Our PDD implementation is meant to be used with an atmosphere model implementing a cosine yearly cycle such as `searise_greenland` (section 6.3), but it is not restricted to parameterizations like these.

This code also implements latitude- and mean July temperature dependent ice and snow factors using formulas (6) and (7) in [3]; set `-pdd_fausto` to enable. The default standard deviation of the daily variability (`-pdd_std_dev` option) is 2.53 degrees under the `-pdd_fausto` option [3]. See also configuration parameters with the `pdd_fausto` prefix.

Note that when used with periodic climate data (air temperature and precipitation) that is read from a file (see section 6.1), use of `-timestep_hit_multiplies X` is recommended. (Here `X` is the length of the climate data period in years.)

5.5 PIK

Command-line option:	`-surface pik`
NetCDF variables:	`climatic_mass_balance` [kg/m^2/s], `lat` (latitude), [degrees north]
C++ class:	`PSConstantPIK`

This surface model component implements the setup used in [10]. The `climatic_mass_balance` is read from an input (`-i`) file; the ice surface temperature is computed as a function of latitude (variable `lat`) and surface elevation (dynamically updated by PISM). See equation (1) in [10].

5.6 Modifier: Scalar temperature offsets

Command-line option:	`-surface ...,delta_T`
NetCDF variables:	`delta_T`
C++ class:	`PS_delta_T`

Command-line options:

- `-surface_delta_T_file` *filename* sets the name of the file PISM will read `delta_T` from.
- `-surface_delta_T_period` *years* sets the period of the forcing data (section 2.1)
- `-surface_delta_T_reference_year` sets the reference year (section 2.1).

The time-dependent scalar offsets `delta_T` are added to `ice_surface_temp` computed by a surface model.

Please make sure that `delta_T` has the units of "`Kelvin`".

This modifier is identical to the corresponding atmosphere modifier, but applies offsets at a different stage in the computation of top-surface boundary conditions needed by the ice dynamics core.

5.7 Modifier: Lapse rate corrections

Command-line option:	`-surface ...,lapse_rate`
NetCDF variables:	`surface_altitude` (CF standard name),
C++ class:	`PSLapseRates`

The `lapse_rate` modifier allows correcting ice-surface temperature and surface mass balance using elevation lapse rates. It uses the following options.

- `-temp_lapse_rate` gives the temperature lapse rate, in K/km. Note that we use the following definition of the temperature lapse rate:

$$\gamma = -\frac{dT}{dz}.$$

- `-smb_lapse_rate` gives the surface mass balance lapse rate, in $m/year/km$. Here, $\gamma = -\frac{dM}{dz}$.
- `-surface_lapse_rate_file` *filename* specifies the file containing the reference surface elevation field (standard name: `surface_altitude`). This file can contain several surface elevation records to use lapse rate corrections relative to time-dependent surface. If one record is provided, the reference surface elevation is assumed to be time-independent.
- `-surface_lapse_rate_period` gives the period, in model years, to use when interpreting data in the file given with `-surface_given_file`,
- `-surface_lapse_rate_reference_year` takes the time T in model years. The record for t years in `-surface_given_file` is interpreted as corresponding to t years since T.

5.8 Modifier: Surface mass flux adjustment

Command-line option:	`-surface ...,forcing`
NetCDF variables:	`thk` (ice thickness), `ftt_mask` (mask of zeros and ones; 1 where surface mass flux is adjusted and 0 elsewhere)
C++ class:	`PSForceThickness`

The `forcing` modifier implements a surface mass balance adjustment mechanism which forces the thickness of grounded ice to a target thickness distribution at the end of the run. The idea behind this mechanism is that spinup of ice sheet models frequently requires the surface elevation to come close to measured values at the end of a run. A simpler alternative to accomplish this, namely option `-no_mass`, represents an unmodeled, frequently large, violation of the mass continuity equation.

In more detail, let H_{tar} be the target thickness. Let H be the time-dependent model thickness. The surface model component described here produces the term M in the mass continuity equation:

$$\frac{\partial H}{\partial t} = M - S - \nabla \cdot \mathbf{q}.$$

(Other details of this equation do not concern us here.) The `forcing` modifier causes M to be adjusted by a multiple of the difference between the target thickness and the current thickness,

$$\Delta M = \alpha (H_{\text{tar}} - H)$$

where $\alpha > 0$. We are adding mass ($\Delta M > 0$) where $H_{\text{tar}} > H$ and ablating where $H_{\text{tar}} < H$.

Option `-force_to_thickness_file` *filename* identifies the file containing the target ice thickness field `thk` and the mask `ftt_mask`. A basic run modifying surface model `given` would look like

```
$ pismr -i foo.nc -surface given,forcing -force_to_thickness_file bar.nc
```

In this case `foo.nc` contains fields `climatic_mass_balance` and `ice_surface_temp`, as normal for `-surface given`, and `bar.nc` contains fields `thk` which will serve as the target thickness and `ftt_mask` which defines the map plane area where this adjustment is applied. Option `-force_to_thickness_alpha` adjusts the value of α, which has a default value specified in the *Source Code Browser* `http://www.pism-docs.org/wiki/doku.php?id=browser`.

In addition to this one can specify a multiplicative factor C used in areas where the target thickness field has less than `-force_to_thickness_ice_free_thickness_threshold` meters of ice; $\alpha_{\text{ice free}} = C \times \alpha$. Use the `-force_to_thickness_ice_free_alpha_factor` option to set C.

5.9 Modifier: Anomalies

Command-line option:	`-surface ...,anomaly`
NetCDF variables:	`ice_surface_temp_anomaly`, `climatic_mass_balance_anomaly` [kg/m^2/s]
C++ class:	`PSAnomaly`

This modifier implements a spatially-variable version of `-surface ...,delta_T` which also applies time-dependent climatic mass balance anomalies.

It takes the following options:

- `-surface_anomaly_file` *filename* specifies a file containing variables `ice_surface_temp_anomaly` and `climatic_mass_balance_anomaly`.
- `-surface_anomaly_period` years specifies the period of the forcing data, in model years; section 2.1
- `-surface_anomaly_reference_year` specifies the reference year; section 2.1.

See also to `-atmosphere ...,anomaly` (section 6.11), which is similar, but applies anomalies at the atmosphere level.

5.10 Modifier: Caching

Command-line option:	`-surface ...,cache`
C++ class:	`PSCache`

This modifier skips surface model updates, so that a surface model is called no more than every `-surface_cache_update_interval` years. A time-step of 1 year is used every time a surface model is updated.

This is useful in cases when inter-annual climate variability is important, but one year differs little from the next. (Coarse-grid paleo-climate runs, for example.)

It takes the following options:

- `-surface_cache_update_interval` *years* Specifies the minimum interval between updates. PISM may take longer time-steps if the adaptive scheme allows it, though.

See also section 7.9.

6 Atmosphere model components

6.1 Reading atmosphere boundary conditions from a file

Command-line option:	`-atmosphere given`
NetCDF variables:	`air_temp`, `precipitation` [m/s ice equivalent]
C++ class:	`PAGivenClimate`

This is the default choice.

Command-line options:

- `-atmosphere_given_file` *filename* prescribes an input file
- `-atmosphere_given_period` *years* makes PISM interpret data in `-atmosphere_given_file` as periodic. See section 2.1.
- `-atmosphere_given_reference_year` sets the reference model year; see section 2.1.

A file `foo.nc` used with `-atmosphere given -atmosphere_given_file foo.nc` should contain several records; the `time` variable should describe what model time these records correspond to.

This model component was created to force PISM with sampled (possibly periodic) climate data, e.g. using monthly records of `air_temp` and `precipitation`.

It can also used to drive a temperature-index (PDD) climatic mass balance computation (section 5.4).

See also section 5.2, which describes a similar surface model component (`-surface given`).

6.2 Cosine yearly cycle

Command-line option:	`-atmosphere yearly_cycle`
NetCDF variables:	`air_temp_mean_annual`, `air_temp_mean_july`, `precipitation` [m/s ice equivalent] `amplitude_scaling`
C++ class:	`PACosineYearlyCycle`

This atmosphere model component computes the near-surface air temperature using the following formula:

$$T(\text{time}) = T_{\text{mean annual}} + A(\text{time}) \cdot (T_{\text{mean July}} - T_{\text{mean annual}}) \cdot \cos(2\pi t),$$

where t is the year fraction “since last July”; the summer peak of the cycle is on `snow_temp_july_day`, which is set to day 196 by default (approximately July 15).

Here $T_{\text{mean annual}}$ (variable `air_temp_mean_annual`) and $T_{\text{mean July}}$ (variable `air_temp_mean_july`) are read from a file selected using the `-atmosphere_yearly_cycle_file` command-line option. A time-independent precipitation field (variable `precipitation`) is read from the same file.

Optionally a time-dependent scalar amplitude scaling $A(t)$ can be used. Specify a file to read it from using the `-atmosphere_yearly_cycle_scaling_file` command-line option. Without this option $A(\text{time}) \equiv 1$.

6.3 SeaRISE-Greenland

Command-line option:	`-atmosphere searise_greenland`
NetCDF variables:	`lon`, `lat`, `precipitation` [m/s ice equivalent]
C++ class:	`PASeariseGreenland`

This atmosphere model component implements a longitude, latitude, and elevation dependent near-surface air temperature parameterization and a cosine yearly cycle described in [3] and uses a constant in time ice-equivalent precipitation field (in units of thickness per time, variable `precipitation`) that is read from an input (`-i`) file. To read time-independent precipitation from a different file, use the option `-atmosphere_searise_greenland_file`.

The air temperature parameterization is controlled by configuration parameters with the `snow_temp_fausto` prefix.

See also the `-atmosphere ...,paleo_precip` modifier, section 6.9, for an implementation of the SeaRISE-Greenland formula for paleo-precipitation correction from present; a 7.3% change of precipitation rate for every one degree Celsius of temperature change [7].

6.4 PIK

Command-line option:	`-atmosphere pik`
NetCDF variables:	`lat`, `precipitation`
C++ class:	`PAConstantPIK`

This model component reads a time-independent precipitation field from an input (`-i`) file and computes near-surface air temperature using a latitude and surface elevation-dependent formula.

The parameterization is the same as in the `-surface pik` model, section 5.5.

6.5 One weather station

Command-line options:	`-atmosphere one_station`, `-atmosphere_one_station_file`
NetCDF variables:	`air_temp` [Kelvin], `precipitation` [m / s ice equivalent]
C++ class:	`PAWeatherStation`

This model component reads scalar time-series of the near-surface air temperature and precipitation from a file specified using the `-atmosphere_one_station_file` option and uses them at *all* grid points in the domain. In other words, resulting climate fields are constant in space but not necessarily in time.

The `-atmosphere one_station` model should be used with a modifier such as `lapse_rate` (see section 6.10) to create spatial variablitity.

6.6 Modifier: Scalar temperature offsets

Command-line option:	`-atmosphere ...,delta_T`
NetCDF variables:	`delta_T`
C++ class:	`PA_delta_T`

This modifier applies scalar time-dependent air temperature offsets to the output of an atmosphere model. It takes the following command-line options.

- `-atmosphere_delta_T_file` *filename* sets the name of the file PISM will read `delta_T` from.
- `-atmosphere_delta_T_period` *years* sets the period of the forcing data (section 2.1).
- `-atmosphere_delta_T_reference_year` sets the reference year (section 2.1).

Please make sure that `delta_T` has the units of "`Kelvin`".

6.7 Modifier: Scalar precipitation offsets

Command-line option:	`-atmosphere ...,delta_P`
NetCDF variables:	`delta_P` [m/s ice equivalent]
C++ class:	`PA_delta_P`

This modifier applies scalar time-dependent precipitation offsets to the output of an atmosphere model. It takes the following command-line options.

- `-atmosphere_delta_P_file` *filename* sets the name of the file PISM will read `delta_P` from.
- `-atmosphere_delta_P_period` *years* sets the period of the forcing data (section 2.1).
- `-atmosphere_delta_P_reference_year` sets the reference year (section 2.1).

6.8 Modifier: Scalar precipitation scaling

Command-line option:	`-atmosphere ...,frac_P`
NetCDF variables:	`frac_P` [no unit]
C++ class:	`PA_frac_P`

This modifier scales precipitation output of an atmosphere model using a scalar time-dependent precipitation fraction, with a value of one corresponding to no change in precipitation. It takes the following command-line options:

- `-atmosphere_frac_P_file` *filename* sets the name of the file PISM will read `frac_P` from.
- `-atmosphere_frac_P_period` *years* sets the period of the forcing data (section 2.1).
- `-atmosphere_frac_P_reference_year` sets the reference year (section 2.1).

6.9 Modifier: Paleo-precipitation correction using scalar temperature offsets

Command-line option:	`-atmosphere ...,paleo_precip`
NetCDF variables:	`delta_T` [degrees Kelvin]
C++ class:	`PA_paleo_precip`

This modifier implements the SeaRISE-Greenland formula for paleo-precipitation correction from present; a 7.3% change of precipitation rate for every one degree Celsius of air temperature change [7]. See `http://websrv.cs.umt.edu/isis/index.php/Model_Initialization#Greenland` for details. The input file should contain air temperature offsets in the format used by `-atmosphere ...,delta_T` modifier, see section 6.6.

It takes the following command-line options.

- `-atmosphere_paleo_precip_file` *filename* sets the name of the file PISM will read `delta_T` from.
- `-atmosphere_paleo_precip_period` *years* sets the period of the forcing data (section 2.1).
- `-atmosphere_paleo_precip_reference_year` sets the reference year (section 2.1).

6.10 Modifier: Lapse rate corrections

Command-line option:	`-atmosphere ...,lapse_rate`
NetCDF variables:	`surface_altitude` (CF standard name)
C++ class:	`PALapseRates`

The `lapse_rate` modifier allows for correcting air temperature and precipitation using elevation lapse rates. It uses the following options.

- `-temp_lapse_rate` gives the temperature lapse rate, in K/km. Note that we use the following definition of the temperature lapse rate:

$$\gamma = -\frac{dT}{dz}.$$

- `-precip_lapse_rate` gives the precipitation lapse rate, in $m/year/km$. Here $\gamma = -\frac{dM}{dz}$.
- `-atmosphere_lapse_rate_file` *filename* specifies a file containing the reference surface elevation field (standard name: `surface_altitude`). This file may contain several surface elevation records to use lapse rate corrections relative to a time-dependent surface. If one record is provided, the reference surface elevation is assumed to be time-independent.
- `-atmosphere_lapse_rate_period` gives the period, in model years; see section 2.1.
- `-atmosphere_lapse_rate_reference_year` specifies the reference date; see section 2.1.

6.11 Modifier: Anomalies

Command-line option:	`-atmosphere ...,anomaly`
NetCDF variables:	`air_temp_anomaly`, `precipitation_anomaly` [m/s ice equivalent]
C++ class:	`PAAnomaly`

This modifier implements a spatially-variable version of `-atmosphere ...,delta_T,delta_P`. It takes the following options:

- `-atmosphere_anomaly_file` *filename* specifies a file containing variables `air_temp_anomaly` and `precipitation_anomaly`.
- `-atmosphere_anomaly_period` years specifies the period of the forcing data, in model years; section 2.1.
- `-atmosphere_anomaly_reference_year` specifies the reference year; section 2.1.

See also to `-surface ...,anomaly` (section 5.9), which is similar, but applies anomalies at the surface level.

7 Ocean model components

PISM Ocean model components provide sub-shelf ice temperature (`shelfbtemp`) and sub-shelf mass flux (`shelfbmassflux`) to the ice dynamics core.

The sub-shelf ice temperature is used as a Dirichlet boundary condition in the energy conservation code. The sub-shelf mass flux is used as a source in the mass-continuity (transport) equation. Positive flux corresponds to ice loss; in other words, this sub-shelf mass flux is a “melt rate”.

7.1 Constant in time and space

Command-line option:	`-ocean constant`
NetCDF variables:	none
C++ class:	`POConstant`

This is the default choice.

This ocean model component implements boundary conditions at the ice/ocean interface that are constant **both** in space and time.

The sub-shelf ice temperature is set to pressure melting and the sub-shelf melt rate is assumed to be proportional to the heat flux from the ocean into the ice (configuration parameter `ocean_sub_shelf_heat_flux_into_ice`).

Alternatively, the sub-shelf melt rate in meters per year can be set using the `-shelf_base_melt_rate` command-line option.

7.2 Reading forcing data from a file

Command-line option:	`-ocean given`
NetCDF variables:	`shelfbtemp` [degrees Kelvin], `shelfbmassflux` [kg m-2 s-1]
C++ class:	`POGivenClimate`

This ocean model component reads sub-shelf ice temperature `shelfbtemp` and the sub-shelf mass flux `shelfbmassflux` from a file. It takes the following command-line options.

- `-ocean_given_file` *filename*: sets the name of the file to read forcing data from. The file may contain several records. If only one record is provided it is interpreted as time-independent.
- `-ocean_given_reference_year` specifies the reference date; see section 2.1.
- `-ocean_given_period` specifies the length of the period of the forcing data, in model years; see section 2.1.

Variables `shelfbtemp` and `shelfbmassflux` may be time-dependent. (The `-ocean given` component is very similar to `-surface given` and `-atmosphere given`.)

7.3 PIK

Command-line option:	`-ocean pik`
NetCDF variables:	none
C++ class:	`POConstantPIK`

This ocean model component implements the ocean forcing setup used in [10]. The sub-shelf ice temperature is set to pressure-melting; the sub-shelf mass flux computation follows [1].

It takes one command-line option:

- `-meltfactor_pik`: a melt factor F_{melt} in sub-shelf-melting parameterization, see equation (5) in [10].

7.4 Basal melt rate and temperature from thermodynamics in boundary layer

Command-line option:	`-ocean th`
NetCDF variables:	`theta_ocean` (absolute potential ocean temperature), [Kelvin], `salinity_ocean` (salinity of the adjacent ocean), [g/kg]
C++ class:	`POGivenTH`

This ocean model component derives basal melt rate and basal temperature from thermodynamics in a boundary layer at the base of the ice shelf. It uses a set of three equations describing

1. the energy flux balance,
2. the salt flux balance,
3. the pressure and salinity dependent freezing point in the boundary layer.

This model is described in [6] and [4].

Inputs are potential temperature (variable `theta_ocean`) and salinity (variable `salinity_ocean`) read from a file.

No ocean circulation is modeled, so melt water computed by this model is not fed back into the surrounding ocean.

This implementation uses different approximations of the temperature gradient at the base of an ice shelf column depending on whether there is sub-shelf melt, sub-shelf freeze-on, or neither (see [6] for details).

It takes two command-line option:

- `-ocean_th_file` *filename*: specifies the NetCDF file providing potential temperature and salinity fields.
- `-clip_shelf_base_salinity`: if this is set (which is the default), the sub-shelf salinity is clipped so that it stays in the $[4, 40]$ psu range. This is done to ensure that we stay in the range of applicability of the melting point temperature parameterization; see [6]. To disable salinity clipping, use the `-no_clip_shelf_base_salinity` option or set the `ocean_three_equation_model_clip_salinity` configuration parameter to “no”.

7.5 Modifier: Scalar sea level offsets

Command-line option:	`-ocean ...,delta_SL`
NetCDF variables:	`delta_SL` [meters]
C++ class:	`PO_delta_SL`

The `delta_SL` modifier implements sea level forcing using scalar offsets.
It takes the following command-line options:

- `-ocean_delta_SL_file` *filename*: specifies the name of the file containing forcing data. This file has to contain the `delta_SL` variable using units "meters" or equivalent.
- `-ocean_delta_SL_period` specifies the length of the period of the forcing data, in model years; see section 2.1.
- `-ocean_delta_SL_reference_year` specifies the reference date; see section 2.1.

7.6 Modifier: Scalar sub-shelf temperature offsets

Command-line option:	`-ocean ...,delta_T`
NetCDF variables:	`delta_T` [Kelvin]
C++ class:	`PO_delta_T`

This modifier implements forcing using sub-shelf ice temperature offsets.
It takes the following command-line options:

- `-ocean_delta_T_file` *filename*: specifies the name of the file containing forcing data. This file has to contain the `delta_T` variable using units of "Kelvin" or equivalent.
- `-ocean_delta_T_period` specifies the length of the period of the forcing data, in model years; see section 2.1.
- `-ocean_delta_T_reference_year` specifies the reference date; see section 2.1.

7.7 Modifier: Scalar sub-shelf mass flux offsets

Command-line option:	`-ocean ...,delta_SMB`
NetCDF variables:	`delta_SMB`
C++ class:	`PO_delta_SMB`

This modifier implements forcing using sub-shelf mass flux (melt rate) offsets.
It takes the following command-line options:

- `-ocean_delta_SMB_file` *filename*: specifies the name of the file containing forcing data. This file has to contain the `delta_SMB` variable using units of ice-equivalent thickness per time.
- `-ocean_delta_SMB_period` specifies the length of the period of the forcing data, in model years; see section 2.1.
- `-ocean_delta_SMB_reference_year` specifies the reference date; see section 2.1.

7.8 Modifier: Scalar melange back pressure fraction offsets

Command-line option:	`-ocean ...,delta_MBP`
NetCDF variables:	`delta_MBP`
C++ class:	`PO_delta_MBP`

This modifier implements forcing using melange back pressure fraction offsets. The variable `delta_MBP` should take on values from 0 to 1; it is understood as the fraction of the maximum melange back pressure possible at a given location. (We assume that melange back pressure cannot exceed the pressure of the ice column at a calving front.)

Please see section 8.4 of the *User's Manual* for details.

This modifier takes the following command-line options:

- `-ocean_delta_MBP_file` *filename*: specifies the name of the file containing forcing data. This file has to contain the `delta_MBP` variable using units of "1" (a dimensionless parameter)
- `-ocean_delta_MBP_period` specifies the length of the period of the forcing data, in model years; see section 2.1.
- `-ocean_delta_MBP_reference_year` specifies the reference date; see section 2.1.

7.9 Modifier: Caching

Command-line option:	`-ocean ...,cache`
C++ class:	`POCache`

This modifier skips ocean model updates, so that a ocean model is called no more than every `-ocean_cache_update_interval` years. A time-step of 1 year is used every time a ocean model is updated.

This is useful in cases when inter-annual climate variability is important, but one year differs little from the next. (Coarse-grid paleo-climate runs, for example.)

It takes the following options:

- `-ocean_cache_update_interval` *years* Specifies the minimum interval between updates. PISM may take longer time-steps if the adaptive scheme allows it, though.

See also section 5.10.

References

[1] A. Beckmann and H. Goosse, *A parameterization of ice shelf-ocean interaction for climate models*, Ocean Modelling, 5 (2003), pp. 157–170.

[2] R. Calov and R. Greve, *Correspondence: A semi-analytical solution for the positive degree-day model with stochastic temperature variations*, J. Glaciol, 51 (2005), pp. 173–175.

[3] R. S. Fausto, A. P. Ahlstrom, D. V. As, C. E. Boggild, and S. J. Johnsen, *A new present-day temperature parameterization for Greenland*, J. Glaciol., 55 (2009), pp. 95–105.

[4] H. H. Hellmer, S. S. Jacobs, and A. Jenkins, *Oceanic erosion of a floating Antarctic glacier in the Amundsen Sea*, American Geophysical Union, 1998.

[5] R. Hock, *Glacier melt: a review of processes and their modelling*, Prog. Phys. Geog., 29 (2005), pp. 362–391.

[6] D. M. Holland and A. Jenkins, *Modeling thermodynamic ice-ocean interactions at the base of an ice shelf*, Journal of Physical Oceanography, 29 (1999), pp. 1787–1800.

[7] P. Huybrechts, *Sea-level changes at the LGM from ice-dynamic reconstructions of the Greenland and Antarctic ice sheets during the glacial cycles*, Quat. Sci. Rev., 21 (2002), pp. 203–231.

[8] J. Imbrie and eight others, *The orbital theory of Pleistocene climate: Support from a revised chronology of the marine* $\delta^{18}O$ *record*, in Milankovitch and Climate: Understanding the Response to Astronomical Forcing, D. Reidel, 1984, pp. 269–305.

[9] S. J. Johnsen, D. Dahl-Jensen, W. Dansgaard, and N. Gundestrup, *Greenland paleotemperatures derived from GRIP bore hole temperature and ice core isotope profiles*, Tellus, 47B (1995), pp. 624–629.

[10] M. A. Martin, R. Winkelmann, M. Haseloff, T. Albrecht, E. Bueler, C. Khroulev, and A. Levermann, *The Potsdam Parallel Ice Sheet Model (PISM-PIK) –Part 2: Dynamic equilibrium simulation of the Antarctic ice sheet*, The Cryosphere, 5 (2011), pp. 727–740.

[11] A. Payne et al., *Results from the EISMINT model intercomparison: the effects of thermomechanical coupling*, J. Glaciol., 153 (2000), pp. 227–238.

[12] C. Ritz, *EISMINT Intercomparison Experiment: Comparison of existing Greenland models.* homepages.vub.ac.be/∼phuybrec/eismint/greenland.html, 1997.

[13] I. Rogozhina and D. Rau, *Vital role of daily temperature variability in surface mass balance parameterizations of the greenland ice sheet*, The Cryosphere, 8 (2014), pp. 575–585.

[14] J. Seguinot, *Spatial and seasonal effects of temperature variability in a positive degree day surface melt model*, J. Glaciol., 59 (2013), pp. 1202–1204.

[15] J. Seguinot and I. Rogozhina, *Daily temperature variability predetermined by thermal conditions over ice sheet surfaces*, J. Glaciol., (2014).

General Index

PISM Command-line options