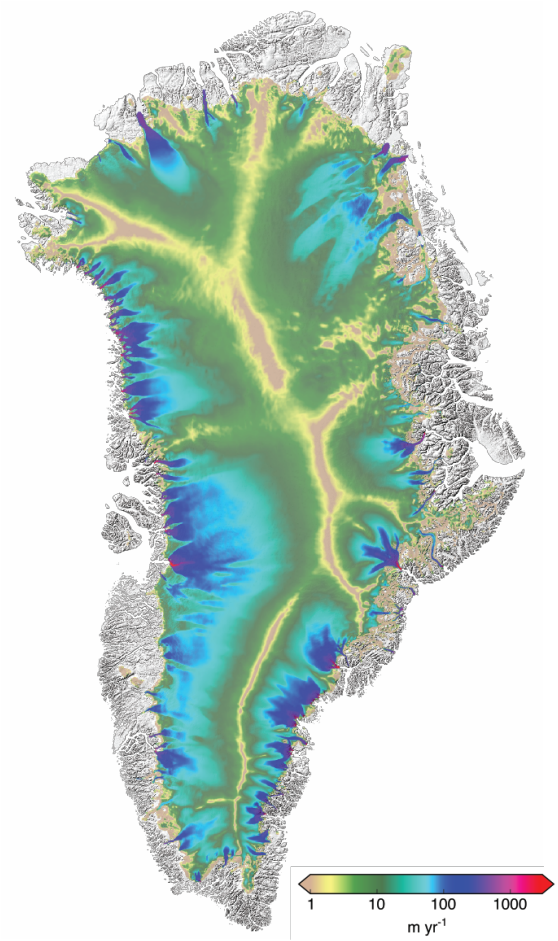
PISM
Parallel Ice Sheet Model

User's Manual



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

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`

Authorship

PISM is a joint project between developers in the ice sheet modeling group at the University of Alaska (UAF), developers at the Potsdam Institute for Climate Impact Research (PIK), and several additional developers listed here. Current (or recent) affiliation and area of contributions shown:

Torsten Albrecht (PIK)	ice shelf physics and numerics
Andy Aschwanden (UAF)	scripts, visualization, thermodynamics, SeaRISE-Greenland
Jed Brown (ANL)	source code original author, SSA numerics, PETSc underpinnings
Ed Bueler (UAF)	principal investigator, verification, earth deformation, SIA numerics, thermodynamics, documentation
Dani DellaGiustina (UAF)	regional tools and modeling
Johannes Feldman (PIK)	marine ice sheet processes
Bob Fischer (GFDL)	coupling design
Marijke Habermann (UAF)	inversion
Marianne Haseloff (UBC)	ice streams: physics and numerics
Regine Hock (UAF)	surface mass and energy balance
Constantine Khroulev (UAF)	source code primary author, input/output, software design, climate couplers, parallelization, testing, user support, most documentation, most bug fixes, regional tools, ...
Anders Levermann (PIK)	calving, ice shelf processes
Craig Lingle (UAF)	original SIA model, earth deformation
Maria Martin (PIK)	SeaRISE-Antarctica, Antarctica processes
Mattias Mengel (PIK)	marine ice sheet processes
David Maxwell (UAF)	inversion, SSA finite elements, python bindings
Ward van Pelt (IMAU)	hydrology analysis and design
Julien Seguinot (INK)	bug fixes, temperature index model
Ricarda Winkelmann (PIK)	Antarctica processes, coupling, and modeling
Florian Ziemen (UAF)	bug fixes, sliding

Email the underlined UAF developers at help@pism-docs.org.

Front page: Magnitude of horizontal surface velocity from PISM run on a horizontal grid resolution of 600 m. Visualization by QGIS.

Copyright (C) 2004–2014 The PISM Authors

This file is part of PISM. PISM is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 3 of the License, or (at your option) any later version. PISM is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with PISM; see `COPYING`*. If not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA*

Acknowledgements

NASA Modeling, Analysis, and Prediction (MAP) program grant # NNX13AM16G and NASA Cryospheric Sciences program grant # NNX13AK27G support the development of PISM from 2013 to 2017. NASA MAP grant # NNX09AJ38G supported the development of PISM from 2009 to 2013. Development from 2002 to 2008 was supported by the NASA Cryospheric Sciences program.

The Snow, Ice, and Permafrost group at the Geophysical Institute is the home for the University of Alaska PISM developers; find us in Elvey 410D. The Arctic Region Supercomputing Center has provided significant computational resources and technical help in the development of PISM.

Thanks for comments/questions from many PISM users around the world, including these not already listed as PISM authors:

> Tolly Aðalgeirsdóttir, Antje Fitzner, Nick Golledge, Tore Hattermann, Moritz Hütten, Thomas Kleiner, Leo van Kampenhout, Marianne Madsen, Malou Maris, Tim Morey, Mirena Olaizola, Christian Rodehacke, Nathan Shemonski, Sebastian Simonsen, Anne Solgaard, Ben Sperisen, Synne Høyer Svendsen, Martin Truffer, Shuting Yang, Ryan Woodard

for helpful comments and questions on PISM and this *Manual*. Dave Covey, Don Bahls, and Greg Newby have supported our hardware, software, and computations. Bob Bindschadler, Sophie Nowicki, Jesse Johnson, and others in the SeaRISE group have motivated and assisted PISM development in many ways.

Contents

1 Introduction **6**

2 Getting started **7**
2.1 A Greenland ice sheet example . . . 7
2.2 Input data . . . 7
2.3 First run . . . 8
2.4 Watching the first run . . . 10
2.5 Second run: a better ice-dynamics model . . . 11
2.6 Third run: higher resolution . . . 14
2.7 Fourth run: paleo-climate model spin-up . . . 15
2.8 Getting serious I: grid sequencing . . . 18
2.9 Getting serious II: an ice dynamics parameter study . . . 21
2.10 Handling NetCDF files . . . 26

3 Ice dynamics, the PISM view **27**
3.1 Two stress balance models: SIA and SSA . . . 27
3.2 A hierarchy of simplifying assumptions for grounded ice flow . . . 28
3.3 Evolutionary versus diagnostic modeling . . . 29
3.4 Climate inputs, and their interface with ice dynamics . . . 30

4 Initialization and bootstrapping **34**
4.1 Initialization from a saved model state . . . 34
4.2 Bootstrapping . . . 35

5 Modeling choices: Grid and time **37**
5.1 Computational box . . . 37
5.2 Spatial grid . . . 37
5.2.1 Parallel domain distribution . . . 38
5.3 Model time . . . 39
5.4 Calendars . . . 39
5.4.1 Re-starting an interrupted run using `-time_file` . . . 41
5.5 Diagnostic computations . . . 42
5.6 Disabling PISM components . . . 42
5.7 Dealing with more difficult modeling choices . . . 43

6 Modeling choices: Ice dynamics and thermodynamics **44**
6.1 Choosing the stress balance . . . 44
6.2 Ice rheology . . . 47
6.3 Surface gradient method . . . 51
6.4 Modeling conservation of energy . . . 51
6.5 Computing ice age . . . 52

7 Modeling choices: The subglacier **53**
7.1 Controlling basal strength . . . 53
7.2 Subglacial hydrology . . . 57
7.3 Earth deformation models . . . 61
7.4 Parameterization of bed roughness in the SIA . . . 62

8 Modeling choices: Marine ice sheet modeling **63**
8.1 PIK options for marine ice sheets . . . 63
8.1.1 Stress condition at calving fronts . . . 64
8.1.2 Partially-filled cells at the boundaries of ice shelves . . . 64
8.1.3 Iceberg removal . . . 65
8.1.4 Sub-grid treatment of the grounding line position . . . 65
8.2 Flotation criterion, mask, and sea level . . . 65
8.3 Calving . . . 66
8.4 Modeling melange back-pressure . . . 67

9 Practical usage **69**
9.1 Input and output . . . 69
9.1.1 PISM file I/O performance . . . 71
9.2 Saving time series of scalar diagnostic quantities . . . 71
9.3 Saving time series of spatially-varying diagnostic quantities . . . 76
9.4 Saving re-startable snapshots of the model state . . . 83
9.5 Run-time diagnostic viewers . . . 84
9.6 PISM's configuration flags and parameters, and how to change them . . . 85
9.7 Regridding . . . 87
9.8 Signals, to control a running PISM model . . . 88
9.9 Understanding adaptive time-stepping . . . 89
9.10 PETSc options for PISM users . . . 90
9.11 Utility and test scripts . . . 93
9.12 Using PISM for flow-line modeling . . . 94
9.13 Managing source code modifications . . . 94

10 Verification **96**

11 Simplified geometry experiments with PISM **102**
11.1 EISMINT II . . . 103
11.2 MISMIP . . . 104
11.3 MISMIP3d . . . 107

12 Validation case studies **111**
12.1 An SIA flow model for a table-top laboratory experiment . . . 111
12.2 An SSA flow model for the Ross Ice Shelf in Antarctica . . . 113

13 Example: A regional model of the Jakobshavn outlet glacier in Greenland **118**

References **124**

General Index **131**

PISM Command-line options **134**

1 Introduction

Welcome! All information about PISM is online at the home page

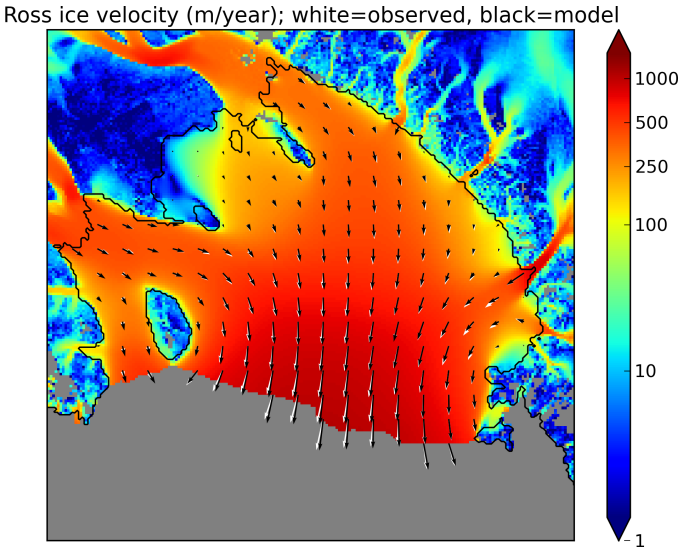
http://www.pism-docs.org

Please see the extensive lists of PISM publications and applications at that page.

This User's Manual gives examples of how to run PISM using publicly-available data for: the whole Greenland ice sheet, the Jakobshavn outlet glacier in Greenland, the Ross ice shelf in Antarctica, and a number of simplified geometry tests. It documents all the PISM options. It summarizes the continuum models used by PISM, and it illustrates how PISM's numerical approximations are verified.

See the PISM Installation Manual[1] for how to download the PISM source code and install it, along with needed libraries. The PISM Climate Forcing Manual[2] extends the User's Manual to cover additional couplings to atmosphere and ocean models and data.

Users who want to understand more deeply how PISM is designed, or who want to extend it, will need to go beyond what is described here. See the *Source Code Browser*, which is online for the latest stable version.[3] It can be generated from source code as described in the PISM Installation Manual. It gives a complete view of the class/object structure of the PISM source code.



WARNING: PISM is an ongoing research project. Ice sheet modeling requires many choices. Please don't trust the results of PISM or any other ice sheet model without a fair amount of exploration. Also, please don't expect all your questions to be answered here. Write to us with questions at help@pism-docs.org.

[1] PDF for latest stable release at http://www.pism-docs.org/wiki/lib/exe/fetch.php?media=installation.pdf.
[2] PDF for latest stable release at http://www.pism-docs.org/wiki/lib/exe/fetch.php?media=forcing.pdf.
[3] At http://www.pism-docs.org/wiki/doku.php?id=browser.

2 Getting started

This introduction is intended to be interactive and participatory, and it should work on *your personal machine* as well as on a supercomputer. Please try the commands and view the resulting files. Do the runs with your own values for the options. We can't hide the fact that PISM has lots of "control knobs," but fiddling with them will help you get going. Give it a try!

To install PISM see the *Getting PISM* tab at `www.pism-docs.org`. Or get the PISM Installation Manual (PDF) at `http://www.pism-docs.org/wiki/lib/exe/fetch.php?media=installation.pdf`. Once PISM is installed, the executable `pismr` should be available on your system's "path"; confirm this with "`which pismr`". The instructions below assume you are using a `bash` shell or one that accepts `bash` syntax. They also assume you have the PISM source code in the directory "`pism/`".

2.1 A Greenland ice sheet example

We get started with an extended example showing how to generate initial states for prognostic model experiments on the Greenland ice sheet. Ice sheet and glacier model studies often involve modeling present and past states using actions like the ones demonstrated here. Our particular choices made here are motivated by the evaluation of initialization methods in [4].

We use data assembled by the Sea-level Response to Ice Sheet Evolution (SeaRISE) assessment process [12]. SeaRISE is a community-organized assessment process providing an upper bound on ice sheet contributions to sea level in the next 100–200 years, especially for the IPCC AR5 report in 2013.

This example is a hands-on first look at PISM. It is not an in-depth tutorial, and some details of what is happening are only explained later in this Manual, which thoroughly discusses PISM options, nontrivial modeling choices, and how to preprocess input data.

The basic runs here, mostly on coarse 20 and 10 km grids, can be done on a typical workstation or laptop. PISM is, however, designed to make high resolution (e.g. 5 km to $\sim$ 500 m grids for whole-Greenland ice sheet modeling) possible by exploiting large-scale parallel processing. See [4, 38, 39], among other published high-resolution PISM examples.

2.2 Input data

The NetCDF data used to initialize SeaRISE runs is freely-available online:

`http://websrv.cs.umt.edu/isis/index.php/Present_Day_Greenland`

To download the specific file we want, namely `Greenland_5km_v1.1.nc`, and preprocess it for PISM, do:

```
$ cd pism/examples/std-greenland
$ ./preprocess.sh
```

The script `preprocess.sh` requires `wget` and also the NetCDF Operators ("NCO"; `http://nco.sourceforge.net/`). It downloads the version 1.1 of the SeaRISE "master" present-day data set, which contains ice thickness and bedrock topography from BEDMAP [10], and modeled precipitation and surface mass balance rates from RACMO [29], among other fields.

In particular, it creates three new NetCDF files which can be read by PISM. The spatially-varying fields, with adjusted metadata, go in `pism_Greenland_5km_v1.1.nc`. The other two new files contain famous time-dependent paleo-climate records from ice and seabed cores: `pism_dT.nc` has the GRIP temperature record [58] and `pism_dSL.nc` has the SPECMAP sea level record [57].

Any of these NetCDF files can be viewed with `ncview` or other NetCDF visualization tools; see Table 1 below. An application of IDV to the master data set produced Figure 1, for example. Use `ncdump -h` to see the metadata and history of the files.

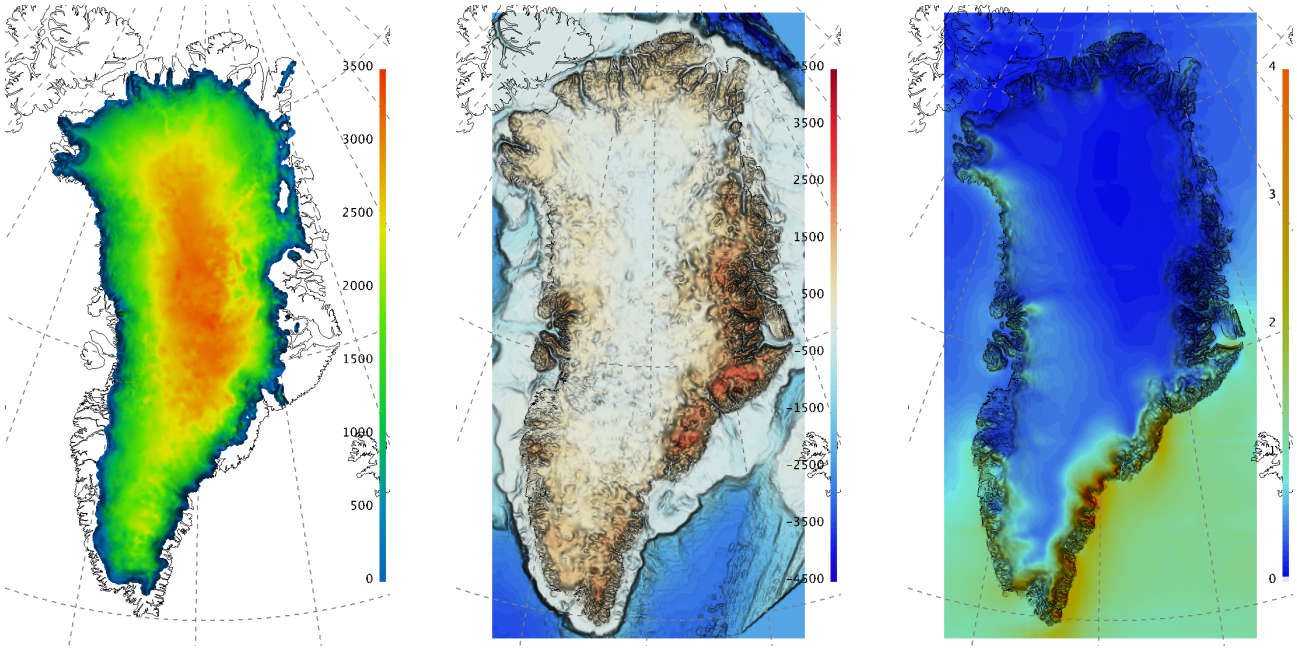


Figure 1: The input file contains present-day ice thickness (left; m), bedrock elevation (center; m), and present-day precipitation (right; m a^{-1} ice equivalent) for SeaRISE-Greenland. These are fields `thk`, `topg`, and `precipitation`, respectively, in `pism_Greenland_5km_v1.1.nc`.

2.3 First run

Like many Unix programs, PISM allows a lot of command-line options. In fact, because the variety of allowed ice sheet, shelf, and glacier configurations, and included sub-models, is so large, the list of possible command-line options covers sections 4 through 9 of this manual. In practice one often builds scripts to run PISM with the correct options, which is what we show here. The script we use is "`spinup.sh`" in the `examples/std-greenland/` subdirectory of `pism/`.

Note that initializing ice sheets, generically called "spin-up", can be done by computing approximate steady states with constant boundary data, or, in some cases, by integrating paleo-climatic and long-time-scale information, also applied at the ice sheet boundary, to build a model for the present state of the ice sheet. Both of these possibilities are illustrated in the `spinup.sh` script. The spin-up stage of using an ice sheet model may actually require more processor-hours than follow-on "experiment" or "forecast" stages.

To see what can be done with the script, read the usage message it produces:

```
$ ./spinup.sh
```

The simplest spin-up approach is to use a "constant-climate" model. We take this approach first. To see a more detailed view of the PISM command for the first run, do:

```
$ PISM_DO=echo ./spinup.sh 4 const 10000 20 sia g20km_10ka.nc
```

Setting the environment variable `PISM_DO` in this way tells `spinup.sh` just to print out the commands it is about to run, not do them. The "proposed" run looks like this:

```
mpiexec -n 4 pismr -i pism_Greenland_5km_v1.1.nc -bootstrap -Mx 76 -My 141 \
  -Mz 101 -Mbz 11 -z_spacing equal -Lz 4000 -Lbz 2000 -skip -skip_max 10 \
  -ys -10000 -ye 0 -surface given -surface_given_file pism_Greenland_5km_v1.1.nc \
  -calving ocean_kill pism_Greenland_5km_v1.1.nc -sia_e 3.0 \
  -ts_file ts_g20km_10ka.nc -ts_times -10000:yearly:0 \
  -extra_file ex_g20km_10ka.nc -extra_times -10000:100:0 \
  -extra_vars diffusivity,temppabase,tempicethk_basal,bmelt,tillwat,velsurf_mag,mask,thk,topg,usurf \
  -o g20km_10ka.nc
```

Let's briefly deconstruct this run.

At the front is "`mpiexec -n 4 pismr`". This means that the PISM executable `pismr` is run in parallel on four processes parallel standard (e.g. cores) under the Message Passing Interface ("MPI"; http://www.mcs.anl.gov/mpi/). Though we are assuming you have a workstation or laptop with at least 4 cores, this example will work with 1 to about 50 processors, with reasonably good scaling in speed. Scaling can be good with more processors if we run at higher spatial resolution [17, 28]. The executable name "`pismr`" stands for the standard "run" mode of PISM (in contrast to specialized modes described later in sections 10 and 11).

Next, the proposed run uses option `-bootstrap` to start the run by "bootstrapping." This term describes the creation, by heuristics and highly-simplified models, of the mathematical initial conditions required for a deterministic, time-dependent ice dynamics model. Then the options describe a 76×141 point grid in the horizontal, which gives 20 km grid spacing in both directions. Then there are choices about the vertical extent and resolution of the computational grid; more on those later. After that we see a description of the time-axis, with a start and end time given: "`-ys -10000 -ye 0`".

Then we get the instructions that tell PISM to read the upper surface boundary conditions (i.e. climate) from a file: "`-surface given -surface_given_file pism_Greenland_5km_v1.1.nc`". For more on these choices, see subsection 3.4, and also the PISM Climate Forcing Manual.

Then there are a couple of options related to ice dynamics. First is a minimal calving model which removes ice at the calving front location given by a thickness field in the input file ("`-calving ocean_kill`"); see subsection 8.3 for this and other calving options). Then there is a setting for enhanced ice softness ("`-sia_e 3.0`"). See subsection 6.2 for more on this enhancement parameter, which we also return to later in the current section in a parameter study.

Then there are longish options describing the fields we want as output, including scalar time series ("`-ts_file ts_g20km_10ka.nc -ts_times -10000:yearly:0`"; see section 9) and space-dependent fields ("`-extra_file ...`"; again see section 9), and finally the named output file ("`-o g20km_10ka.nc`").

Note that the modeling choices here are reasonable, but they are not the only way to do it! The user is encouraged to experiment; that is the point of a model.

Now let's actually get the run going:

```
$ ./spinup.sh 4 const 10000 20 sia g20km_10ka.nc &> out.g20km_10ka &
```

The terminating "`&`", which is optional, asks unix to run the command in the background, so we can keep working in the current shell. Because we have re-directed the text output ("`&> out.g20km_10ka`"), PISM will show what it is doing in the text file `out.g20km_10ka`. Using `less` is a good way to watch such a growing text-output file. This run should take 20 minutes or less.

2.4 Watching the first run

As soon as the run starts it creates time-dependent NetCDF files `ts_g20km_10ka.nc` and `ex_g20km_10ka.nc`. The latter file, which has spatially-dependent fields at each time, is created after the first 100 model years, a few wall clock seconds in this case. The command `-extra_file ex_g20km_10ka.nc -extra_times -10000:100:0` adds a spatially-dependent "frame" at model times -9900, -9800, ..., 0.

To look at the spatial-fields output graphically, do:

```
$ ncview ex_g20km_10ka.nc
```

We see that `ex_g20km_10ka.nc` contains growing "movies" of the fields chosen by the `-extra_vars` option. A frame of the ice thickness field `thk` is shown in Figure 2 (left).

The time-series file `ts_g20km_10ka.nc` is also growing. It contains spatially-averaged "scalar" diagnostics like the total ice volume or the ice-sheet-wide maximum velocity (variable `ivol` and `max_hor_vel`, respectively). It can be viewed

```
$ ncview ts_g20km_10ka.nc
```

The growing time series for `ivol` is shown in Figure 2 (right). Recall that our intention was to generate a minimal model of the Greenland ice sheet in approximate steady-state with a steady (constant-in-time) climate. The measurable steadiness of the `ivol` time series is a possible standard for steady state [85, for example].

At the end of the run the output file `g20km_10ka.nc` is generated. Figure 3 shows some fields from this file. In the next subsections we consider their "quality" as model results. To see a report on computational performance, we do:

```
$ ncdump -h g20km_10ka.nc |grep history
    :history = "user@machine 2013-11-23 15:57:22 AKST: PISM done.  Performance stats:
0.3435 wall clock hours, 1.3738 proc.-hours, 7274.0065 model years per proc.-hour,
PETSc MFlops = 0.03.\n",
```

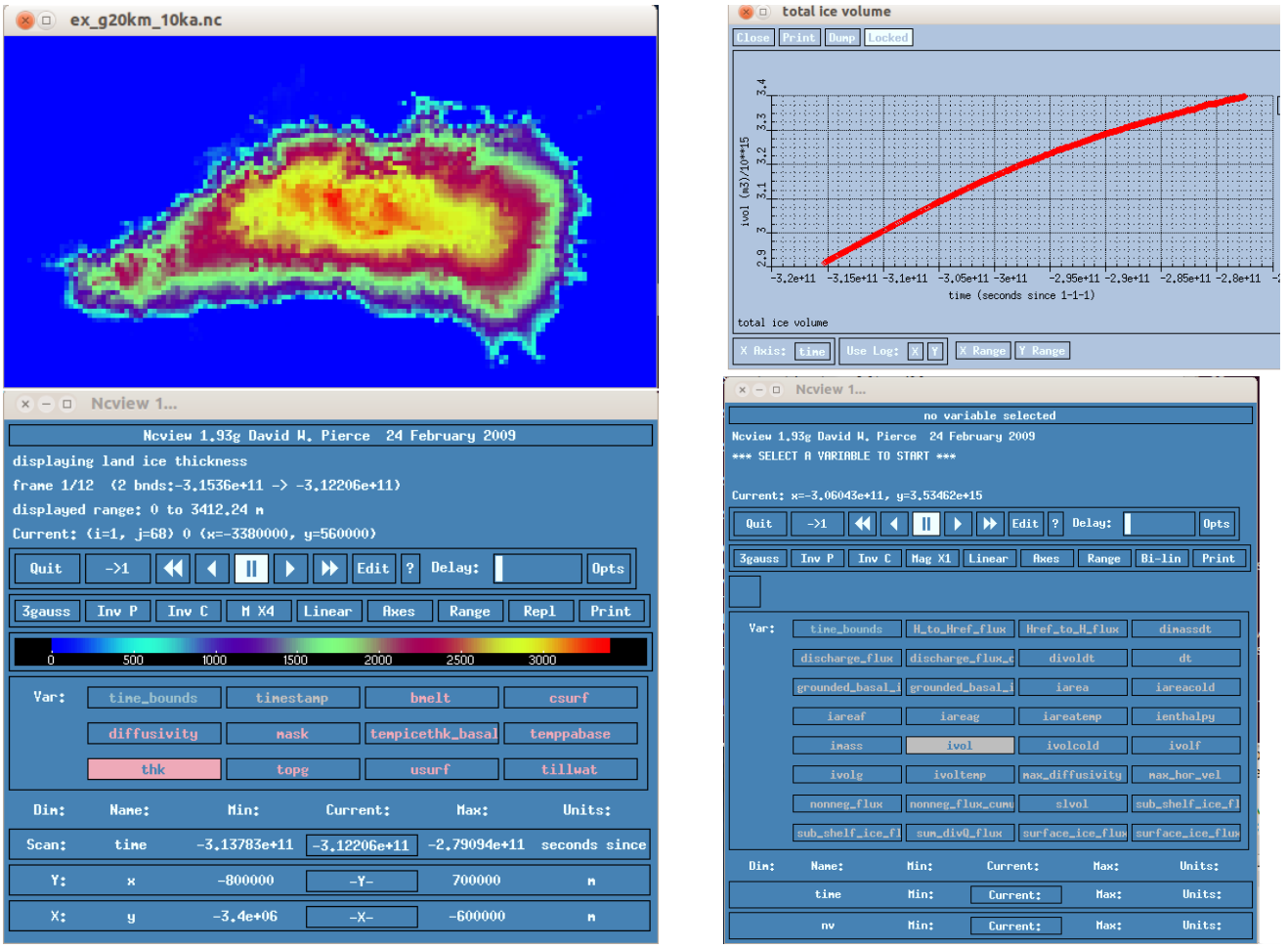


Figure 2: Two views produced by `ncview` during a PISM model run. Left: `thk`, the ice sheet thickness, a space-dependent field, from file `ex_g20km_10ka.nc`. Right: `ivol`, the total ice sheet volume time-series, from file `ts_g20km_10ka.nc`.

2.5 Second run: a better ice-dynamics model

It is widely-understood that ice sheets slide on their bases, especially when liquid water is present at the base (see [62, 71], among others). An important aspect of modeling such sliding is the inclusion of membrane or "longitudinal" stresses into the stress balance [17]. The basic stress balance in PISM which involves membrane stresses is the Shallow Shelf Approximation (SSA) [112]. The stress balance used in the previous section was, by contrast, the (thermomechanically-coupled) non-sliding, non-membrane-stress Shallow Ice Approximation (SIA) [18, 85]. The preferred ice dynamics model within PISM, that allows both sliding balanced by membrane stresses and shear flow as described by the SIA, is the SIA+SSA "hybrid" model [17, 115]. For more on stress balance theories see section 3 of this Manual.

The practical issue with models of sliding is that a distinctly-uncertain parameter space must be introduced. This especially involves parameters controlling the amount and pressure of subglacial water

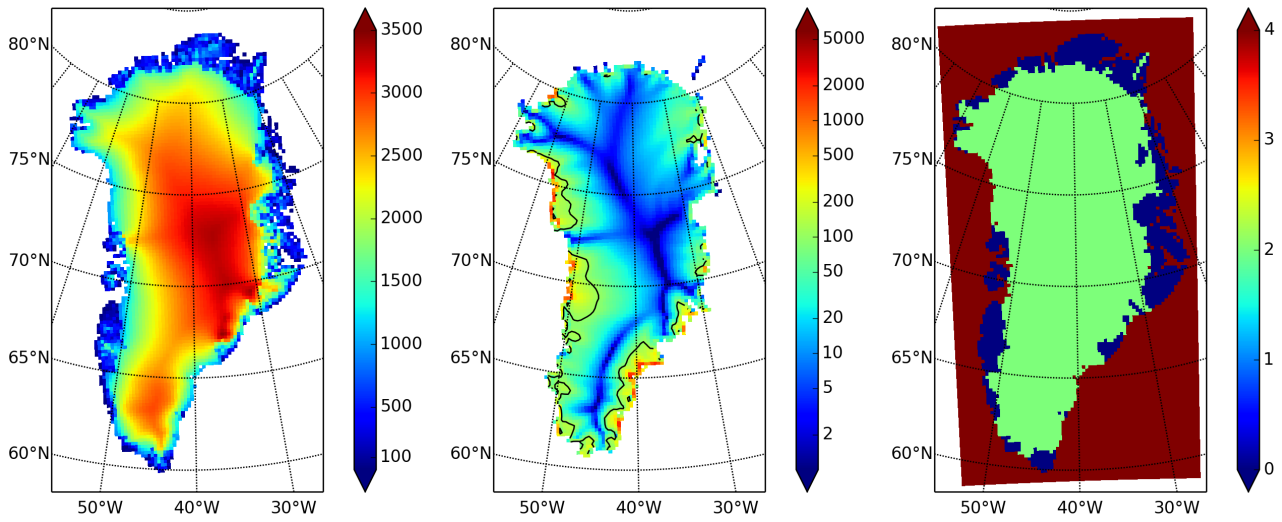


Figure 3: Fields from output file `g20km_10ka.nc`. Left: `usurf`, the ice sheet surface elevation in meters. Middle: `velsurf_mag`, the surface speed in meters/year (= m/a), including the 100 m/a contour (solid black). Right: `mask`, with 0 = ice-free land, 2 = grounded ice, 4 = ice-free ocean.

(see [4, 24, 108, 110] among other references). In this regard, PISM uses the concept of a saturated and pressurized subglacial till with a modeled distribution of yield stress [17, 99]. The yield stress arises from the PISM model of the production of subglacial water, which is itself computed through the conservation of energy model [6]. We use such models in the rest of this Getting Started section.

While the `spinup.sh` script has default sliding-related parameters, for demonstration purposes we change one parameter. We replace the default power $q = 0.25$ in the sliding law (the equation which relates both the subglacial sliding velocity and the till yield stress to the basal shear stress which appears in the SSA stress balance) by a less "plastic" and more "linear" choice $q = 0.5$. See subsection 7.1 for more on sliding laws. To see the run we propose, do

```
$ PISM_DO=echo PARAM_PPQ=0.5 ./spinup.sh 4 const 10000 20 hybrid g20km_10ka_hy.nc
```

Now remove "`PISM_DO=echo`" and redirect the text output into a file to start the run:

```
$ PARAM_PPQ=0.5 ./spinup.sh 4 const 10000 20 hybrid g20km_10ka_hy.nc &> out.g20km_10ka_hy &
```

This run should take 30 minutes or less.[4]

When this run is finished it produces `g20km_10ka_hy.nc`. As before do

```
$ ncdump -h g20km_10ka_hy.nc |grep history
```

[4]Regarding the relative speeds of the runs that produce `g20km_10ka.nc` and `g20km_10ka_hy.nc`, note that the computation of the SSA stress balance is substantially more expensive than the SIA in a per-step sense. However, the SSA stress balance in combination with the mass continuity equation causes the maximum diffusivity in the ice sheet to be substantially lower during the run. Because the maximum diffusivity controls the time-step in the PISM adaptive time-stepping scheme [18], the number of time steps is reduced in the hybrid run. To see this contrast use `ncview ts_g20km_10ka*nc` to view variables `max_diffusivity` and `dt`.

to see performance results for your machine. The number reported as "`PETSc MFlops`" from this run is about 3×10^5, much larger than the previous run, because now calls to the PETSc library are used when solving the non-local SSA stress balance in parallel.

The results of this run are shown in Figure 4. We show the basal sliding speed field `velbase_mag` in this Figure, where Figure 3 had the `mask`, but the reader can check that `velbase_mag`=0 in the nonsliding SIA-only result `g20km_10ka.nc`.

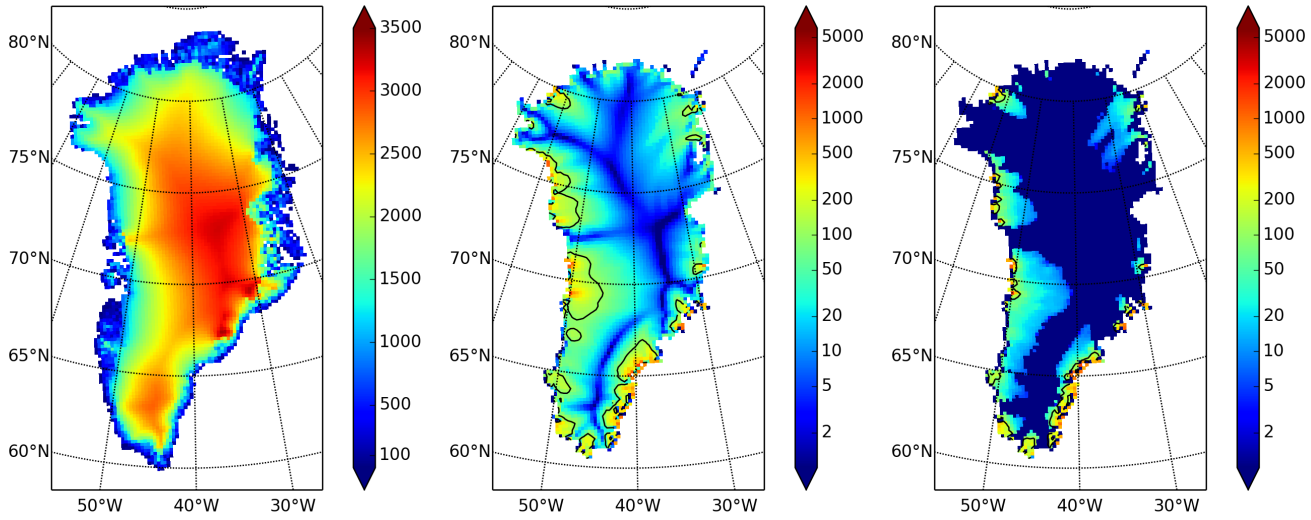


Figure 4: Fields from output file `g20km_10ka_hy.nc`. Left: `usurf`, the ice sheet surface elevation in meters. Middle: `velsurf_mag`, the surface speed in m/a, including the 100 m/a contour (solid black). Right: the sliding speed `velbase_mag`, shown the same way as `velsurf_mag`.

The hybrid model includes sliding, and it is important to evaluate that aspect of the output. However, though it is critical to the response of the ice to changes in climate, basal sliding velocity is essentially unobservable in real ice sheets. On the other hand, because of relatively-recent advances in radar and image technology and processing [59], the surface velocity of an ice sheet is an observable.

So, how good is our model result `velsurf_mag`? Figure 5 compares the radar-observed `surfvelmag` field in the downloaded SeaRISE-Greenland data file `Greenland_5km_v1.1.nc` with the just-computed PISM result. The reader might agree with these broad qualitative judgements:

- the model results and the observed surface velocity look similar, and
- slow near-divide flow is generally in the right areas and of generally the right magnitude, but
- the observed Northeast Greenland ice stream is more distinct than in the model.

We can compare these PISM results to other observed-vs-model comparisons of surface velocity maps, for example Figure 1 in [90] and Figure 8 in [65]. Only ice-sheet-wide parameters and models were used here in PISM, that is, each location in the ice sheet was modeled by the same physics. By comparison, those published comparisons involved tuning a large number of subglacial parameters to values which

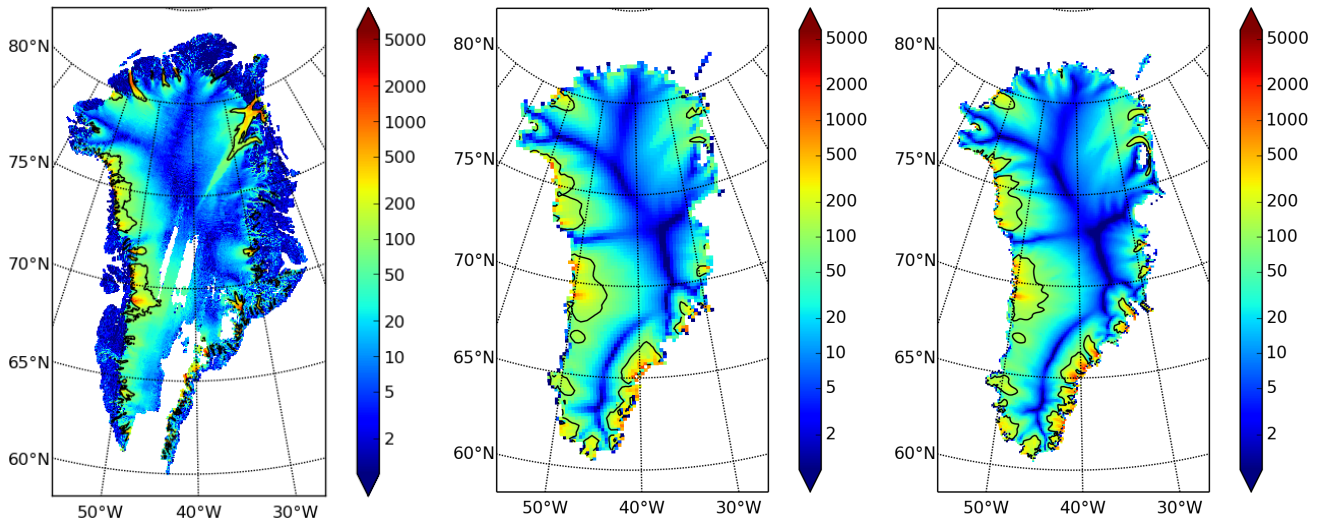


Figure 5: Comparing observed and modeled surface speed. All figures have a common scale (m/a), with 100 m/a contour shown (solid black). Left: `surfvelmag`, the observed values from SeaRISE data file `Greenland_5km_v1.1.nc`. Middle: `velsurf_mag` from `g20km_10ka_hy.nc`. Right: `velsurf_mag` from `g10km_10ka_hy.nc`.

would yield close match to observations of the surface velocity. Such tuning techniques, called "inversion" or "assimilation" of the surface velocity data, are also possible in PISM,[5] but the advantage of having few parameters in a model is well-known: the results reflect the underlying model not the flexibility of many parameters.

We have only tried two of the many models possible in PISM, and we are free to identify and adjust important parameters. The first parameter change we consider, in the next subsection, is one of the most important: grid resolution.

2.6 Third run: higher resolution

Now we change one key parameter, the grid resolution. Model results differ even when the only change is the resolution. Using higher resolution "picks up" more detail in the bed elevation and climate data.

If you can let it run overnight, do

```
$ PARAM_PPQ=0.5 ./spinup.sh 4 const 10000 10 hybrid g10km_10ka_hy.nc &> out.g10km_10ka_hy &
```

This run might take 4 to 6 hours. However, supposing you have a larger parallel computer, you can change "`mpiexec -n 4`" to "`mpiexec -n N`" where `N` is a substantially larger number, up to 100 or so with an expectation of reasonable scaling on this grid [17, 28].

Some fields from the result `g10km_10ka_hy.nc` are shown in Figure 6. Figure 5 also compares observed velocity to the model results from 20 km and 10 km grids. As a different comparison, Figure 7 shows

[5] See [111] (inversion of DEMs for basal topography) and [44] (inversion surface velocities for basal shear stress) for PISM-based inversion methods and analysis.

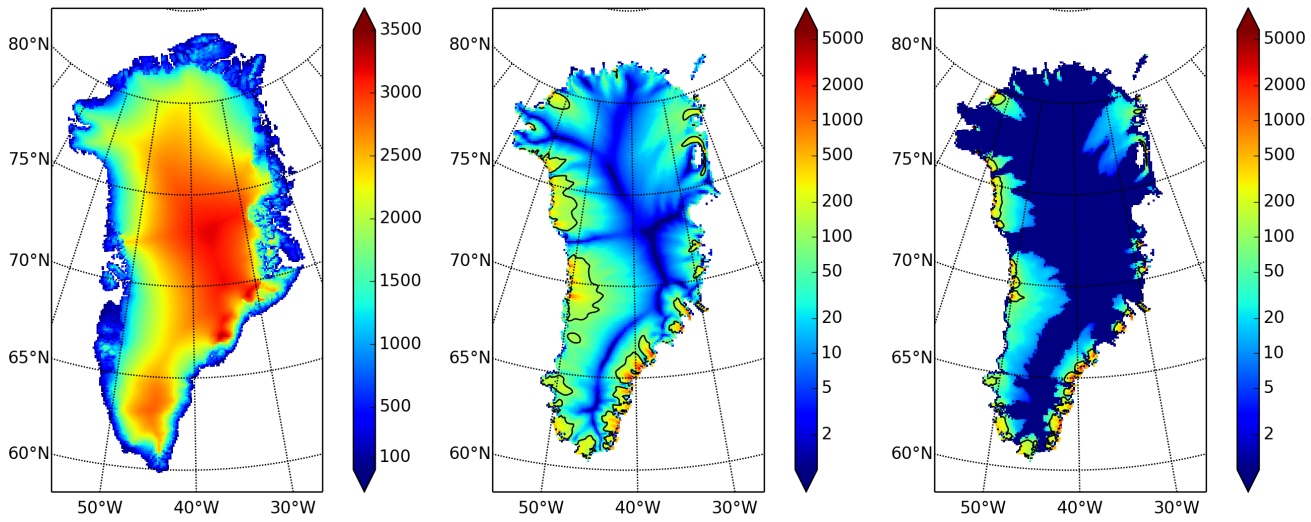


Figure 6: Fields from output file `g10km_10ka_hy.nc`. Compare Figure 4, which only differs by resolution. Left: `usurf` in meters. Middle: `velsurf_mag` in m/a. Right: `velbase_mag` in m/a.

ice volume time series `ivol` for 20 km and 10 km runs done here. We see that this result depends on resolution, in particular because higher resolution grids allow the model to better resolve the flux through topographically-controlled outlet glaciers (compare [88]). However, because the total ice sheet volume is a highly-averaged quantity, the `ivol` difference from 20 km and 10 km resolution runs is only about one part in 60 (about 1.5%) at the final time. By contrast, as is seen in the near-margin ice in various locations shown in Figure 5, the ice velocity at a particular location may change by 100% when the resolution changes from 20 km to 10 km.

Roughly speaking, the reader should only consider trusting those model results which are demonstrated to be robust across a range of model parameters, and, in particular, which are shown to be relatively-stable among relatively-high resolution results for a particular case. Using a supercomputer is justified merely to confirm that lower-resolution runs were already "getting" a given feature or result.

2.7 Fourth run: paleo-climate model spin-up

A this point we have barely mentioned one of the most important players in an ice sheet model: the surface mass balance (SMB) model. Specifically, an SMB model combines precipitation (e.g. [8] for present-day Greenland) and a model for melt. Melt models are always based on some approximation of the energy available at the ice surface [49]. Previous runs in this section used a "constant-climate" assumption, which specifically meant using the modeled present-day SMB rates from the regional climate model RACMO [29], as contained in the SeaRISE-Greenland data set `Greenland_5km_v1.1.nc`.

While a physical model of ice dynamics only describes the movement of the ice, the SMB (and the sub-shelf melt rate) are key inputs which directly determine changes in the boundary geometry. Boundary geometry changes then feedback to determine the stresses seen by the stress balance and thus the motion.

There are other methods for producing SMB than using present-day modeled values. We now try such

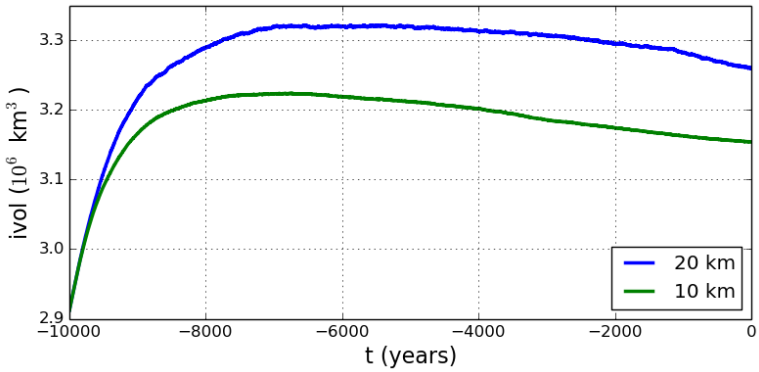


Figure 7: Time series of modeled ice sheet volume `ivol` on 20km and 10km grids. The present-day ice sheet has volume about $2.9 \times 10^6\,\text{km}^3$ [10], the initial value seen in both runs.

a method, a "paleo-climate spin-up" for our Greenland ice sheet model. Of course, direct measurements of prior climates in Greenland are not available as data! There are, however, estimates of past surface temperatures at the locations of ice cores [58, for GRIP], along with estimates of past global sea level [57] which can be used to determine where the flotation criterion is applied—this is how PISM's `mask` variable is determined. Also, models have been constructed for how precipitation differs from the present-day values [54]. For demonstration purposes, these are all used in the next run. The relevant options are further documented in PISM's Climate Forcing Manual.

As noted, one must compute melt in order to compute SMB. Here this is done using a temperature-index, "positive degree-day" (PDD) model [49]. Such a PDD model has parameters for how much snow and/or ice is melted when surface temperatures spend time near or above zero degrees. Again, see the PISM Climate Forcing Manual for relevant options.

To summarize the paleo-climate model applied here, temperature offsets from the GRIP core record affect the snow energy balance, and thus the rates of melting and runoff calculated by the PDD model. In warm periods there is more marginal ablation, but precipitation may also increase (according to a temperature-offset model [54]). Additionally sea level undergoes changes in time and this affects which ice is floating. Finally we add an earth deformation model, which responds to changes in ice load by changing the bedrock elevation [19].

To see how all this translates into PISM options, do

```
$ PISM_DO=echo PARAM_PPQ=0.5 REGRIDFILE=g20km_10ka_hy.nc \
  ./spinup.sh 4 paleo 25000 20 hybrid g20km_25ka_paleo.nc
```

You will see an impressively-long command, which you can compare to the one on page 9. There are several key changes. First, we do not start from scratch but instead from a previously computed near-equilibrium result:

```
  -regrid_file g20km_10ka_hy.nc -regrid_vars litho_temp,thk,enthalpy,tillwat,bmelt
```

For more on regridding see subsection 9.7. Then we turn on the earth deformation model with option `-bed_def lc`; see subsection 7.3. After that the atmosphere and surface (PDD) models are turned on

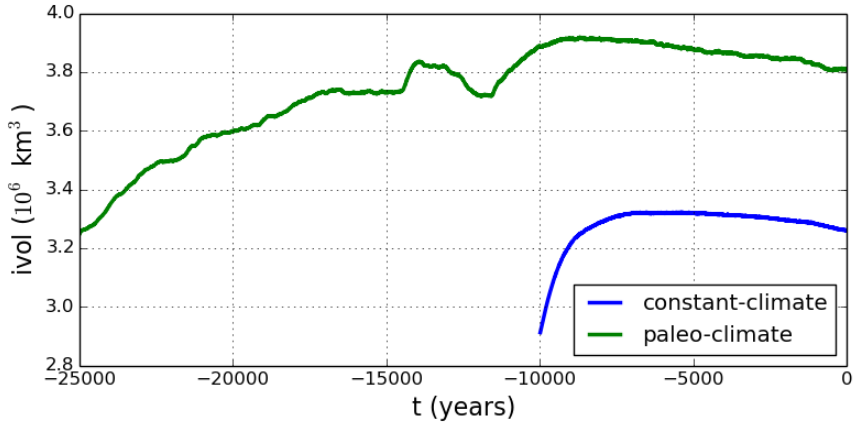


Figure 8: Time series of modeled ice sheet volume `ivol` from constant-climate (blue; `ts_g20km_10ka_hy.nc`) and paleo-climate (red; `ts_g20km_25ka_paleo.nc`) spinup runs. Note that the paleo-climate run started with the ice geometry at the end of the constant-climate run.

and the files they need are identified:

```
-atmosphere searise_greenland,delta_T,paleo_precip -surface pdd \
-atmosphere_paleo_precip_file pism_dT.nc -atmosphere_delta_T_file pism_dT.nc
```

Then the ocean model, which provides both a subshelf melt rate and a time-dependent sealevel to the ice dynamics core, is turned on with `-ocean constant,delta_SL` and the file it needs is identified with `-ocean_delta_SL_file pism_dSL.nc`. For all of these "forcing" options, see the PISM Climate Forcing Manual. The remainder of the options are similar or identical to the run that created `g20km_10ka_hy.nc`.

To actually start the run, which we rather arbitrarily start at year -25000, essentially at the LGM, do:

```
$ PARAM_PPQ=0.5 REGRIDFILE=g20km_10ka_hy.nc \
  ./spinup.sh 4 paleo 25000 20 hybrid g20km_25ka_paleo.nc &> out.g20km_25ka_paleo &
```

This run should only take one or two hours, noting it is at a coarse 20 km resolution.

The fields `usurf`, `velsurf_mag`, and `velbase_mag` from file `g20km_25ka_paleo.nc` are sufficiently similar to those shown in Figure 4 that they are not shown here. Close inspection reveals differences, but of course these runs only differ in the applied climate and run duration and not in resolution or ice dynamics parameters.

To see the difference between runs more clearly, Figure 8 compares the time-series variable `ivol`. We see the effect of option `-regrid_file g20km_10ka_hy.nc -regrid_vars ...,thk,...`, which implies that the paleo-climate run starts with the ice geometry from the end of the constant-climate run.

Another time-series comparison, of the variable `ivoltemp`, the total volume of temperate (at 0°C) ice, appears in Figure 9. The paleo-climate run shows the cold period from ≈ -25 ka to ≈ -12 ka. Both constant-climate and paleo-climate runs then come into rough equilibrium in the holocene. The bootstrapping artifact, seen at the start of the constant-climate run, which disappears in less than 1000 years, is avoided in the paleo-climate run by starting with the constant-climate end-state. The reader is

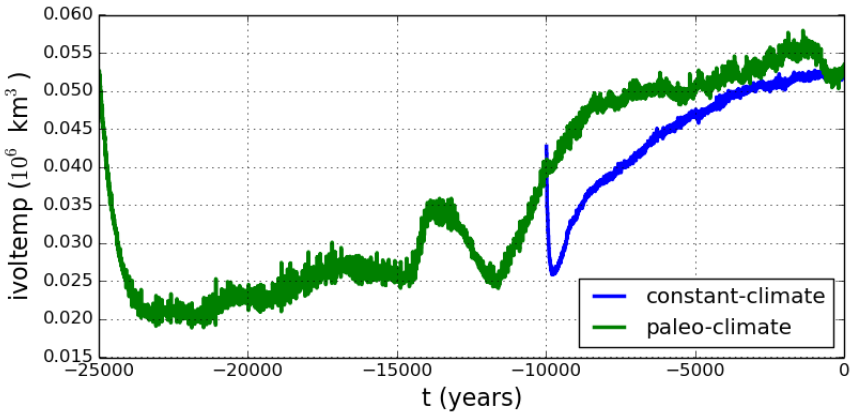


Figure 9: Time series of temperate ice volume `ivoltemp` from constant-climate (blue; `ts_g20km_10ka_hy.nc`) and paleo-climate (red; `ts_g20km_25ka_paleo.nc`) spinup runs. The cold of the last ice age affects the fraction of temperate ice. Note different volume scale compared to that in Figure 8; only about 1% of ice is temperate (by volume).

encouraged to examine the diagnostic files `ts_g20km_25ka_paleo.nc` and `ex_g20km_25ka_paleo.nc` to find more evidence of the (modeled) climate impact on the ice dynamics.

2.8 Getting serious I: grid sequencing

The previous sections were not very ambitious. We were just getting started! Now we demonstrate a serious PISM capability, the ability to change, specifically to *refine*, the grid resolution at runtime.

One can of course do the longest model runs using a coarse grid, like the 20 km grid used first. It is, however, only possible to pick up detail from high quality data, for instance bed elevation and/or high-resolution climate data, using high grid resolution.

A 20 or 10 km grid is inadequate for resolving the flow of the ice sheet through the kind of fjord-like, few-kilometer-wide topographical confinement which occurs, for example, at Jakobshavn Isbrae in the west Greenland ice sheet [64], an important outlet glacier which both flows fast and drains a large fraction of the ice sheet. One possibility is to set up an even higher-resolution PISM regional model covering only one outlet glacier, but this requires decisions about coupling to the whole ice sheet flow. (See section 13.) But here we will work on high resolution for the whole ice sheet, and thus all outlet glaciers.

Consider the following command; compare it to the one on page 9:

```
mpiexec -n 4 pismr -i pism_Greenland_5km_v1.1.nc -bootstrap -Mx 301 -My 561 \
  -Mz 201 -Mbz 21 -z_spacing equal -Lz 4000 -Lbz 2000 -ys -200 -ye 0 \
  -regrid_file g20km_10ka_hy.nc -regrid_vars litho_temp,thk,enthalpy,tillwat,bmelt ...
```

Instead of a 20 km grid in the horizontal (`-Mx 76 -My 141`) we ask for a 5 km grid (`-Mx 301 -My 561`). Instead of vertical grid resolution of 40 m (`-Mz 101 -z_spacing equal -Lz 4000`) we ask for a vertical resolution of 20 m (`-Mz 201 -z_spacing equal -Lz 4000`).[6] Most significantly, however, we say

[6] See subsections 4.2, 5.1, and 5.2 for more about determining the computation domain and grid at bootstrapping.

`-regrid_file g20km_10ka_hy.nc` to regrid—specifically, to bilinearly-interpolate—fields from a model result computed on the coarser 20 km grid. The regridded fields (`-regrid_vars litho_temp,...`) are the evolving mass and energy state variables which are already approximately at equilibrium on the coarse grid. Because we are bootstrapping (i.e. using the `-bootstrap` option), the other variables, especially the bedrock topography `topg` and the climate data, are brought in to PISM at "full" resolution, that is, on the original 5 km grid in the data file `pism_Greenland_5km_v1.1.nc`.

This technique could be called "grid sequencing".[7] The result of the above command will be to compute the near-equilibrium result on the fine 5 km grid, taking advantage of the coarse-gridded computation of approximate equilibrium, and despite a run of only 200 model years (`-ys -200 -ye 0`). How close to equilibrium we get depends on both durations, i.e. on both the coarse and fine grid run durations, but certainly the computational effort is reduced by doing a short run on the fine grid. Note that in the previous subsection we also used regridding. In that application, however, `-regrid_file` only "brings in" fields from a run on the same resolution.

Generally the fine grid run duration in grid sequencing should be at least $t = \Delta x/v_{\text{min}}$ where Δx is the fine grid resolution and v_{min} is the lowest ice flow speed that we expect to be relevant to our modeling purposes. That is, the duration should be such that slow ice at least has a chance to cross one grid cell. In this case, if $\Delta x = 5$ km and $v_{\text{min}} = 25$ m/a then we get $t = 200$ a. Though we use this as the duration, it is a bit short, and the reader might compare $t = 500$ results (i.e. using $v_{\text{min}} = 10$ m/a).

Actually we will demonstrate how to go from 20 km to 5 km in two steps, 20 km $\to$ 10 km $\to$ 5 km, with durations of 10 ka, 2 ka, and 200 a, respectively. The 20 km coarse grid run is already done; the result is in `g20km_10ka_hy.nc`. So we run the following script which is `gridseq.sh` in `examples/std-greenland/`. It calls `spinup.sh` to collect all the right PISM options:

```
#!/bin/bash
NN=4
export PARAM_PPQ=0.5
export REGRIDFILE=g20km_10ka_hy.nc
export EXSTEP=100
./spinup.sh $NN const 2000  10 hybrid g10km_gridseq.nc
export REGRIDFILE=g10km_gridseq.nc
export EXSTEP=10
./spinup.sh $NN const 200    5 hybrid  g5km_gridseq.nc
```

Environment variable `EXSTEP` specifies the time in years between writing the spatially-dependent, and large-file-size-generating, frames for the `-extra_file ...` diagnostic output.

Before you run the above script, however, an important

WARNING: the 5 km run requires 8 Gb of memory at minimum!

If you try it without at least 8 Gb of memory then your machine will "bog down" and start using the hard disk for swap space! The run will not complete and your hard disk will get a lot of wear! (If you have less than 8 Gb memory, comment out the last three lines of the above script—e.g. using the "`#`" character at the beginning of the line—so that you only do the 20 km $\to$ 10 km refinement.)

[7] It is not quite "multigrid." That would both involve refinement and coarsening stages in computing the fine grid solution.

Run the script like this:

```
$ ./gridseq.sh &> out.gridseq &
```

The 10 km run takes under two wall-clock hours (8 processor-hours) and the 5 km run takes about 6 wall-clock hours (24 processor-hours).

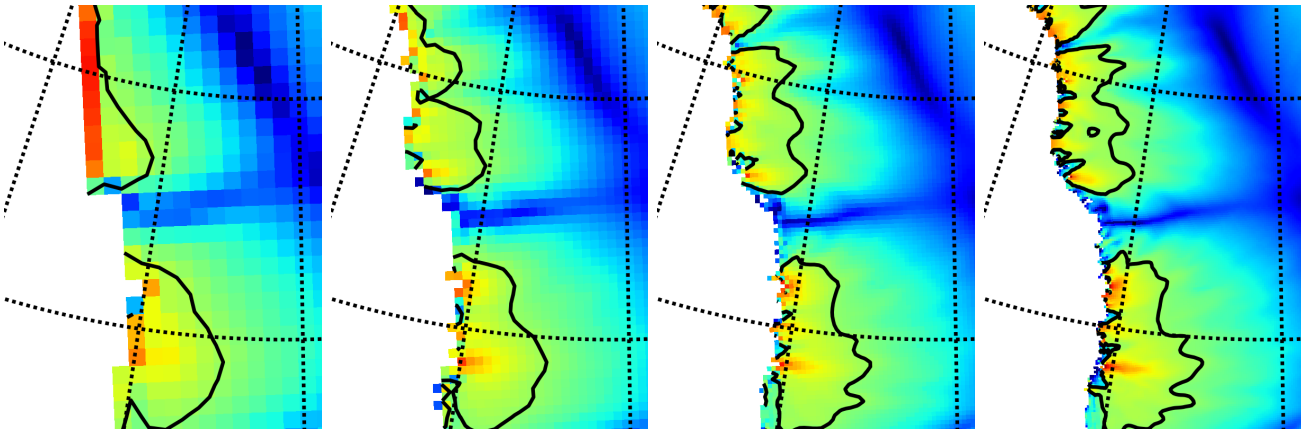


Figure 10: Detail of field `velsurf_mag` showing the central western coast of Greenland, including Jakobshavn Isbrae (lowest major flow), from runs of resolution 40, 20, 10, 5 km (left-to-right). Color scheme and scale, including 100 m/a contour (solid black), are all identical to `velsurf_mag` Figures 4, 5, and 6.

Figure 10, showing only a detail of the western coast of Greenland, with several outlet glaciers visible, suggests what is accomplished: the high resolution runs have separated outlet glacier flows, as they are in fact. Note that all of these results were generated in a few wall clock hours on a laptop! The surface speed `velsurf_mag` from files `g10km_gridseq.nc` and `g5km_gridseq.nc` is shown (two right-most subfigures). In the two left-hand subfigures we show the same field from NetCDF files `g40km_10ka_hy.nc` and `g20km_10ka_hy.nc`; the former is an added 40 km result using an obvious modification of the run in section 2.5.

Figure 11, which shows time series of ice volume, also shows the cost of high resolution, however. The short 200 a run on the 5 km grid took about 3 wall-clock hours compared to the 10 minutes taken by the 10 ka run on a 20 km grid. The fact that the time series for ice volume on 10 km and 5 km grids are not very "steady" also suggests that these runs should actually be longer.

In this vein, if you have an available supercomputer then a good exercise is to extend our grid sequencing example to 3 km or 2 km resolutions [4]; these grids are already supported in the script `spinup.sh`. Note that the vertical grid also generally gets refined as the horizontal grid is refined.

Going to a 1km grid is possible, but you will start to see the limitations of distributed file systems in writing the enormous NetCDF files in question [28]. Notice that a factor-of-five refinement in all three dimensions, e.g. from 5 km to 1 km in the horizontal, and from 20 m to 4 m in the vertical, generates an output NetCDF file which is 125 times larger. Since the already-generated 5 km result `g5km_gridseq.nc` is over 0.5 Gb, the result is a very large file at 1 km.

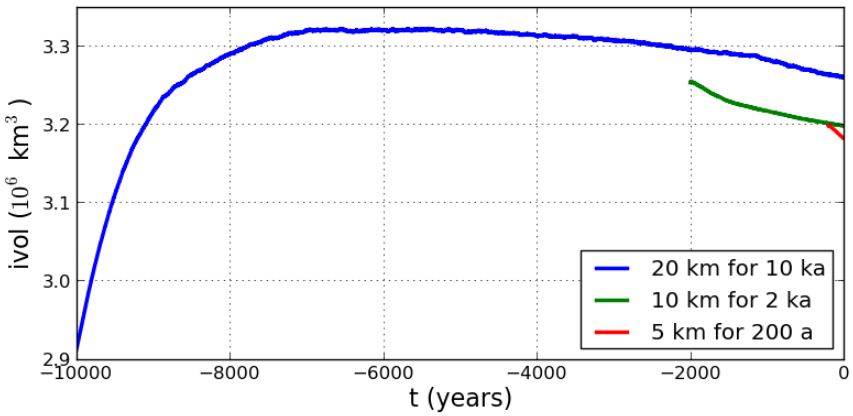


Figure 11: Time series of ice volume `ivol` from the three runs in our grid sequencing example: 20 km for 10 ka = `ts_g20km_10ka_hy.nc`, 10 km for 2 ka = `ts_g10km_gridseq.nc`, and 5 km for 200 a = `ts_g5km_gridseq.nc`.

On the other hand, on fine grids we observe that *memory* parallelism, i.e. spreading the stored model state over the separated memory of many nodes of supercomputers, is as important as the usual *computation* (CPU) parallelism.

This subsection has emphasized the "P" in PISM, the nontrivial parallelism in which the solution of the conservation equations, especially the stress balance equations, is distributed across processors. An easier and more common mode of parallelism is to distribute distinct model runs, each with different parameter values, among the processors. For scientific purposes, such parameter studies, whether parallel or not, are at least as valuable as individual high-resolution runs.

2.9 Getting serious II: an ice dynamics parameter study

The readers of this manual should not assume the PISM authors know all the correct parameters for describing ice flow. While PISM must have *default* values of all parameters, to help users get started,[8] it has more than two hundred user-configurable parameters. The goal in this manual is to help the reader adjust them to their desired values. While "correct" values may never be known, or may not exist, examining the behavior of the model as it depends on parameters is both a nontrivial and an essential task.

For some parameters used by PISM, changing their values within their ranges of experimental uncertainty is unlikely to affect model results in any important manner (e.g. `sea_water_density`). For others, however, for instance for the exponent in the basal sliding law, changing the value is highly-significant to model results, as we'll see in this subsection. This is also a parameter which is very uncertain given current glaciological understanding [26].

To illustrate a parameter study in this Manual we restrict consideration to just two important parameters for ice dynamics,

[8] They are stored in human-readable form in the file `src/pism_config.cdl`.

- q = `pseudo_plastic_q`: exponent used in the sliding law which relates basal sliding velocity to basal shear stress in the SSA stress balance; see subsection 7.1 for more on this parameter, and
- e = `sia_enhancement_factor`: values larger than one give flow "enhancement" by making the ice deform more easily in shear than is determined by the standard flow law [69, 79]; applied only in the SIA stress balance; see subsection 6.2 for more on this parameter.

By varying these parameters over full intervals of values, say $0.1 \le q \le 1.0$ and $1 \le e \le 6$, we could explore a two-dimensional parameter space. But of course each (q, e) pair needs a full computation, so we can only sample this two-dimensional space. Furthermore we must specify a concrete run for each parameter pair. In this case we choose to run for 1000 model years, in every case initializing from the stored state `g10km_gridseq.nc` generated in the previous subsection 2.8.

The next script, which is `param.sh` in `examples/std-greenland/`, gets values $q \in \{0.1, 0.5, 1.0\}$ and $e \in \{1, 3, 6\}$ in a double `for`-loop. It generates a run-script for each (q, e) pair. For each parameter setting it calls `spinup.sh`, with the environment variable `PISM_DO=echo` so that `spinup.sh` simply outputs the run command. This run command is then redirected into an appropriately-named `.sh` script file:

```
#!/bin/bash
NN=4
DUR=1000
START=g10km_gridseq.nc
for PPQ in 0.1 0.5 1.0 ; do
  for SIAE in 1 3 6 ; do
     PISM_DO=echo REGRIDFILE=$START PARAM_PPQ=$PPQ PARAM_SIAE=$SIAE \
       ./spinup.sh $NN const $DUR 10 hybrid p10km_${PPQ}_${SIAE}.nc \
       &> p10km_${PPQ}_${SIAE}.sh
  done
done
```

Notice that, because the stored state `g10km_gridseq.nc` used $q = 0.5$ and $e = 3$, one of these runs simply continues with no change in the physics.

To set up and run the parameter study, without making a mess from all the generated files, do:

```
$ cd examples/std-greenland/           # g10km_gridseq.nc should be in this directory
$ mkdir paramstudy
$ cd paramstudy
$ ln -s ../g10km_gridseq.nc            # these four lines make links to ...
$ ln -s ../pism_Greenland_5km_v1.1.nc  #
$ ln -s ../spinup.sh                   #
$ ln -s ../param.sh                    # ... existing files in examples/std-greenland/
$ ./param.sh
```

The result of the last command is to generate nine run scripts,

```
p10km_0.1_1.sh  p10km_0.1_3.sh  p10km_0.1_6.sh
p10km_0.5_1.sh  p10km_0.5_3.sh  p10km_0.5_6.sh
p10km_1.0_1.sh  p10km_1.0_3.sh  p10km_1.0_6.sh
```

The reader should inspect a few of these scripts. They are all very similar, of course, but, for instance, the `p10km_0.1_1.sh` script uses options `-pseudo_plastic_q 0.1` and `-sia_e 1`.

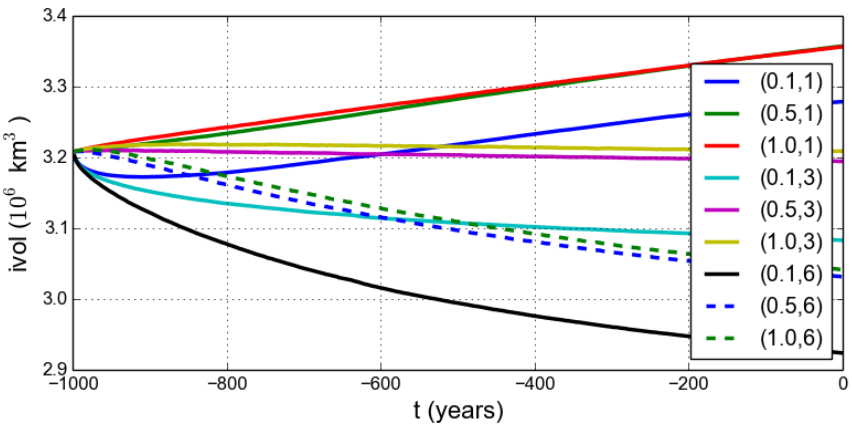


Figure 12: Time series of ice volume `ivol` from nine runs in our parameter study example, with parameter choices (q, e) given.

We have not yet run PISM, but only asked one script to create nine others. We now have the option of running them sequentially or in parallel. Each script itself does a parallel run, over the `NN=4` processes specified by `param.sh` when generating the run scripts. If you have $4 \times 9 = 36$ cores available then you can do the runs fully in parallel (this is `runparallel.sh` in `examples/std-greenland/`):

```
#!/bin/bash
for scriptname in $(ls p10km*sh) ; do
  echo ; echo "starting ${scriptname} ..."
  bash $scriptname &> out.$scriptname &  # start immediately in background
done
```

Otherwise you should do them in sequence (this is `runsequential.sh` in `examples/std-greenland/`):

```
#!/bin/bash
for scriptname in $(ls p10km*sh) ; do
  echo ; echo "starting ${scriptname} ..."
  bash $scriptname                       # will wait for completion
done
```

On the same old 2012-era 4 core laptop, `runsequential.sh` took a total of just under 7 hours to complete the whole parameter study. The runs with $q = 0.1$ (the more "plastic" end of the basal sliding spectrum) took up to four times longer than the $q = 0.5$ and $q = 1.0$ runs. Roughly speaking, values of q which are close to zero imply a subglacial till model with a true yield stress, and the result is that even small changes in overall ice sheet state (geometry, energy, ...) will cause *some* location to exceed its yield stress and suddenly change flow regime. This will shorten the time steps. By contrast, the e value is much less significant in determining run times.

On a supercomputer, the `runparallel.sh` script generally should be modified to submit jobs to the scheduler. See example scripts `advanced/paramspawn.sh` and `advanced/paramsubmit.sh` for a parameter study that does this. (But see your system administrator if you don't know what a "job scheduler" is!) Of course, if you have a supercomputer then you can redo this parameter study on a 5 km grid.

Results from these runs are seen in Figures 12 and 13. In the former we see that the $(0.5, 3)$ run simply continues the previous initialization run. In some other graphs we see abrupt initial changes, caused by abrupt parameter change, e.g. when the basal sliding becomes much more plastic ($q = 0.1$). In all cases with $e = 1$ the flow slows and the sheet grows in volume as discharge decreases, while in all cases with $e = 6$ the flow accelerates and the sheet shrinks in volume as discharge increases.

In Figure 13 we can compare the surface speed model results to observations. Roughly speaking, the ice softness parameter e has effects seen most-clearly by comparing the interior of the ice sheet; scan left-to-right for the $e = 1, 3, 6$ subfigures. The basal sliding exponent q has effects seen most-clearly by comparing flow along the very steep margin, especially in the southern half of the ice sheet; scan top-to-bottom for $q = 0.1, 0.5, 1.0$, going from nearly-plastic at top to linear at bottom.

From such figures we can make an informal assessment and comparison of the results, but objective assessment is important. Example objective functionals include: *(i)* compute the integral of the square (or other power) of the difference between the model and observed surface velocity [4], or *(ii)* compute the model-observed differences between the histogram of the number of cells with a given surface speed [22]. Note that these functionals are measuring the effects of changing a small number of parameters, namely two parameters in the current study. So-called "inversion" might use the same objective functionals but with a much larger parameter space. Inversion is therefore capable of achieving much smaller objective measures [44, 65, 90], though at the cost of less understanding, perhaps, of the meaning of the optimal parameter values.

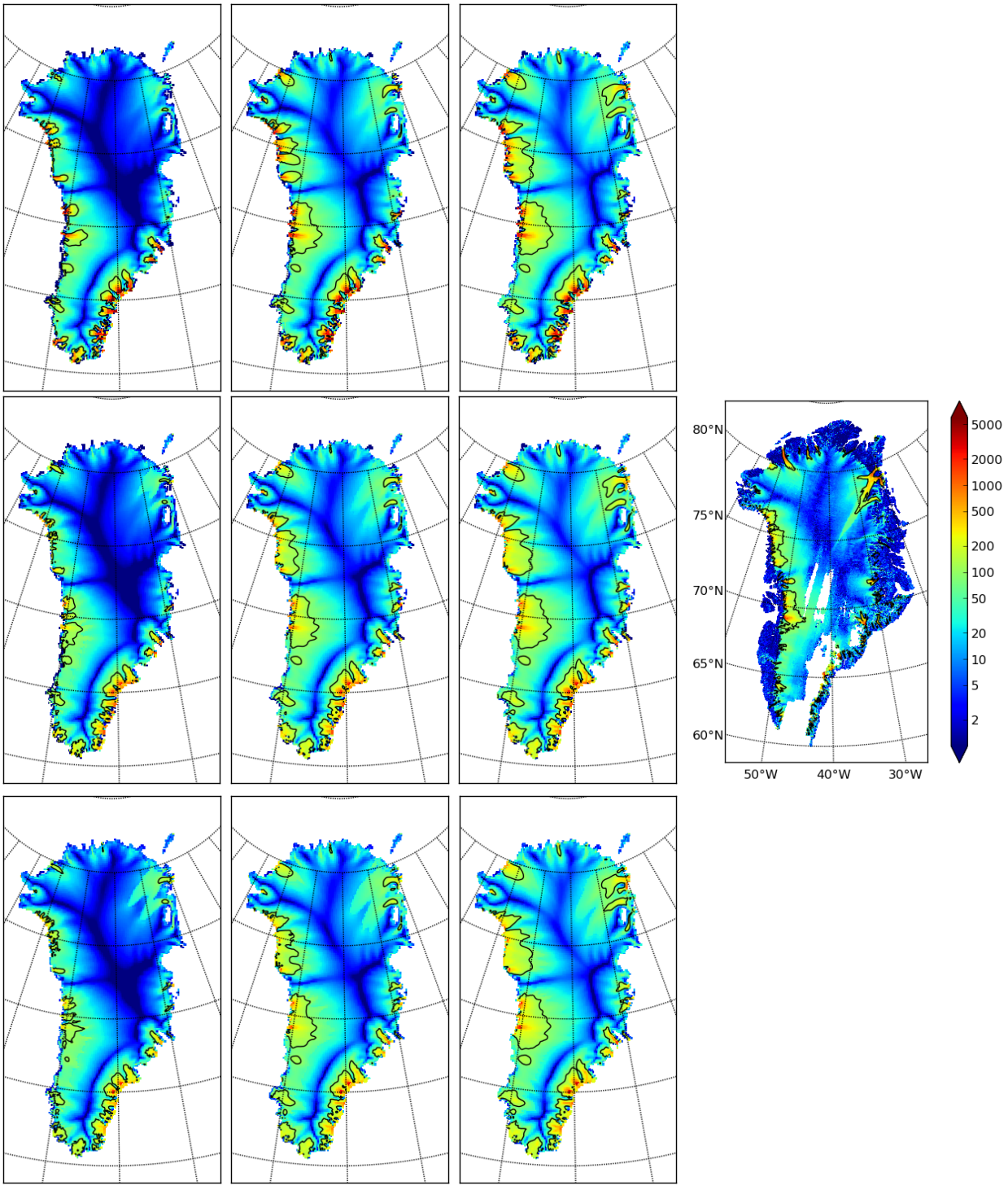


Figure 13: Surface speed `velsurf_mag` from a 10 km grid parameter study. Right-most subfigure is observed data from `Greenland_5km_v1.1.nc`. Top row: $q = 0.1$ and $e = 1, 3, 6$ (left-to-right). Middle row: $q = 0.5$. Bottom row: $q = 1.0$. All subfigures have common color scale (velocity m/a), as shown in the right-most figure, with 100 m/a contour shown in all cases (solid black).

2.10 Handling NetCDF files

PISM takes one or more NetCDF files as input, then it does some computation, and then it produces one or more NetCDF files as output. But other tools are usually needed to help to extract meaning from NetCDF files, and yet more NetCDF tools help with creating PISM input files or post-processing PISM output files. Thus we finish this section with a list of NetCDF tools in Table 1.

The PISM authors use `ncview` and "`ncdump -h`" for quick visualization and metadata examination. NCO has powerful command-line manipulation of NetCDF files, but requires some learning. Another such command-line tool is CDO, but to use CDO on PISM files first run the script `nc2cdo.py`, from the `util/` PISM directory, on the file to fix the metadata so that CDO will understand the mapping. Finally, Python scripts using the `netcdf4-python` package (see the PISM Installation Manual) are often the best way to non-trivially change a NetCDF file or make publishable figures from it. Matlab also has good NetCDF I/O capabilities.

See Table 2 in subsection 3.2 for an overview on the data necessary for modeling. For more information on the format of input files for PISM, see section 4.

Tool	**Site**	**Function**
	`www.unidata.ucar.edu/software/netcdf/`	root for NetCDF information
`ncdump`	*included with any NetCDF distribution*	dump binary NetCDF as `.cdl` (text) file
`ncgen`	*included with any NetCDF distribution*	convert `.cdl` file to binary NetCDF
CDO	`http://code.zmaw.de/projects/cdo`	= Climate Data Operators; command-line tools, including conservative re-mapping
IDV	`http://www.unidata.ucar.edu/software/idv/`	more complete visualization
NCO	`http://nco.sourceforge.net/`	= NetCDF Operators; command-line tools
NCL	`http://www.ncl.ucar.edu`	= NCAR Command Language
`ncview`	`meteora.ucsd.edu/~pierce`	quick graphical view
PyNGL	`http://www.pyngl.ucar.edu`	Python version of NCL

Table 1: A selection of tools for viewing and modifying NetCDF files.

3 Ice dynamics, the PISM view

3.1 Two stress balance models: SIA and SSA

At each time-step of a typical PISM run, the geometry, temperature, and basal strength of the ice sheet are included into stress (momentum) balance equations to determine the velocity of the flowing ice. The "full" stress balance equations for flowing ice form a non-Newtonian Stokes model [31]. PISM does not attempt to solve the Stokes equations themselves, however. Instead it can numerically solve, in parallel, two different shallow approximations which are well-suited to ice sheet and ice shelf systems:

- the non-sliding shallow ice approximation (SIA) [53], also called the "lubrication approximation" [31], which describes ice as flowing by shear in planes parallel to the geoid, with a strong connection of the ice base to the bedrock, and
- the shallow shelf approximation (SSA) [112], which describes a membrane-type flow of floating ice [75], or of grounded ice which is sliding over a weak base [71, 99].

The SIA equations are easier to solve numerically than the SSA, and easier to parallelize, because they are local in each column of ice. Specifically, they describe the vertical shear stress as a local function of the driving stress [78]. They can confidently be applied to those grounded parts of ice sheets for which the basal ice is frozen to the bedrock, or which is minimally sliding, and where the bed topography is relatively slowly-varying in the map-plane [31]. These characteristics apply to the majority (by area) of the Greenland and Antarctic ice sheets.

We solve the SIA with a non-sliding base because the traditional [40, 55, 86] addition of ad hoc "sliding laws" into the SIA stress balance, and especially schemes which "switch on" at the pressure-melting temperature [85], have bad continuum [32] and numerical [17, appendix B] modeling consequences.

The SSA equations can confidently be applied to large floating ice shelves, which have small depth-to-width ratio and negligible basal resistance [75, 76]. The flow speeds in ice shelves are frequently an order-of-magnitude higher than in the non-sliding, grounded parts of ice sheets.

Terrestrial ice sheets also have fast-flowing grounded parts, however, called "ice streams" or "outlet glaciers" [107]. Such features appear at the margin of, and sometimes well into the interior of, the Greenland [62] and Antarctic [9] ice sheets. Describing these faster-flowing grounded parts of ice sheets requires something more than the non-sliding SIA. This is because adjacent columns of ice which have different amounts of basal resistance exert strong "longitudinal" or "membrane" stresses [99] on each other.

In PISM the SSA may be used as a "sliding law" for grounded ice which is already modeled everywhere by the non-sliding SIA [17, 115]. For grounded ice, in addition to including shear in planes parallel to the geoid, we must balance the membrane stresses where there is sliding. This inclusion of a membrane stress balance is especially important when there are spatial and/or temporal changes in basal strength. This "sliding law" role for the SSA is in addition to its more obvious role in ice shelf modeling. The SSA plays both roles in a PISM whole ice sheet model in which there are large floating ice shelves (e.g. as in Antarctica [37, 74, 115]; see also section 12.2 of the current Manual).

The "SIA+SSA hybrid" model is recommended for most whole ice sheet modeling purposes because it seems to be a good compromise given currently-available data and computational power. A related hybrid

model described by Pollard and deConto [89] adds the shear to the SSA solution in a slightly-different manner, but it confirms the success of the hybrid concept.

By default, however, PISM does not turn on (activate) the SSA solver. This is because a decision to solve the SSA must go with a conscious user choice about basal strength. The user must both use a command-line option to turn on the SSA (e.g. option `-stress_balance ssa`; see section 6.1) and also make choices in input files and runtime options about basal strength (see section 7.1). Indeed, uncertainties in basal strength boundary conditions usually dominate the modeling error made by not including higher-order stresses in the balance.

When the SSA model is applied a parameterized sliding relation must be chosen. A well-known SSA model with a linear basal resistance relation is the Siple Coast (Antarctica) ice stream model by MacAyeal [71]. The linear sliding law choice is explained by supposing the saturated till is a linearly-viscous fluid. A free boundary problem with the same SSA balance equations but a different sliding law is the Schoof [99] model of ice streams, using a plastic (Coulomb) sliding relation. In this model ice streams appear where there is "till failure" [78], i.e. where the basal shear stress exceeds the yield stress. In this model the location of ice streams is not imposed in advance.

As noted, both the SIA and SSA models are *shallow* approximations. These equations are derived from the Stokes equations by distinct small-parameter arguments, both based on a small depth-to-width ratio for the ice sheet. For the small-parameter argument in the SIA case see [31]. For the corresponding SSA argument, see [112] or the appendices of [99]. Schoof and Hindmarsh [104] have analyzed the connections between these shallowest models and higher-order models, while [41] discusses ice dynamics and stress balances comprehensively. Note that SIA, SSA, and higher-order models all approximate the pressure as hydrostatic.

Instead of a SIA+SSA hybrid model as in PISM, one might use the Stokes equations, or a "higher-order" model (i.e. less-shallow approximations [13, 83]), but this immediately leads to a resolution-versus-stress-inclusion tradeoff. The amount of computation per map-plane grid location is much higher in higher-order models, although careful numerical analysis can generate large performance improvements for such equations [14].

Time-stepping solutions of the mass conservation and energy conservation equations, which use the ice velocity for advection, can use any of the SIA or SSA or SIA+SSA hybrid stress balances. No user action is required to turn on these conservation models. They can be turned off by user options `-no_mass` (ice geometry does not evolve) or `-energy none` (ice enthalpy and temperature does not evolve), respectively.

3.2 A hierarchy of simplifying assumptions for grounded ice flow

Table 2 describes a hierarchy of models, listed roughly in order of increasing effectiveness in modeling grounded ice sheets with fast flow features. This is also the order of increasing need for data to serve as boundary and initial conditions, however, as also described in the Table.

It may also be helpful to view the implemented stress balances as PISM software components (C++ classes). Figure 14 shows the sequences of actions taken by the SIA-only, SSA-only, and SIA+SSA hybrid model components. In each case a membrane stress solution is generated first, then a distribution of vertical shear in the column of ice is generated second, and finally a use of incompressibility computes the vertical component of the velocity. The nonsliding SIA-only model has a trivialized membrane stress

Model	Assumptions	Required data
perfectly-plastic ice	*steady state*; ice has shear stresses below a predetermined ice "yield stress"; also needs predetermined location of ice sheet margin	• bed elevation
balance velocities	*steady state*; ice flows down surface gradient [61]	[*same as above, plus:*] • surface mass balance • (initial) ice thickness
ISOTHERMAL SIA	non-sliding lubrication flow, fixed softness [20, 56]	[*same as above, but time-dependence is allowed*]
THERMO-COUPLED SIA	non-sliding lubrication flow, temperature-dependent softness [18, 85]	[*same as above, plus:*] • surface temperature • geothermal flux
POLYTHERMAL SIA	allows liquid water fraction in temperate ice; conserves energy better [6, 40]	[*same as above*]
SIA + SSA HYBRID	SSA as a sliding law for thermo-coupled SIA [17, 115]; polythermal by default	[*same as above, plus:*] • model for subglacial water • model for basal resistance
Blatter-Pattyn	"higher-order", bridging stresses [13, 83, 102]	[*same as above*]

Table 2: Hierarchy of flow models in PISM for the grounded parts of ice sheets. Listed from most to fewest simplifying assumptions *and* from least to greatest need for boundary data. The *italicized* models are planned for future versions of PISM but are not implemented so far.

solution. The SSA-only model has a trivialized computation of vertical shear.

3.3 Evolutionary versus diagnostic modeling

The main goal of a numerical ice sheet model like PISM is to be a dynamical system which evolves as similarly as possible to the modeled ice sheet. Such a goal assumes one has the "right" climate inputs and parameter choices at each time step. It also assumes one has the "right" initial conditions, such as an adequate description of the present state of the ice sheet, but this assumption is rarely satisfied. Instead a variety of heuristics must be used to minimally-initialize an ice sheet model. For options associated to establishing mathematical initial conditions when first starting PISM, see section 4.

Inside PISM are evolution-in-time partial differential equations which are solved by taking small time steps. "Small" may vary from thousandths to tens of model years, in practice, depending primarily on grid resolution, but also on modeled ice geometry and flow speed. Time steps are chosen adaptively in PISM, according to the stability criteria of the combined numerical methods [17, 18].

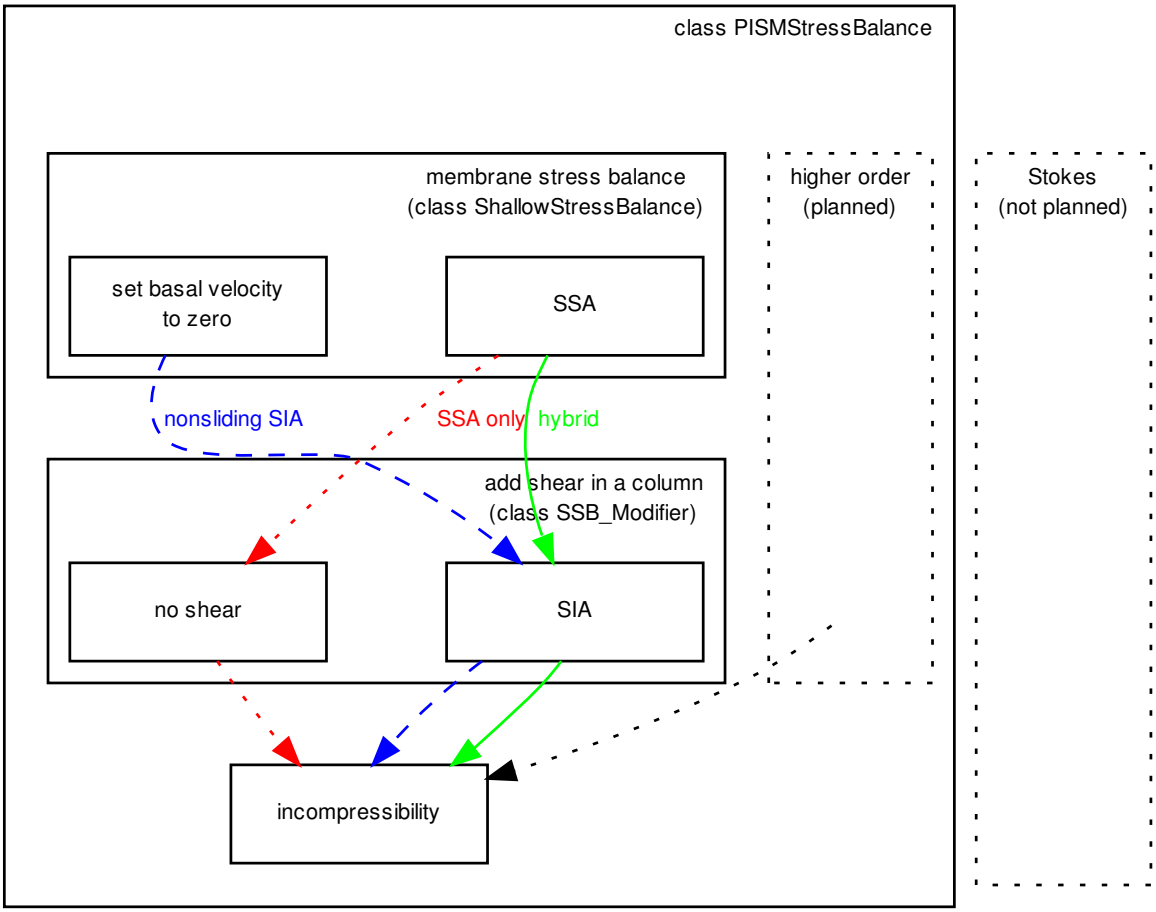


Figure 14: The SIA-only, SSA-only, and SIA+SSA hybrid models represent different "routes" through stress balance PISM components. In each case the inputs are ice geometry and boundary stresses, and the final output is a three-dimensional velocity field within the ice.

However, especially for ice streams and shelves, non-time-stepping "diagnostic" solution of the stress balance partial differential equations might be the desired computation, and PISM can also produce such "diagnostic" velocity fields. Such computations necessarily assume that the ice geometry, viscosity, and boundary stresses are known. Because of the slowness of the ice, in the sense that inertia can be neglected in the stress balance [31], such computations can determine the ice velocity.

Sections 2 and 12.2 give examples illustrating evolutionary and diagnostic modes of PISM, respectively. The first describes time-stepping evolution models for the Greenland ice sheet, while the second describes a diagnostic SSA model for the Ross ice shelf.

3.4 Climate inputs, and their interface with ice dynamics

Because PISM's job is to approximate ice flow, its "world view" is centered around ice dynamics. The discussion of boundary conditions in this Manual is thus ice-dynamics-centric. On the other hand, there is no constraint on the nature of, or completeness of, climate models which could be coupled to PISM. This

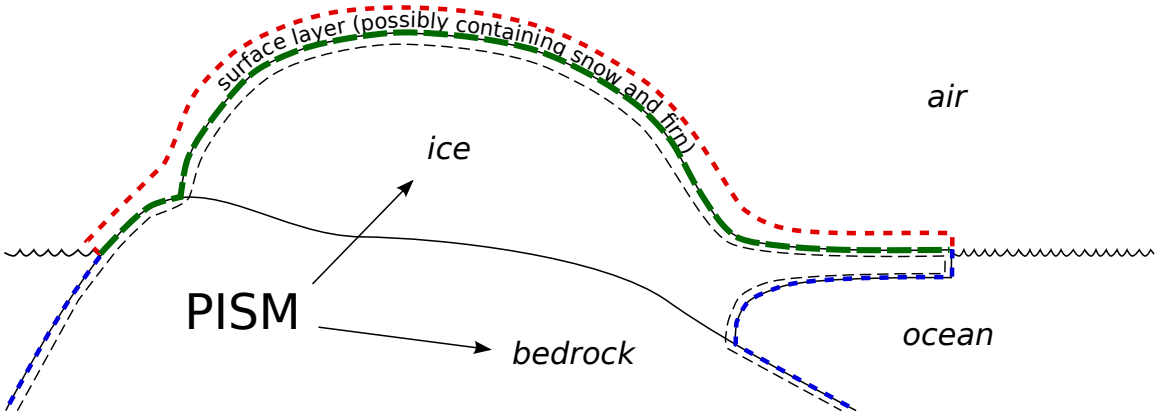


Figure 15: PISM's view of interfaces between an ice sheet and the outside world

section therefore explains a PISM organizing principle, namely that *climate inputs affect ice dynamics by a well-defined interface*.

Almost no attempt is made here to describe the physics of the climate around ice sheets, so see [25] for terminology and [49] for a review of how surface melt can be modeled. See the Climate Forcing Manual for much more information on PISM's climate-coupling-related options and on the particular fields which are shared between the ice dynamics core and the climate model. Table 3 lists fields which are needed as boundary conditions at the interfaces.

All PISM ice sheet models have some kind of interface (**green** in Figure 15) to a subaerial surface processes layer containing snow, firn, and liquid (or refrozen) runoff. The surface layer is assumed to cover the whole surface of the ice, and all grounded areas that the ice might occupy, including ablation areas and ice-free land. We also always have an interface (**blue**) to the ocean, but this interface is inactive if there is no floating ice.

Boundary surface	**Fields (conditions)**
upper surface of the surface processes layer (**red**)	*optional*; typically: air temperature, precipitation
top ice surface, but below firn (**green**)	*required*: boundary temperature (or enthalpy), mass flux (SMB) into the ice
ice shelf basal surface (**blue**)	*required*: mass flux into the ocean, boundary temperature
bottom surface of thermally-modeled bedrock layer (not shown)	*required*: geothermal flux

Table 3: Boundary conditions required by PISM's ice dynamics core; see Figure 15. The optional **red** interface is absent if PISM does not "own" the surface processes layer.

The surface processes layer might be very simple. It might either read the important fields from a file or otherwise transfer them from a separate (non-PISM) climate model. If, however, the surface processes layer is "owned" by the PISM model then there is an additional interface (**red**) to the atmosphere above. In no case does PISM "own" the atmosphere; if it has an interface to the atmosphere at all then it reads atmosphere fields from a file or otherwise transfers them from a climate model.

Regarding the base of the ice, the temperature of a layer of bedrock in contact with grounded ice is generally included in PISM's conservation of energy model; see subsections 5.1 and 5.2. Also, as described in section 7.3, PISM can apply an optional bed deformation component approximating the movement of the Earth's crust and upper mantle in response to changing ice load. In these senses everything below the black dashed line in Figure 15 is always "owned" by PISM.

The PISM ice dynamics core would like to get the required fields listed in Table 3 directly from observations or measurements, or directly from a GCM. In many realistic modeling situations, however, PISM code must be used for all or part of the surface processes modeling necessary to provide the ice-dynamics core with the needed fields. Due to differences in model resolutions and required downscaling, this need for some PISM-based boundary-processes modelling may occur even in some cases where PISM is coupled to a GCM. Thus, to be able to use the data that is available, a PISM run might use components that are responsible for modeling surface (snow) processes or sub-shelf/ocean interaction. These components might be very minimal, merely turning data that we already have into data in the right units and with the right metadata.

Thus we have PISM's design: the ice-dynamics PISM core does not contain any parameterization or other model for boundary mass or energy fluxes into or out of the ice. These boundary parameterizations and models are present in the PISM source code, however, as instances of *pism::Component* classes. This simplifies customizing and debugging PISM's climate inputs, and it promotes code reuse. It isolates the code that needs to be changed to couple PISM to different climate models.

The classes *pism::SurfaceModel*, *pism::AtmosphereModel*, and *pism::OceanModel* are all derived from *pism::Component*. Corresponding to the **red** dashed line in Figure 15, a *pism::AtmosphereModel* might not even be present in some PISM configurations. While they are required, *pism::SurfaceModel* and *pism::OceanModel* may contain (hide) anything from nearly-trivial parameterizations of ice surface temperatures and mass fluxes to a GCM of great complexity.

The "modifiers" in Figure 16 adjust the climate model inputs. Modifiers can be chained together so that multiple modifications are made to the outputs of the original component. For example, ice-core-derived air temperature offsets, used to model the space-time distribution of paleo-climatic surface temperature, is an example of an implemented modifier. Please see the Climate Forcing Manual for a list of climate components and modifiers included in PISM source code and other details. Users wishing to customize PISM's climate inputs and/or couple PISM to a climate model should additionally see the *PISM Source Browser* at `http://www.pism-docs.org/wiki/doku.php?id=browser` and the documentation therein.

Figure 16 illustrates the data flow needed by the ice dynamics core. The data flow in the other direction, i.e. needed by the model to which PISM is coupled, depends on particular modeling choices, but great flexibility is allowed.

Why describe all this structure here? On the one hand, some users may be interested in coupling PISM to other models. On the other hand, the PISM authors do not claim expertise in modeling atmosphere,

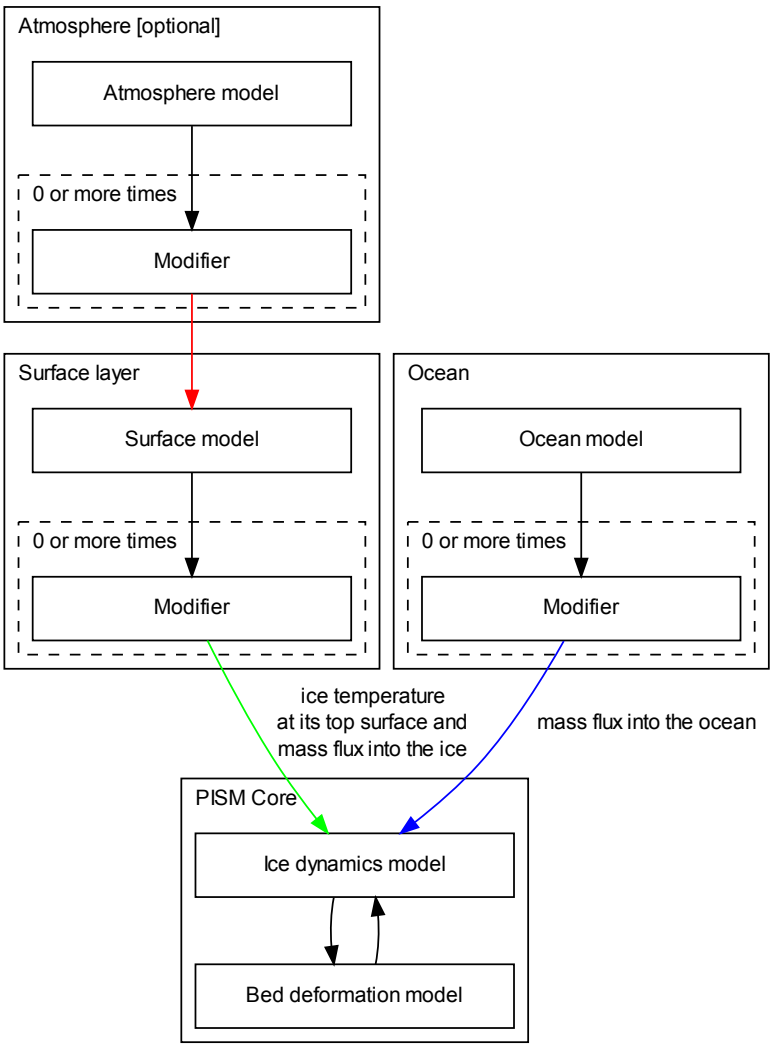


Figure 16: PISM climate input data flow. Colored arrows correspond to interfaces in Figure 15.

ocean, or even snow processes. This separation has a definite code-reliability purpose. PISM users are ultimately responsible for providing the climate inputs they intend.

4 Initialization and bootstrapping

There are three ways to start PISM:

- option `-i` *filename* reads a previously-saved "complete" PISM model state in a NetCDF file, or
- option `-i` *filename* `-bootstrap` reads an "incomplete" NetCDF file and uses heuristics to fill in needed fields, or
- one of the executables `pisms` or `pismv` is used to initialize simplified-geometry experiments or verification tests from formulas in the source code, and thus no input file is required.

One of the first two choices is required when using the executable `pismr`. Modeling usually starts with the `-i input.nc -bootstrap` option because real ice sheet observations are never complete initial conditions. Runs with multiple stages often use the `-i` option after the first stage.

4.1 Initialization from a saved model state

"Initialization" has the specific, simple meaning in PISM that option "`-i`" was used. If a previous PISM run has saved a NetCDF file using "`-o`" then that file will contain complete initial conditions for continuing the run. The output file from the last run can be loaded with "`-i`":

```
$ pisms -eisII A -y 100 -o foo.nc
$ pisms -eisII A -i foo.nc -y 100 -o bar.nc
```

As noted verification tests (section 10) and simplified-geometry experiments (section 11) do not need input files at all because they initialize from formulas in the source code. They can, however, be continued from saved model states using `-i`. Specifying the simplified geometry experiment or verification test *is*, however, necessary if the run is to continue with the climate inputs for that experiment or test. For example, based on the above `pisms` runs, it is valid to do

```
$ pismr -i foo.nc -y 100 -o bar.nc
```

but the climate and other parameters use PISM default values, and thus are not (necessarily) the values specified in EISMINT II.

As a technical note about saved states, a PISM run with `-stress_balance ssa` also saves the last SSA velocities to the output file in variables `u_ssa` and `v_ssa`. The presence of these velocities adds efficiency in restarting because an initial estimate speeds up the solution of the SSA stress balance equations. If you want to use `-i` but also ignore these velocities then use option `-dontreadSSAvels`.

`-i` file format

PISM produces CF-1.5 compliant NetCDF files. The easiest way to learn the output format *and* the `-i` format is to do a simple run and then look at the metadata in the resulting file, like this:

```
$ pisms -eisII A -y 10 -o foo.nc
$ ncdump -h foo.nc | less
```

Note that variables in the output file have a `pism_intent` attribute. When `pism_intent` = `diagnostic`, the variable can be deleted from the file without affecting whether PISM can use it as a `-i` input file. Variables with `pism_intent` = `model_state`, by contrast, must be present when using `-i`.

The automatically-produced `time` variable has a `units` attribute like `"seconds since 1-1-1"` because the CF metadata conventions require a reference date. By default PISM ignores this reference date except when it is used in unit conversions based on a calendar (see below).

4.2 Bootstrapping

"Bootstrapping" in PISM means starting a modeling run with less than sufficient data, and letting essentially heuristic models fill in needed fields. These heuristics are applied before the first time step is taken, so they are part of an initialization process. Bootstrapping uses the option `-bootstrap`; see subsection 2.3 for an example.

The need for an identified stage like "bootstrapping" comes from the fact that initial conditions for the evolution equations describing an ice sheet are not all observable. As a principal example of this problem, these initial conditions include the temperature within the ice. Glaciological observations, specifically remote-sensed observations which cover a large fraction or all of an ice sheet, never include this temperature field in practice. Thus ice sheet modelling often does something like this to get "reasonable" initial fields within the ice:

1. start only with (potentially) observable quantities like surface elevation, ice thickness, ice surface temperature, surface mass balance, and geothermal flux,

2. "bootstrap" as defined here, using heuristics to fill in temperatures at depth and to give a preliminary estimate of the basal sliding condition and the three-dimensional velocity field, and

3. a) *either* do a long run, often holding the current geometry and surface conditions steady, to evolve toward a steady state which has compatible temperature, stress, and velocity fields,

   b) *or* do a long run using an additional (typically spatially-imprecise) historical record from an ice core or a sea bed core (or both), to apply forcing to the surface temperature or sea level (for instance), but with the same functional result of filling in temperature, stress, and velocity fields.

When using `-bootstrap` you will need to specify both grid dimensions (using `-Mx`, `-My` and `-Mz`; see subsection 5.2) and the height of the computational box for the ice with `-Lz` (subsection 5.1). The data read from the file can determine the horizontal extent of the model, if options `-Lx`, `-Ly` are not set. The additional required specification of vertical extent by `-Lz` is reasonably natural because typical data used in "bootstrapping" are two-dimensional. Using `-bootstrap` without specifying all four options `-Mx`, `-My`, `-Mz`, `-Lz` is an error.

If `-Lx` and `-Ly` specify horizontal grid dimensions smaller than in the bootstrapping file, PISM will cut out the center portion of the domain. Alternatively, options `-x_range` and `-y_range` each take a list of two numbers, a list of minimum and maximum x and y coordinates, respectively (in meters), which makes it possible to select a subset that is not centered in the bootstrapping file's grid.

For the key issue of what heuristic is used to determine the temperatures at depth, there are two methods. The default method uses ice thickness, surface temperature, surface mass balance, and geothermal flux. The temperature is set to the solution of a steady one-dimensional differential equation in which conduction and vertical advection are in balance, and the vertical velocity linearly-interpolates between the surface mass balance rate at the top and zero at the bottom. The non-default method, set with option `-boot_temperature_heuristic quartic_guess`, was the default in older PISM versions (`stable0.5` and earlier); it does not use the surface mass balance and instead makes a more-heuristic estimate of the vertical temperature profile based only on the ice thickness, surface temperature, and geothermal flux.

`-bootstrap` file format

Allowed formats for a bootstrapping file are relatively simple to describe.

1. NetCDF variables should have the `units` containing a UDUNITS-2-compatible string. If this attribute is missing, PISM will assume that a field uses MKS units.[9]

2. NetCDF coordinate variables should have `standard_name` or `axis` attributes. These are used to determine which *spatial* dimension a NetCDF dimension corresponds to; for example see `ncdump -h` output from a file produced by PISM. The `x` and `y` dimensions need not be called "`x`" and "`y`".

3. Coordinate variables have to be strictly-increasing.

4. Three-dimensional variables will be ignored in bootstrapping.

5. The `standard_name` attribute is used, when available, to identify a variable, so variable names need not match corresponding variables in a PISM output file. See http://www.pism-docs.org/wiki/doku.php?id=browser for a list of CF standard names used in PISM. Specifically, the bed elevation (topography) is read by `standard_name` = `bedrock_altitude` and the ice thickness by `standard_name` = `land_ice_thickness`.

6. Any two-dimensional variable except bed topography and ice thickness may be missing. For missing variables some heuristic will be applied. See table 2 for a sketch of the data necessary for bootstrapping; see `src/base/iMbootstrap.cc` for all further details.

7. Surface elevation is ignored if present. Users with surface elevation and bed elevation data should compute the ice thickness variable, put it in the bootstrapping file, and set its `standard_name` to `land_ice_thickness`.

[9] PISM uses a library called UDUNITS-2 to convert data present in an input file to MKS. This means that having ice thickness in feet or temperature in Fahrenheit *is* allowed.

5 Modeling choices: Grid and time

5.1 Computational box

PISM does all simulations in a computational box which is rectangular in the PISM coordinates. The coordinate system has horizontal coordinates x, y and a vertical coordinate z. The z coordinate is measured positive upward from the base of the ice. The vector of gravity is in the negative z direction. The surface $z = 0$ is the base of the ice, however, and thus is usually not horizontal in the sense of being parallel to the geoid. The surface $z = 0$ is the base of the ice both when the ice is grounded and when the ice is floating.

When the ice is grounded, the true physical vertical coordinate z', namely the coordinate measure relative to a reference geoid, is given by $z' = z + b(x, y)$ where $b(x, y)$ is the bed topography. The surface $z' = h(x, y)$ is the surface of the ice. In the grounded case the equation $h(x, y) = H(x, y) + b(x, y)$ always applies if $H(x, y)$ is the thickness of the ice.

In the floating case, the physical vertical coordinate is $z' = z - (\rho_i/\rho_s)H(x, y)$ where ρ_i is the density of ice and ρ_s the density of sea water. Again $z = 0$ is the base of the ice, which is the surface $z' = -(\rho_i/\rho_s)H(x, y)$. The surface of the ice is $h(x, y) = (1 - \rho_i/\rho_s)H(x, y)$. The *flotation criterion* $-(\rho_i/\rho_s)H(x, y) > b(x, y)$ applies.

The computational box can extend downward into the bedrock. As $z = 0$ is the base of the ice, the bedrock corresponds to negative z values regardless of its true (i.e. z') elevation.

The extent of the computational box, along with its bedrock extension downward, is determined by four numbers `Lx`, `Ly`, `Lz`, and `Lbz` (see Figure 17 and Table 4). The first two of these are half-widths and have units of kilometers when set by options or displayed.

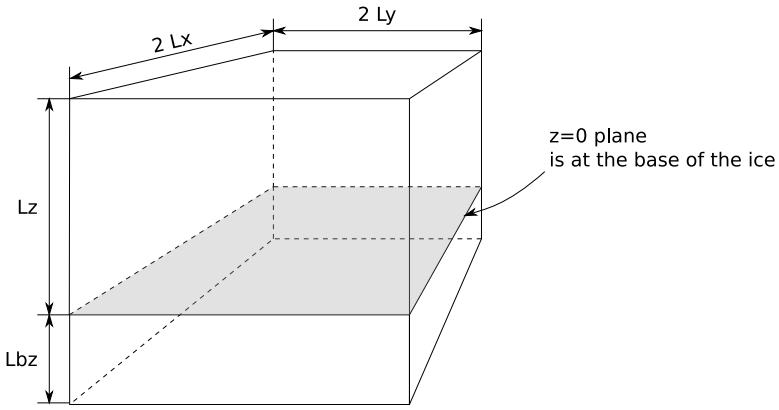


Figure 17: PISM's computational box.

5.2 Spatial grid

The PISM grid covering the computational box is equally spaced in horizontal (x and y) directions. Vertical spacing in the ice is quadratic by default but optionally equal spacing can be chosen; choose with options `-z_spacing` [`quadratic`, `equal`] at bootstrapping. The grid read from a "`-i`" input file is

Option	Meaning
`-Lx` (km)	Half-width of the computational domain (in the x-direction)
`-Ly` (km)	Half-width of the computational domain (in the y-direction)
`-Lz` (meters)	Height of the computational domain; must exceed maximum ice thickness
`-Lbz` (meters)	Depth of the computational domain in the bedrock thermal layer

Table 4: Options defining the extent of PISM's computational box.

used as is. The bedrock thermal layer model always uses equal vertical spacing.

The grid is described by four numbers, namely the number of grid points `Mx` in the x direction, the number `My` in the y direction, the number `Mz` in the z direction within the ice, and the number `Mbz` in the z direction within the bedrock thermal layer. These are specified by options `-Mx`, `-My`, `-Mz`, and `-Mbz`, respectively. The defaults are 61, 61, 31, and 1, respectively. Note that `Mx`, `My`, `Mz`, and `Mbz` all indicate the number of grid *points* so the number of grid *spaces* are one less, thus 60, 60, 30, and 0 in the default case.

The lowest grid point in a column of ice, at $z = 0$, coincides with the highest grid point in the bedrock, so `Mbz` must always be at least one. Choosing `Mbz`> 1 is required to use the bedrock thermal model. When a thermal bedrock layer is used, the distance `Lbz` is controlled by the `-Lbz` option. Note that `Mbz` is unrelated to the bed deformation model (glacial isostasy model); see section 7.3.

In the quadratically-spaced case the spacing near the ice/bedrock interface is about four times finer than it would be with equal spacing for the same value of `Mz`, while the spacing near the top of the computational box is correspondingly coarser. For a detailed description of the spacing of the grid, see the documentation on `IceGrid::compute_vertical_levels()` in the PISM class browser.

The user should specify the grid when using `-bootstrap` or when initializing a verification test (section 10) or a simplified-geometry experiment (section 11). If one initializes PISM from a saved model state using `-i` then the input file determines all grid parameters. For instance, the command

```
$  pismr -i foo.nc -y 100
```

should work fine if `foo.nc` is a PISM output file. Because `-i` input files take precedence over options,

```
$  pismr -i foo.nc -Mz 201 -y 100
```

will give a warning that "`PISM WARNING: ignoring command-line option '-Mz'`".

5.2.1 Parallel domain distribution

When running PISM in parallel with `mpiexec -n N`, the horizontal grid is distributed across N processes[10]. PISM divides the grid into N_x parts in the x direction and N_y parts in the y direction. By

[10] In most cases one process corresponds to one "core" of your computer.

Option	Meaning
`-y` (years)	Number of model years to run.
`-ys` (years)	Model year at which to start the run. Also resets the model time, ignoring any time in the input file.
`-ye` (years)	Model year at which to end the run.

Table 5: Command-line options controlling PISM time.

default this is done automatically, with the goal that $N_x \times N_y = N$ and N_x is as close to N_y as possible. Note that N should, therefore, be a composite (not prime) number.

Users seeking to override this default can specify N_x and N_y using the `-Nx` and `-Ny` command-line options.

Once N_x and N_y are computed, PISM computes sizes of sub-domains $M_{x,i}$ so that $\sum_{i=1}^{N_x} M_{x,i} = \mathrm{Mx}$ and $M_{x,i} - \lfloor \mathrm{Mx}/N_x \rfloor < 1$. To specify strip widths $M_{x,i}$ and $M_{y,i}$, use command-line options `-procs_x` and `-procs_y`. Each option takes a comma-separated list of numbers as its argument. For example,

```
$ mpiexec -n 3 pisms -Mx 101 -My 101 -Nx 1 -Ny 3 -procs_x 101 -procs_y 20,61,20
```

splits a 101×101 grid into 3 strips along the x axis.

To see the parallel domain decomposition from a completed run, see the `rank` variable in the output file, e.g. using `-o_size big`. The same `rank` variable is available as a spatial diagnostic field (subsection 9.3).

5.3 Model time

Table 5 gives the command-line options which control PISM time. If option `-ys` is absent then the start year is read from the input file (if present) or it defaults to zero. The default value for the end year is the start year plus the given (`-y`) run length. If both `-ys` and `-ye` are used then the run length is set to the difference. Using all three of `-ys`, `-y` and `-ys` is not allowed; this generates an error message.

5.4 Calendars

Most of PISM, and its ice dynamics core in particular, only needs to know the length of the current time-step. Internally PISM stores time in "seconds since a specified moment" and thus PISM generally does not use or need a calendar.[11] We refer to PISM internal time as *model time*.

One can select a calendar for more precise control of the model time, however. A "calendar" is a concept that is part of the CF Metadata Conventions. Choosing a calendar is appropriate for runs for specific temporal periods like "the 18th-century" or "1989–2010". The calendar is generally needed because specific

[11] Note seconds are part of SI units.

`gregorian` or `standard`	Mixed Gregorian/Julian calendar used today.
`proleptic_gregorian`	Gregorian calendar extended to dates before 1582-10-15.
`noleap` or `365_day`	Calendar with fixed-length 365-day years
`360_day`	Calendar with fixed-length 360-day years divided into 30-day months
`julian`	Julian calendar
`none`	no calendar

Table 6: Calendars supported by PISM. Please see CalCalcs documentation for details.

knowledge of lengths of months and years is required to use climate data properly or to facilitate model validation.

PISM uses CalCalcs by David W. Pierce to perform calendric computations. This lets us support all the calendars defined by the CF Metadata Conventions document except for the `366_day` (`all_leap`) calendar.

Time units in PISM's output files always contain a reference date because it is required by the CF metadata conventions.

By default PISM does not use a calendar. This is appropriate for runs that do not require precise application of forcing data or reporting on particular dates (paleo-climate runs, for example). In this mode PISM ignores the reference date in time unit specifications (such as "`days since 1969-7-20`"), though the value set using `reference_date` configuration parameter is saved in (is passed forward into) output files.

Selecting a calendar using the `calendar` configuration parameter or the `-calendar` command-line option enables calendar-based time management; see Table 6. The implications of selecting a calendar are:

- PISM uses the `units` attribute of coordinate variables *literally* (including the reference date) in unit conversions. Please make sure that the `time` variable in all forcing files has the units attribute such as "`days since 2012-1-1`". PISM will stop with an error message if a time variable does not have a reference date in its unit specification.
- It is important to use units that are a fixed multiple of "seconds", such as "`minutes since 1989-1-1`" or "`days since 1999-12-31`" and avoid "months" and "years". (PISM uses UDUNITS-2 to convert units, and in UDUNITS one month is always interpreted as $\frac{1}{12} \cdot 365.242198781$ days.) Please see the CF Conventions document for details.
- PISM uses dates in standard output:

```
...
   time interval (length)   [2012-01-01, 2021-12-31]  (10.000 years)
...
S 2012-05-26:  0.00011    0.6306   0.00000000           0.00000
```

```
$v$Eh m (dt=0.10000)
S 2012-07-01:  0.00014    0.6306   0.00000000           0.00000
```

Just like in the no-calendar mode, run length, run start and run end times are specified using `-y`, `-ys` and `-ye` command-line options, respectively. Arguments of these options are interpreted in a slightly different manner, though:

- the run length option `-y` takes an *integer* argument, interpreted as the number of *calendar* years
- options `-ys` and `-ye` take *dates* as arguments.

For example, either of the following commands sets up a run covering the 21^{st} century:

```
$ pismr -calendar gregorian -ys 2001-1-1 -y 100 ...
$ pismr -calendar standard -ys 2001-1-1 -ye 2101-1-1 ...
```

(These option combinations are equivalent.)

It is also possible to run PISM for the duration of the available forcing data using the `-time_file` *filename* option. The command

```
$ pismr -calendar gregorian -time_file forcing.nc
```

will extract the reference date and run length from `forcing.nc`, respecting time bounds.

When a non-trivial calendar is selected, spatial and scalar time-series can be saved daily, monthly or yearly using these calendric computations. See sections 9.2 and 9.3.

5.4.1 Re-starting an interrupted run using `-time_file`

If a run using `-time_file` gets interrupted but manages to save a backup, re-starting with `-time_file` will attempt to re-do the entire run because options `-y`, `-ys`, and `-ye` are ignored:

```
# This run gets killed but leaves backup.nc:
$ pismr -i input.nc -time_file time.nc -o output.nc
# This WILL NOT start from the time saved in backup.nc
# and continue until the end time in time.nc
$ pismr -i backup.nc -time_file time.nc -o output.nc
```

In this case we want to set the start time of the run from `backup.nc`, but use the end time from `time.nc`. To achieve this, use the option `-time_file_continue_run`.

```
# This run gets killed but leaves backup.nc:
$ pismr -i input.nc -time_file time.nc -o output.nc
# This WILL continue until the end time in time.nc, starting from backup.nc
$ pismr -i backup.nc -time_file time.nc -o output.nc -time_file_continue_run
```

5.5 Diagnostic computations

A "diagnostic" computation can be defined as one where the internal state does not evolve. The internal state of PISM is the set of variables read by "`-i`". You can ask PISM to do a diagnostic computation by setting the run duration to a small number such as 0.001 years (about 9 hours). The duration to use depends on the modeling setup, but should be smaller than the maximum time-step allowed by PISM's stability criteria. Such short runs can also be used to look at additional fields corresponding to the current model state.

As an example, consider these two runs:

```
pisms -y 6000 -o foo.nc
pismr -i foo.nc -y 0.001 -o bar.nc -o_size big
```

The result of the second (short) run is a NetCDF file `bar.nc` which contains the full three-dimensional velocity field in the scalar NetCDF variables `uvel`, `vvel`, and `wvel`, as well as many other variables. The file `foo.nc` does not contain many of these fields because it was written with the default output size of `medium`. The "`-y 0.001`" run has diagnostically "filled-in" all the fields which PISM can model at a time step, but the run duration was chosen so as to avoid significant model state evolution during the run.

This diagnostic mode is often associated to the modeling of ice shelves and ice streams. Subsection 12.2 describes using a short "diagnostic" run to model the Ross ice shelf [72]. Verification tests I and J, section 10, are diagnostic calculations using the SSA.

The NetCDF model state saved by PISM at the end of an *evolution* run (i.e. with "`-y` Y" for $Y > 0$) does not, under the default `-o_size medium` output size, contain the three-dimensional velocity field. Instead, it contains just a few more variables than those which are needed to restart the run with `-i`. One can force PISM to save all the supported diagnostic quantities at the end of a time-stepping run using the option `-o_size big`. Or one can go back and do a "`-y small_number`" diagnostic run using `-o_size big`.

5.6 Disabling PISM components

Certain major model components, unlike more peripheral ones like bed deformation or calving, are "on" by default. They do not need to be turned on explicitly. For example, the SIA computation is so common that it would be a hassle to require an option to turn it on every time you need it.

But sometimes one wants to disable particular components, during model spin-up, for example. PISM has the following "off" switches:

- `-no_mass` disables the mass-continuity (conservation of mass) step
- `-energy none` disables the conservation of energy computation
- `-energy cold` makes PISM use temperature instead of enthalpy in the energy conservation code
- `-stress_balance none` disables the stress balance computation (useful for testing surface mass balance inputs)

5.7 Dealing with more difficult modeling choices

Most uses of an ice sheet model depend on careful modeling choices in situations where there are considerable uncertainties *and* the model results depend strongly on those choices. There may be, at the present state of knowledge, *no clear default values* that PISM can provide. Furthermore, the available PISM options and sub-models are known to *not* be sufficient for all users. Thus there are modelling situations for which we know the user may have to do a great deal more hard work than just choose among PISM runtime options.

Here are example cases where users have worked hard:

- User made use of available data in order to choose parameters for existing PISM models. These parameters then override PISM defaults.

  > *Example.* Use regional atmosphere model output to identify PDD parameters suitable for modeling surface mass balance on a particular ice sheet. Then supply these parameters to PISM by a `-config_override` file.

- User wrote code, including code which modified current PISM internals, either to add additional processes or to "correct" PISM default process models.

  > *Example.* Add a new sub-ice-shelf melt model by modifying C++ code in the `src/coupler/` directory.

- User simplified the model in use, instead of the default which was more elaborate.

  > *Example.* Instead of using the PISM default mechanism connecting basal melt rate and basal strength, bypass this mechanism by generating a map of yield stress `tauc` directly and supplying it as input.

6 Modeling choices: Ice dynamics and thermodynamics

6.1 Choosing the stress balance

The basic stress balance used for all grounded ice in PISM is the non-sliding, thermomechanically-coupled SIA [18]. For the vast majority of most ice sheets, as measured by area or volume, this is an appropriate model, which is an $O(\epsilon^2)$ approximation to the Stokes model if ϵ is the depth-to-length ratio of the ice sheet [31].

The shallow shelf approximation (SSA) stress balance applies to floating ice. See the Ross ice shelf example in section 12.2 for an example in which the SSA is only applied to floating ice.

The SSA is also used in PISM to describe the sliding of grounded ice and the formation of ice streams [17]. Specifically for the SSA with "plastic" (Coulomb friction) basal resistance, the locations of ice streams are determined as part of a free boundary problem of Schoof [99], a model for emergent ice streams within a ice sheet and ice shelf system. This model explains ice streams through a combination of plastic till failure and SSA stress balance.

This SSA description of ice streams is, however, also the preferred "sliding law" for the SIA [17, 115]. The SSA should be combined with the SIA, in this way, in preference to classical SIA sliding laws which make ice basal velocity a local function of the basal value of the driving stress.. The resulting combination of SIA and SSA is a "hybrid" approximation of the Stokes model [115]. Option `-stress_balance ssa+sia` turns on this "hybrid" model. In this use of the SSA as a sliding law, floating ice is also subject to the SSA.

Of course there is more to the use of a stress balance than just turning it on! At all grounded points a yield stress, or a pseudo-yield-stress in the case of power law sliding (subsection 7.1), is computed from the amount of stored basal water and from a (generally) spatially-varying till strength. The amount of stored basal water is modeled by the subglacial hydrology mode choice (subsection 7.2) based on the basal melt rate which is, primarily, thermodynamically-determined (subsection 7.1).

Table 7 describes the basic choice of stress balance. If the SSA stress balance is used, a choice of two solvers is available, namely `-ssa_method fd` (default) or `-ssa_method fem`. See Table 8, which describes additional controls on the numerical solution of the stress balance equations. If option `-ssa_method fd` is chosen then several more controls on numerics are available; see Table 9. If the ice sheet being modeled has any floating ice then the user is advised to read section 8.1 on modeling marine ice sheets.

Option	Semantics
`-stress_balance none`	Turn off ice flow completely.
`-stress_balance sia` (default)	Grounded ice flows by the non-sliding SIA. Floating ice essentially doesn't flow, so this model is not recommended for marine ice sheets.
`-stress_balance ssa`	Use the SSA model exclusively. Horizontal ice velocity is constant throughout ice columns.
`-stress_balance prescribed_sliding`	Use the constant-in-time prescribed sliding velocity field read from a file set using `-prescribed_sliding_file` *filename*, variables `ubar` and `vbar`. Horizontal ice velocity is constant throughout ice columns.
`-stress_balance ssa+sia`	The recommended sliding law, which gives the SIA+SSA hybrid stress balance. Combines SSA-computed velocity, using pseudo-plastic till, with SIA-computed velocity according to the combination in [115]; similar to [17]. Floating ice uses SSA only.
`-stress_balance prescribed_sliding+sia`	Use the constant-in-time prescribed sliding velocity in combination with the non-sliding SIA.

Table 7: The basic choice of stress balance.

Option (default)	Description
`-ssa_method` [`fd`|`fem`]	Both finite difference (`fd`; the default) and finite element (`fem`) versions of the SSA numerical solver are implemented in PISM. The `fd` solver is the only one which allows PIK options (section 8.1). `fd` uses Picard iteration [17], while `fem` uses a Newton method. The `fem` solver has surface velocity inversion capability [44].
`-ssa_eps` (10^{13})	The numerical schemes for the SSA compute an effective viscosity ν which depends on strain rates and ice hardness (thus temperature). The minimum value of the effective viscosity times the thickness (i.e. νH) largely determines the difficulty of solving the numerical SSA. This constant is added to keep νH bounded away from zero: $\nu H \to \nu H + $ `ssa_eps`. Units of `ssa_eps` are Pa m s. Set to zero to turn off this lower bound.
`-ssa_view_nuh`	View the product νH for your simulation as a runtime viewer (section 9.5). In a typical Greenland run we see a wide range of values for νH from $\sim 10^{14}$ to $\sim 10^{20}$ Pa m s.

Table 8: Choice of, and controls on, the numerical SSA stress balance.

Option (default)	**Description**
`-ssa_maxi` (300)	Set the maximum allowed number of Picard (nonlinear) iterations in solving the shallow shelf approximation.
`-ssa_rtol` (10^{-4})	The Picard iteration computes a vertically-averaged effective viscosity which is used to solve the equations for horizontal velocity. Then the new velocities are used to recompute an effective viscosity, and so on. This option sets the relative change tolerance for the effective viscosity. The Picard iteration stops when successive values $\nu^{(k)}$ of the vertically-averaged effective viscosity satisfy $$\|(\nu^{(k)} - \nu^{(k-1)})H\|_1 \le Z\|\nu^{(k)}H\|_1$$ where $Z =$ `ssa_rtol`.
`-ssafd_ksp_rtol` (10^{-5})	Set the relative change tolerance for the iteration inside the Krylov linear solver used at each Picard iteration.

Table 9: Controls on the numerical iteration of the `-ssa_method fd` solver.

6.2 Ice rheology

The "rheology" of a viscous fluid refers to the relation between the applied stress and the resulting deformation, the strain rate. The models of ice rheology available in PISM are all isotropic [78]. A rheology in this class is described by a "flow law", which is, in the most general case in PISM, a function $F(\sigma, T, \omega, P, d)$ in the "constitutive relation" form

$$D_{ij} = F(\sigma, T, \omega, P, d)\, \sigma'_{ij}. \tag{1}$$

Here D_{ij} is the strain rate tensor, σ'_{ij} is the stress deviator tensor, T is the ice temperature, ω is the liquid water fraction, P is the pressure, d is the grain size, and $\sigma^2 = \frac{1}{2}\|\sigma'_{ij}\|_F = \frac{1}{2}\sigma'_{ij}\sigma'_{ij}$ defines the second invariant σ of the stress deviator tensor.

Form (1) of the flow law is used in the SIA, but the "viscosity" form of a flow law, found by inverting the constitutive relation (1), is needed for ice shelf and ice stream (SSA) flow [17]:

$$\sigma'_{ij} = 2\nu(D, T, \omega, P, d)\, D_{ij} \tag{2}$$

Here $\nu(D, T, \omega, P, d)$ is the "effective viscosity" and $D^2 = \frac{1}{2}D_{ij}D_{ij}$.

Most of the flow laws in PISM are of Glen-Nye single-power type. For example,

$$F(\sigma, T) = A(T)\sigma^{n-1} \tag{3}$$

is the common temperature-dependent Glen law [79, 18] (which has no dependence on liquid water fraction, pressure, or grain size). If the ice softness $A(T) = A_0$ is constant then the law is isothermal, whereas if there is dependence on temperature then $A(T)$ is usually a generalization of "Arrhenius" form $A(T) = A\exp(-Q/(RT))$.

The more elaborate Goldsby-Kohlstedt law [36] is a function $F(\sigma, T, P, d)$, but in this case the function F cannot be factored into a product of a function of T, P, d and a single power of σ, as in form (3).

There is only one choice for the flow law which takes full advantage of the enthalpy mode of PISM, which is the thermodynamical modeling (i.e. conservation of energy) default. Namely the Glen-Paterson-Budd-Lliboutry-Duval flow law [6, 69, 79], which is a function $F(\sigma, T, \omega, P)$. This law is the only one in the literature where the ice softness depends on both the temperature and the liquid water fraction, so it parameterizes the (observed) softening of pressure-melting-temperature ice as its liquid fraction increases. One can use this default polythermal law or one may choose among a number of "cold ice" laws listed in Table 10 which do not use the liquid water fraction.

All flow law parameters can be changed using configuration parameters; see section 9.6 and the implementation of flow laws in the *Source Code Browser*. Note that different flow laws have different numbers of parameters, but all have at least two parameters (e.g. A_0 and n in `isothermal_glen`). One can create a new, and reasonably arbitrarily, scalar function F by modifying source code; see source files `flowlaws.hh`, `flowlaws.cc` in `src/base/rheology/`. To assist such modifications, note that Table 10 below also lists the C++ classes declared in `flowlaw.hh`.

Choosing the flow laws for SIA and SSA stress balances

Command-line options `-sia_flow_law` and `-ssa_flow_law` choose which flow law is used by the SIA and SSA stress balances, respectively. Allowed arguments are listed in Tables 10 and 11 below. Viscosity

form (2) is not known for the Goldsby-Kohlstedt law [36], so option "`-ssa_flow_law gk`" is an error.

Name	C++ Class	**Comments and References**
`gpbld`	`GPBLD`	Glen-Paterson-Budd-Lliboutry-Duval law [69], the enthalpy-based default in PISM [6]. Extends the Paterson-Budd law (below) to positive liquid water fraction. If $A_c(T)$ is from Paterson-Budd then this law returns $A(T,\omega) = A_c(T)(1 + C\omega)$, where ω is the liquid water fraction, C is a configuration parameter `gpbld_water_frac_coeff` [default $C = 181.25$], and ω is capped at level `gpbld_water_frac_observed_limit`.
`pb`	`PatersonBudd`	Paterson-Budd law, the cold-mode default. Fixed Glen exponent $n = 3$. Has a split "Arrhenius" term $A(T) = A\exp(-Q/RT^*)$ where $A = 3.615\times10^{-13}\,\mathrm{s^{-1}\,Pa^{-3}}$, $Q = 6.0\times10^4\,\mathrm{J\,mol^{-1}}$ if $T^* < 263$ K and $A = 1.733\times10^{3}\,\mathrm{s^{-1}\,Pa^{-3}}$, $Q = 13.9\times10^4\,\mathrm{J\,mol^{-1}}$ if $T^* > 263$ K; here T^* is pressure-adjusted temperature [79].
`arr`	`PatersonBuddCold`	*Cold* part of Paterson-Budd. Regardless of temperature, the A and Q values for $T^* < 263$ K in the Paterson-Budd law apply. This is the flow law used in the thermomechanically-coupled exact solutions run by `pismv -test F` and `pismv -test G` [18, 16].
`arrwarm`	`PatersonBuddWarm`	*Warm* part of Paterson-Budd. Regardless of temperature, the A and Q values for $T^* > 263$ K in Paterson-Budd apply.
`hooke`	`Hooke`	Hooke law with $A(T) = A\exp(-Q/(RT^*) + 3C(T_r - T^*)^\kappa)$. Fixed Glen exponent $n = 3$ and constants as in [51, 86].
`isothermal_glen`	`IsothermalGlen`	The isothermal Glen flow law. Here $F(\sigma) = A_0\sigma^{n-1}$ with inverse $\nu(D) = \frac{1}{2}B_0 D^{(1-n)/(2n)}$ where A_0 is the ice softness and $B_0 = A_0^{-1/n}$ is the ice hardness.

Table 10: Single-power flow laws. Choose the ice rheology using `-sia_flow_law` and `-ssa_flow_law` and one of the above names. Flow law choices other than `gpbld` do not use the liquid water fraction ω but only the temperature T.

Choose enhancement factor and exponent

An enhancement factor can be added to any flow law through a runtime option. Single-power laws also permit control of the flow law exponent through a runtime option.

Options `-sia_e` and `-ssa_e` set flow enhancement factors for the SIA and SSA respectively. Option `-sia_e` sets "e" in "$D_{ij} = e\,F(\sigma,T,\omega,P,d)\,\sigma'_{ij}$," in equation (1). Option `-ssa_e` sets "e" in the viscosity form so that "$\sigma'_{ij} = e^{-1/n}\,2\,\nu(D,T,\omega,P,d)\,D_{ij}$."

Options `-sia_n` and `-ssa_n` set the exponent when a single-power flow law is used (see Table 10). Simply changing to a different value from the default $n = 3$ is not recommended without a corresponding

Name	C++ Class	Comments and References
`gk`	`GoldsbyKohlstedt`	This law has a combination of exponents from $n = 1.8$ to $n = 4$ [36]. It can only be used by the SIA stress balance. Because it has more than one power, option `-sia_n` has no effect, though `-sia_e` works as expected. This law does not use the liquid water fraction, but only the temperature.

Table 11: The Goldsby-Kohlstedt flow law. Use option `-sia_flow_law gk`.

change to the enhancement factor, however. This is because the coefficient and the power are non-trivially linked when a power law is fit to experimental data [26, 79].

Here is a possible approach to adjusting both the enhancement factor and the exponent. Suppose σ_0 is preferred as a scale (reference) for the driving stress that appears in both SIA and SSA models. Typically this is on the order of one bar or 10^5 Pa. Suppose one wants the same amount of deformation D_0 at this reference driving stress as one changes from the old exponent n_{old} to the new exponent n_{new}. That is, suppose one wants both

$$D_0 = E_{old}\, A\, \sigma_0^{n_{old}} \qquad \text{and} \qquad D_0 = E_{new}\, A\, \sigma_0^{n_{new}}$$

to be true with a new enhancement factor E_{new}. Eliminating D_0 and solving for the new enhancement factor gives

$$E_{new} = E_{old}\, \sigma_0^{n_{old}-n_{new}}. \tag{4}$$

It follows, for example, that if one has a run with values

```
-sia_e 3.0 -sia_n 3.0
```

then a new run with exponent $n = 6.0$ and the same deformation at the reference driving stress of 10^5 Pa will use

```
-sia_e 3.0e-15 -sia_n 6.0
```

because $E_{new} = 3.0\sigma_0^{3-6} = 3.0 \times (10^5)^{-3}$ from equation (4).

A corresponding formula applies to `-ssa_e` if the `-ssa_n` value changes.

Option (default)	Configuration parameter	Comments
`-sia_e` (1.0)	`sia_enhancement_factor`	Note [4, see supplement] used 3.0 for Greenland ice sheet simulations while [74] used 4.5 for simulations of the Antarctic ice sheet with PISM-PIK.
`-sia_n` (3.0)	`sia_Glen_exponent`	See text and eqn (4) to also set `-sia_e` if `-sia_n` changes.
`-ssa_e` (1.0)	`ssa_enhancement_factor`	Note [74] used 0.512 for simulations of the Antarctic ice sheet with PISM-PIK.
`-ssa_n` (3.0)	`ssa_Glen_exponent`	See text and eqn (4) to also set `-ssa_e` if `-ssa_n` changes.

Table 12: For all flow laws, an enhancement factor can be added by a runtime option. For the single-power flow laws in Table 10, the (Glen) exponent can be controlled by a runtime option.

6.3 Surface gradient method

PISM computes surface gradients to determine the "driving stress"

$$(\tau_{d,x}, \tau_{d,y}) = -\rho g H \nabla h,$$

where H is the ice thickness, and $h = H + b$ is the ice surface elevation. The driving stress enters into both the SIA and SSA stress balances, but in the former the driving stress is needed on a staggered grid, while in the latter the driving stress is needed on the regular grid.

Surface gradients are computed by finite differences in several slightly-different ways. There are options for choosing which method to use, but to the best of our knowledge there is no theoretical advice on the best, most robust mechanism. There are three `-gradient` methods in PISM:

`-gradient mahaffy` This most "standard" way computes the surface slope onto the staggered grid for the SIA [73]. It makes $O(\Delta x^2, \Delta y^2)$ errors. For computations of driving stress on the regular grid, centered differencing is used instead.

`-gradient haseloff` This is the default method. It only differs from `mahaffy` at ice-margin locations, where it alters the formula for the slope in cases where an adjacent ice-free bedrock surface elevation is above the ice elevation.

`-gradient eta` In this method we first transform the thickness H by $\eta = H^{(2n+2)/n}$ and then differentiate the sum of the thickness and the bed using centered differences:

$$\nabla h = \nabla H + \nabla b = \frac{n}{(2n+2)} \eta^{(-n-2)/(2n+2)} \nabla \eta + \nabla b.$$

Here b is the bed elevation and h is the surface elevation. This transformation sometimes has the benefits that the surface values of the horizontal velocity and vertical velocity, and the driving stress, are better behaved near the margin. See [20] for technical explanation of this transformation and compare [95]. The actual finite difference schemes applied to compute the surface slope are similar to option `mahaffy`.

6.4 Modeling conservation of energy

In normal use PISM solves the conservation of energy problem within the ice, the thin subglacial layer, and a layer of thermal bedrock. For the ice and the subglacial layer it uses an enthalpy-based scheme [6] which allows the energy to be conserved even when the temperature is at the pressure-melting point.

Ice at the melting point is called "temperate" ice. Part of the thermal energy of temperate ice is in the latent heat of the liquid water stored between the crystals of the temperate ice. Part of the thermal energy of the whole glacier is in the latent heat of the liquid water under the glacier. The enthalpy scheme correctly models these storehouses of thermal energy, and thus it allows polythermal and fully-temperate glaciers to be modeled [5].

The state of the full conservation of energy model includes the 3D `enthalpy` variable plus the 2D `bwat` and `tillwat` subglacial hydrology state variables (subsection 7.2), all of which are seen in output files. The important basal melt rate computation involves all of these energy state variables, because the basal melt rate (`bmelt` in output files) comes from conserving energy across the ice-bedrock layer [6]. Fields `temp`, `liqfrac`, and `temp_pa` seen in output files are all actually diagnostic outputs because all of these can be recovered from the enthalpy and the ice geometry.

Because this part of PISM is just a conservation law, there is little need for the user to worry about controlling it. If desired, however, conservation of energy can be turned off entirely with `-energy none`. The default enthalpy-based conservation of energy model (i.e. `-energy enthalpy`) can be replaced by the temperature-based (i.e. "cold ice") method used in [17] and verified in [18] by setting option `-energy cold`.

The thermal bedrock layer model is turned off by setting `-Mbz 1` (i.e. zero spaces) while it is turned on by choosing a depth and number of points, as in `-Lbz 1000 -Mbz 21`, for example, which gives a layer depth of 1000 m and grid spaces of 50 m (= 1000/20). The input geothermal flux (`bheatflx` in output files) is applied at the bottom of the bedrock thermal layer if such a layer is present and otherwise it is applied at the base of the ice.

6.5 Computing ice age

By default, PISM does not compute the age of the ice because it does not directly impact ice flow when using the default flow laws. It is very easy to turn on. Just set `-age`. A 3D variable `age` will appear in output files. It is read at input if `-age` is set and otherwise it is ignored even if present in the input file. If `-age` is set and the variable `age` is absent in the input file then the initial age is set to zero.

7 Modeling choices: The subglacier

7.1 Controlling basal strength

When using option `-stress_balance ssa+sia`, the SIA+SSA hybrid stress balance, a model for basal resistance is required. This model for basal resistance is based, at least conceptually, on the hypothesis that the ice sheet is underlain by a layer of till [24]. The user can control the parts of this model:

- the so-called sliding law, typically a power law, which relates the ice base (sliding) velocity to the basal shear stress, and which has a coefficient which is or has the units of a yield stress,
- the model relating the effective pressure on the till layer to the yield stress of that layer, and
- the model for relating the amount of water stored in the till to the effective pressure on the till.

This subsection explains the relevant options.

The primary example of `-stress_balance ssa+sia` usage is in section 2 of this Manual, but the option is also used in sections 11.2, 11.3, and 13.

In PISM the key coefficient in the sliding is always denoted as yield stress τ_c, which is `tauc` in PISM output files. This parameter represents the strength of the aggregate material at the base of an ice sheet, a poorly-observed mixture of pressurized liquid water, ice, granular till, and bedrock bumps. The yield stress concept also extends to the power law form, and thus most standard sliding laws can be chosen by user options (below). One reason that the yield stress is a useful parameter is that it can be compared, when looking at PISM output files, to the driving stress (`taud_mag` in PISM output files). Specifically, where `tauc`<`taud_mag` you are likely to see sliding if option `-stress_balance ssa+sia` is used.

A historical note on modeling basal sliding is in order. Sliding can be added directly to a SIA stress balance model by making the sliding velocity a local function of the basal value of the driving stress. Such an SIA sliding mechanism appears in ISMIP-HEINO [23] and in EISMINT II experiment H [85], among other places. This kind of sliding is *not* recommended, as it does not make sense to regard the driving stress as the local generator of flow if the bed is not holding all of that stress [17, 32]. Within PISM, for historical reasons, there is an implementation of SIA-based sliding only for verification test E; see section 10. PISM does *not* support this SIA-based sliding mode in other contexts.

Choosing the sliding law

In PISM the sliding law can be chosen to be a purely-plastic (Coulomb) model, namely,

$$|\boldsymbol{\tau}_b| \le \tau_c \quad \text{and} \quad \boldsymbol{\tau}_b = -\tau_c \frac{\mathbf{u}}{|\mathbf{u}|} \quad \text{if and only if} \quad |\mathbf{u}| > 0. \tag{5}$$

Equation (5) says that the (vector) basal shear stress $\boldsymbol{\tau}_b$ is at most the yield stress τ_c, and that only when the shear stress reaches the yield value can there be sliding. The sliding law can, however, also be chosen to be the power law

$$\boldsymbol{\tau}_b = -\tau_c \frac{\mathbf{u}}{u_{\text{threshold}}^q |\mathbf{u}|^{1-q}}, \tag{6}$$

Option	Description
`-pseudo_plastic`	Enables the pseudo-plastic power law model. If this is not set the sliding law is purely-plastic, so `pseudo_plastic_q` and `pseudo_plastic_uthreshold` are inactive.
`-plastic_reg` (m/a)	Set the value of ϵ regularization of the plastic law, in the formula $\boldsymbol{\tau}_b = -\tau_c \mathbf{u}/\sqrt{\lvert\mathbf{u}\rvert^2 + \epsilon^2}$. The default is 0.01 m/a. This parameter is inactive if `-pseudo_plastic` is set.
`-pseudo_plastic_q`	Set the exponent q in (6). The default is 0.25.
`-pseudo_plastic_uthreshold` (m/a)	Set $u_{\text{threshold}}$ in (6). The default is 100 m/a.

Table 13: Sliding law command-line options

where $u_{\text{threshold}}$ is a parameter with units of velocity (see below). Condition (5) is studied in [99] and [100] in particular, while power laws for sliding are common across the glaciological literature (e.g. see [26, 41]). Notice that the coefficient τ_c in (6) has units of stress, regardless of the power q.

In both of the above equations (5) and (6) we call τ_c the *yield stress*. It corresponds to the variable `tauc` in PISM output files. We call the power law (6) a "pseudo-plastic" law with power q and threshold velocity $u_{\text{threshold}}$. At the threshold velocity the basal shear stress $\boldsymbol{\tau}_b$ has exact magnitude τ_c. In equation (6), q is the power controlled by `-pseudo_plastic_q`, and the threshold velocity $u_{\text{threshold}}$ is controlled by `-pseudo_plastic_uthreshold`. The plastic model (5) is the $q = 0$ case of (6).

See Table 13 for options controlling the choice of sliding law. The purely plastic case is the default; just use `-stress_balance ssa+sia` to turn it on. (Or use `-stress_balance ssa` if a model with no vertical shear is desired.)

> **WARNING!** Options `-pseudo_plastic_q` and `-pseudo_plastic_uthreshold` have no effect if `-pseudo_plastic` is not set.

Equation (6) is a very flexible power law form. For example, the linear case is $q = 1$, in which case if $\beta = \tau_c/u_{\text{threshold}}$ then the law is of the form

$$\boldsymbol{\tau}_b = -\beta \mathbf{u}$$

(The "β" coefficient is also called β^2 in some sources [71, for example].) If you want such a linear sliding law, and you have a value $\beta =$`beta` in $\text{Pa}\,\text{s}\,\text{m}^{-1}$, then use this option combination:

```
-pseudo_plastic -pseudo_plastic_q 1.0 -pseudo_plastic_uthreshold 3.1556926e7 \
  -yield_stress constant -tauc beta
```

This sets $u_{\text{threshold}}$ to 1 $\text{m}\,\text{s}^{-1}$ but using units $\text{m}\,\text{a}^{-1}$.

More generally, it is common in the literature to see power-law sliding relations in the form

$$\boldsymbol{\tau}_b = -C|\mathbf{u}|^{m-1}\mathbf{u},$$

where C is a constant, as for example in sections 11.2 and 11.3. In that case, use this option combination:

```
-pseudo_plastic -pseudo_plastic_q m -pseudo_plastic_uthreshold 3.1556926e7 \
  -yield_stress constant -tauc C
```

Determining the yield stress

Other than setting it to a constant, which only applies in some special cases, the above discussion does not determine the yield stress τ_c. As shown in Table 14, there are two schemes for determining τ_c in a spatially-variable manner:

- `-yield_stress mohr_coulomb` (the default) determines the yields stress by models of till material property (the till friction angle) and of the effective pressure on the saturated till, or
- `-yield_stress constant` allows the yield stress to be supplied as time-independent data, read from the input file.

In normal modelling cases, variations in yield stress are part of the explanation of the locations of ice streams [99]. The default model `-yield_stress mohr_coulomb` determines these variations in time and space. The value of τ_c is determined in part by a subglacial hydrology model, including the modeled till-pore water amount `tillwat` (subsection 7.2), which then determines the effective pressure N_{til} (see below). The value of τ_c is also determined in part by a material property field ϕ =`tillphi`, the "till friction angle". These quantities are related by the Mohr-Coulomb criterion [26]:

$$\tau_c = c_0 + (\tan\phi)\, N_{til}. \tag{7}$$

Here c_0 is called the "till cohesion", whose default value in PISM is zero [99, formula (2.4)] but which can be set by option `-till_cohesion`.

Option combination `-yield_stress constant -tauc X` can be used to fix the yield stress to have value τ_c =`X` at all grounded locations and all times if desired. This is unlikely to be a good modelling choice for real ice sheets.

We find that an effective, though heuristic, way to determine ϕ =`tillphi` in (7) is to make it a function of bed elevation [4, 74, 115]. This heuristic is motivated by hypothesis that basal material with a marine history should be weak [55]. PISM has a mechanism setting ϕ=`tillphi` to be a *piecewise-linear* function of bed elevation. The option is

```
-topg_to_phi phimin,phimax,bmin,bmax
```

Thus the user supplies 4 parameters: $\phi_{\min}$, $\phi_{\max}$, $b_{\min}$, $b_{\max}$, where b stands for the bed elevation. To explain these, we define $M = (\phi_{\max} - \phi_{\min})/(b_{\max} - b_{\min})$. Then

$$\phi(x,y) = \begin{cases} \phi_{\min}, & b(x,y) \le b_{\min}, \\ \phi_{\min} + (b(x,y) - b_{\min})\, M, & b_{\min} < b(x,y) < b_{\max}, \\ \phi_{\max}, & b_{\max} \le b(x,y). \end{cases} \tag{8}$$

Option	Description
`-yield_stress mohr_coulomb`	The default. Use equation (7) to determine τ_c. Only effective if `-stress_balance ssa` or `-stress_balance ssa+sia` is also set.
`-till_cohesion`	Set the value of the till cohesion (c_0) in the plastic till model. The value is a pressure, given in kPa.
`-tauc_slippery_grounding_lines`	If set, reduces the basal yield stress at grounded-below-sea-level grid points one cell away from floating ice or ocean. Specifically, it replaces the normally-computed τ_c from the Mohr-Coulomb relation, which uses the effective pressure from the modeled amount of water in the till, by the minimum value of τ_c from Mohr-Coulomb, i.e. using the effective pressure corresponding to the maximum amount of till-stored water. Does not alter the reported amount of till water, nor does this mechanism affect water conservation.
`-plastic_phi` (degrees)	Use a constant till friction angle. The default is 30°.
`-topg_to_phi` *list of 4 numbers*	Compute ϕ using equation (8).
`-yield_stress constant`	Keep the current values of the till yield stress τ_c. That is, do not update them by the default model using the stored basal melt water. Only effective if `-stress_balance ssa` or `-stress_balance ssa+sia` is also set.
`-tauc`	Directly set the till yield stress τ_c, in units Pa, at all grounded locations and all times. Only effective if used with `-yield_stress constant`, because otherwise τ_c is updated dynamically.

Table 14: Command-line options controlling how yield stress is determined.

It is worth noting that an earth deformation model (see section 7.3) changes $b(x, y)$ =`topg` used in (8), so that a sequence of runs such as

```
pismr -i foo.nc -bed_def lc -stress_balance ssa+sia -topg_to_phi 10,30,-50,0 ... -o bar.nc
pismr -i bar.nc -bed_def lc -stress_balance ssa+sia -topg_to_phi 10,30,-50,0 ... -o baz.nc
```

will use *different* `tillphi` fields in the first and second runs. PISM will print a warning during initialization of the second run:

```
* Initializing the default basal yield stress model...
  option -topg_to_phi seen; creating tillphi map from bed elev ...
PISM WARNING: -topg_to_phi computation will override the 'tillphi' field
              present in the input file 'bar.nc'!
```

Omitting the `-topg_to_phi` option in the second run would make PISM continue with the same `tillphi` field which was set in the first run.

Determining the effective pressure

When using the default option `-yield_stress mohr_coulomb`, the effective pressure on the till N_{til} is determined by the modeled amount of water in the till. Lower effective pressure means that more of the weight of the ice is carried by the pressurized water in the till and thus the ice can slide more easily. That is, equation (7) sets the value of τ_c proportionately to N_{til}. The amount of water in the till is, however, a nontrivial output of the hydrology (subsection 7.2) and conservation-of-energy (section 6.4) submodels in PISM.

Following [108], based on laboratory experiments with till extracted from an ice stream in Antarctica, we use the following parameterization:

$$N_{til} = \delta P_o \, 10^{(e_0/C_c)\left(1-(W_{til}/W_{til}^{max})\right)}. \tag{9}$$

Here P_o is the ice overburden pressure which is determined entirely by the ice thickness and density, W_{til} = `tillwat` is the effective thickness of water in the till, W_{til}^{max} =`hydrology_tillwat_max` is the maximum amount of water in the till (see subsection 7.2), and the remaining parameters are set by options in Table 15. While there is experimental support for the default values of e_0 and C_c, the value of δ =`till_effective_fraction_overburden` should be regarded as uncertain, important, and subject to parameter studies to assess its effect.

FIXME: THIS IS DOC ON EVOLVING CODE: If the `tauc_add_transportable_water` configuration flag is set (either in the configuration file or using the `-tauc_add_transportable_water` option), then the above formula becomes FIXME

7.2 Subglacial hydrology

At the present time, two simple subglacial hydrology models are implemented *and documented* in PISM, namely `-hydrology null` and `-hydrology routing`; see Table 16. In both models, some of the water in the subglacial layer is stored locally in a layer of subglacial till by the hydrology model. In the `routing`

Option	Description
`-till_reference_void_ratio`	$= e_0$ in (9), dimensionless, with default value 0.69 [108]
`-till_compressibility_coefficient`	$= C_c$ in (9), dimensionless, with default value 0.12 [108]
`-till_effective_fraction_overburden`	$= \delta$ in (9), dimensionless, with default value 0.02

Table 15: Command-line options controlling how till effective pressure N_{til} in equation (7) is determined.

Option	Description
`-hydrology null`	The default model with only a layer of water stored in till. Not mass conserving in the map-plane but much faster than `-hydrology routing`. Based on "undrained plastic bed" model of [109]. The only state variable is `tillwat`.
`-hydrology routing`	A mass-conserving horizontal transport model in which the pressure of transportable water is equal to overburden pressure. The till layer remains in the model, so this is a "drained and conserved plastic bed" model. The state variables are `bwat` and `tillwat`.

Table 16: Command-line options to choose the hydrology model.

model water is conserved by horizontally-transporting the excess water (namely `bwat`) according to the gradient of the modeled hydraulic potential. In both hydrology models a state variable `tillwat` is the effective thickness of the layer of liquid water in the till; it is used to compute the effective pressure on the till (see the previous subsection). The pressure of the transportable water `bwat` in the `routing` model does not relate directly to the effective pressure on the till.

See Table 17 for options which apply to all hydrology models. Note that the primary water source for these models is the energy conservation model which computes the basal melt rate `bmelt`. There is, however, also option `-hydrology_input_to_bed_file` which allows the user to *add* water directly into the subglacial layer, in addition to the computed `bmelt` values. Thus `-hydrology_input_to_bed_file` allows the user to model drainage directly to the bed from surface runoff, for example. Also option `-hydrology_bmelt_file` allows the user to replace the computed `bmelt` rate by values read from a file, thereby effectively decoupling the hydrology model from the ice dynamics (esp. conservation of energy).

Option	Description
`-hydrology_bmelt_file` *filename*	Specifies a NetCDF file which contains a time-independent field `bmelt` which has units of water thickness per time. This rate *replaces* the conservation-of-energy computed `bmelt` rate.
`-hydrology_const_bmelt` (m/a)	If `-hydrology_use_const_bmelt` is set then use this to set the constant rate (water thickness per time).
`-hydrology_input_to_bed_file` *filename*	Specifies a NetCDF file which contains a time-dependent field `inputtobed` which has units of water thickness per time. This rate is *added to* the `bmelt` rate.
`-hydrology_input_to_bed_period` (a)	The period, in years, of `-hydrology_input_to_bed_file` data.
`-hydrology_input_to_bed_reference_year` (a)	The reference year for periodizing the `-hydrology_input_to_bed_file` data.
`-hydrology_tillwat_max` (m)	Maximum effective thickness for water stored in till.
`-hydrology_tillwat_decay_rate` (m/a)	Water accumulates in the till at the basal melt rate `bmelt`, minus this rate.
`-hydrology_use_const_bmelt`	Replace the conservation-of-energy basal melt rate `bmelt` with a constant.

Table 17: Subglacial hydrology command-line options which apply to all hydrology models.

The default model `-hydrology null` model

In this model the water is *not* conserved but it is stored locally in the till up to a specified amount; option `-hydrology_tillwat_max` sets that amount. The water is not conserved in the sense that water above the `hydrology_tillwat_max` level is lost permanently. This model is based on the "undrained plastic bed" concept of [109]; see also [17].

In particular, denoting `tillwat` by W_{til}, the till-stored water layer effective thickness evolves by the simple equation

$$\frac{\partial W_{til}}{\partial t} = \frac{m}{\rho_w} - C \tag{10}$$

where m =`bmelt` (kg m^{-2} s^{-1}), ρ_w is the density of fresh water, and C =`hydrology_tillwat_decay_rate`. At all times bounds $0 \le W_{til} \le W_{til}^{max}$ are satisfied.

This `-hydrology null` model has been extensively tested in combination with the Mohr-Coulomb till (subsection 7.1 above) for modelling ice streaming [4, 17, among others].

The mass-conserving `-hydrology routing` model

In this model the water *is* conserved in the map-plane. Water does get put into the till, with the same maximum value `hydrology_tillwat_max`, but excess water is horizontally-transported. An additional state variable `bwat`, the effective thickness of the layer of transportable water, is used by `routing`. This transportable water will flow in the direction of the negative of the gradient of the modeled hydraulic potential. In the `routing` model this potential is calculated by assuming that the transportable subglacial water is at the overburden pressure [105]. Ultimately the transportable water will reach the ice sheet grounding line or ice-free-land margin, at which point it will be lost. The amount that is lost this way is reported to the user.

In this model `tillwat` also evolves by equation (10), but several additional parameters are used in determining how the transportable water `bwat` flows in the model; see Table 18. Specifically, the horizontal subglacial water flux is determined by a generalized Darcy flux relation [24, 103]

$$\mathbf{q} = -k\, W^{\alpha}\, |\nabla\psi|^{\beta-2}\nabla\psi \tag{11}$$

where $\mathbf{q}$ is the lateral water flux, W = `bwat` is the effective thickness of the layer of transportable water, ψ is the hydraulic potential, and k, α, β are controllable parameters (Table 18).

In the `routing` model the hydraulic potential ψ is determined by

$$\psi = P_o + \rho_w g(b + W) \tag{12}$$

where $P_o = \rho_i g H$ is the ice overburden pressure, g is gravity, ρ_i is ice density, ρ_w is fresh water density, H is ice thickness, and b is the bedrock elevation.

For most choices of the relevant parameters and most grid spacings, the `routing` model is at least two orders of magnitude more expensive computationally than the `null` model. This follows directly from the CFL-type time-step restriction on lateral flow of a fluid with velocity on the order of centimeters to meters per second (i.e. the subglacial liquid water `bwat`). (By comparison, much of PISM ice dynamics time-stepping is controlled by the much slower velocity of the flowing ice.) Therefore the user should start

Option	Description
`-hydrology_hydraulic_conductivity` k	$= k$ in formula (11).
`-hydrology_null_strip` (km)	In the boundary strip water is removed and this is reported. This option specifies the width of this strip, which should typically be one or two grid cells.
`-hydrology_gradient_power_in_flux` β	$= \beta$ in formula (11).
`-hydrology_thickness_power_in_flux` α	$= \alpha$ in formula (11).
`-report_mass_accounting`	At each major (ice dynamics) time-step, the duration of hydrology time steps is reported, along with the amount of subglacial water lost to ice-free land, to the ocean, and into the "null strip".

Table 18: Command-line options specific to hydrology model `routing`.

with short runs of order a few model years. The option `-report_mass_accounting` is also recommended, so as to see the time-stepping behavior at `stdout`. Finally, `daily` or even `hourly` reporting for scalar and spatially-distributed time-series to see hydrology model behavior, especially on fine grids (e.g. < 1 km).

7.3 Earth deformation models

The option `-bed_def` [`iso, lc`] turns one of the two available bed deformation models.

The first model `-bed_def iso`, is instantaneous pointwise isostasy. This model assumes that the bed at the starting time is in equilibrium with the load. Then, as the ice geometry evolves, the bed elevation is equal to the starting bed elevation minus a multiple of the increase in ice thickness from the starting time: $b(t, x, y) = b(0, x, y) - f[H(t, x, y) - H(0, x, y)]$. Here f is the density of ice divided by the density of the mantle, so its value is determined by setting the values of `lithosphere_density` and `ice_density` in the configuration file; see subsection 9.6. For an example and verification, see Test H in Verification section.

The second model `-bed_def lc` is much more physical. It is based on papers by Lingle and Clark [68] and Bueler and others [19]. It generalizes and improves the most widely-used earth deformation model in ice sheet modeling, the flat earth Elastic Lithosphere Relaxing Asthenosphere (ELRA) model [42]. It imposes essentially no computational burden because the Fast Fourier Transform is used to solve the linear differential equation [19]. When using this model in PISM, the rate of bed movement (uplift) is stored in the PISM output file and then is used to initialize the next part of the run. In fact, if gridded "observed" uplift data is available, for instance from a combination of actual point observations and/or paleo ice load modeling, and if that uplift field is put in a NetCDF variable with standard name `tendency_of_bedrock_altitude` in the input file, then this model will initialize so that it starts with

the given uplift rate.

Here are minimal example runs to compare these models:

```
$ mpiexec -n 4 pisms -eisII A -y 8000 -o eisIIA_nobd.nc
$ mpiexec -n 4 pisms -eisII A -bed_def iso -y 8000 -o eisIIA_bdiso.nc
$ mpiexec -n 4 pisms -eisII A -bed_def lc -y 8000 -o eisIIA_bdlc.nc
```

Compare the `topg`, `usurf`, and `dbdt` variables in the resulting output files. See also the comparison done in [19].

7.4 Parameterization of bed roughness in the SIA

Schoof [98] describes how to alter the SIA stress balance to model ice flow over bumpy bedrock topgraphy. One computes the amount by which bumpy topography lowers the SIA diffusivity. An internal quantity used in this method is a smoothed version of the bedrock topography. As a practical matter for PISM, this theory improves the SIA's ability to handle bed roughness because it parameterizes the effects of "higher-order" stresses which act on the ice as it flows over bumps. For additional technical description of PISM's implementation, see the *Browser* page "Using Schoof's (2003) parameterized bed roughness technique in PISM".

This parameterization is "on" by default when using `pismr`. There is only one associated option: `-bed_smoother_range` gives the half-width of the square smoothing domain in meters. If zero is given, `-bed_smoother_range 0` then the mechanism is turned off. The mechanism is on by default using executable `pismr`, with the half-width set to 5 km (`-bed_smoother_range 5.0e3`), giving Schoof's recommended smoothing size of 10 km [98].

This mechanism is turned off by default in executables `pisms` and `pismv`.

Under the default setting `-o_size medium`, PISM writes fields `topgsmooth` and `schoofs_theta` from this mechanism. The thickness relative to the smoothed bedrock elevation, namely `topgsmooth`, is the difference between the unsmoothed surface elevation and the smoothed bedrock elevation. It is *only used internally by this mechanism*, to compute a modified value of the diffusivity; the rest of PISM does not use this or any other smoothed bed. The field `schoofs_theta` is a number θ between 0 and 1, with values significantly below zero indicating a reduction in diffusivity, essentially a drag coefficient, from bumpy bed.

8 Modeling choices: Marine ice sheet modeling

PISM is often used to model whole ice sheets surrounded by ocean, with attached floating ice shelves, or smaller regions like outlet glaciers flowing into embayments and possibly generating floating tongues. This section explains the geometry and stress balance mechanisms in PISM that apply to floating ice, at the vertical calving faces of floating ice, or at marine grounding lines. The physics at calving fronts is very different from elsewhere on an ice sheet, because the flow is nothing like the lubrication flow addressed by the SIA, and nor is the physics like the sliding flow in the interior of an ice domain. The needed physics at the calving front can be thought of as boundary condition modifications to the mass continuity equation and to the SSA stress balance equation. The physics of grounding lines are substantially handled by recovering sub-grid information through interpolation.

8.1 PIK options for marine ice sheets

References [2, 67, 115] by the research group of Prof. Anders Levermann at the Potsdam Institute for Climate Impact Research ("PIK"), Germany, describe most of the mechanisms covered in this section. These are all improvements to the grounded, SSA-as-a-sliding law model of [17]. These improvements make PISM an effective Antarctic model, as demonstrated by [39, 74, 114], among other publications. These improvements had a separate existence as the "PISM-PIK" model from 2009–2010, but since PISM stable0.4 are part of PISM itself.

Option	Description
`-cfbc`	apply the stress boundary condition along the ice shelf calving front [115]
`-kill_icebergs`	identify and eliminate free-floating icebergs, which cause well-posedness problems for the SSA stress balance solver [115]
`-part_grid`	allow the ice shelf front to advance by a part of a grid cell, avoiding the development of unphysically-thinned ice shelves [2]
`-part_redist`	additional scheme which makes `-part_grid` conserve mass [2]
`-subgl`	apply interpolation to compute basal shear stress and basal melt near the grounding line [30]
`-no_subgl_basal_melt`	**don't** apply interpolation to compute basal melt near the grounding line if `-subgl` is set [30]
`-pik`	equivalent to option combination "`-cfbc -kill_icebergs -part_grid -part_redist -subgl`"

Table 19: Options which turn on PIK ice shelf front and grounding line mechanisms. A calving law choice is needed in addition to these options.

A summary of options to turn on most of these "PIK" mechanisms is in Table 19. More information on the particular mechanisms is given in sub-sections 8.1.1 through 8.1.4 that follow the Table.

> *When in doubt, PISM users should set option* `-pik` *to turn on all of mechanisms in Table 19. The user should also choose a calving model from Table 21. However, the* `-pik` *mechanisms will not be effective if the non-default FEM stress balance* `-ssa_method fem` *is chosen.*

8.1.1 Stress condition at calving fronts

The vertically integrated force balance at floating calving fronts has been formulated by [75] as

$$\int_{z_s-\frac{\rho}{\rho_w}H}^{z_s+(1-\frac{\rho}{\rho_w})H} \sigma \cdot \mathbf{n}\, dz = \int_{z_s-\frac{\rho}{\rho_w}H}^{z_s} \rho_w g(z-z_s)\, \mathbf{n}\, dz. \tag{13}$$

with $\mathbf{n}$ being the horizontal normal vector pointing from the ice boundary oceanward, σ the *Cauchy* stress tensor, H the ice thickness and ρ and ρ_w the densities of ice and seawater, respectively, for a sea level of z_s. The integration limits on the right hand side of equation (13) account for the pressure exerted by the ocean on that part of the shelf, which is below sea level (bending and torque neglected). The limits on the left hand side change for water-terminating outlet glacier or glacier fronts above sea level according to the bed topography. By applying the ice flow law (section 6.2), equation (13) can be rewritten in terms of strain rates (velocity derivatives), as one does with the SSA stress balance itself.

Note that the discretized SSA stress balance, in the default finite difference discretization chosen by `-ssa_method fd`, is solved with an iterative matrix scheme. If option `-cfbc` is set then, during matrix assembly, those equations which are for fully-filled grid cells along the ice domain boundary have terms replaced according to equation (13), so as to apply the correct stresses [2, 115].

8.1.2 Partially-filled cells at the boundaries of ice shelves

Albrecht et al [2] argue that the correct movement of the ice shelf calving front on a finite-difference grid, assuming for the moment that ice velocities are correctly determined (see below), requires tracking some cells as being partially-filled (option `-part_grid`). If the calving front is moving forward, for example, then the neighboring cell gets a little ice at the next time step. It is not correct to add that little mass as a thin layer of ice which fills the cell's horizontal extent, as that would smooth the steep ice front after a few time steps. Instead the cell must be regarded as having ice which is comparably thick to the upstream cells, but where the ice only partially fills the cell.

Specifically, the PIK mechanism turned on by `-part_grid` adds mass to the partially-filled cell which the advancing front enters, and it determines the coverage ratio according to the ice thickness of neighboring fully-filled ice shelf cells. If option `-part_grid` is used then the PISM output file will have field `Href` which shows the amount of ice in the partially-filled cells as a thickness. When a cell becomes fully-filled, in the sense that the `Href` thickness equals the average of neighbors, then the residual mass is redistributed to neighboring partially-filled or empty grid cells if option `-part_redist` is set.

The stress balance equations determining the velocities are only sensitive to "fully-filled" cells. Similarly, advection is controlled only by values of velocity in fully-filled cells. Adaptive time stepping (specifically:

the CFL criterion) limits the speed of ice front propagation so that at most one empty cell is filled, or one full cell emptied, per time step by the advance or retreat, respectively, of the calving front.

8.1.3 Iceberg removal

Any calving mechanism (see subsection 8.3) removes ice along the seaward front of the ice shelf domain. This can lead to isolated cells either filled or partially-filled with floating ice, or to patches of floating ice (icebergs) fully surrounded by ice free ocean neighbors. This ice is detached from the flowing and partly-grounded ice sheet. That is, calving can lead to icebergs.

In terms of our basic model of ice as a viscous fluid, however, the stress balance for an iceberg is not well-posed because the ocean applies no resistance to balance the driving stress. (See [99].) In this situation the numerical SSA stress balance solver will fail.

Option `-kill_icebergs` turns on the mechanism which cleans this up. This option is therefore generally needed if there is nontrivial calving. The mechanism identifies free-floating icebergs by using a 2-scan connected-component labeling algorithm. It then eliminates such icebergs, with the corresponding mass loss reported as a part of the 2D discharge flux diagnostic (see subsection 9.3).

8.1.4 Sub-grid treatment of the grounding line position

The command-line option `-subgl` turns on a parameterization of the grounding line position based on the "LI" parameterization described in [34] and [30]. With this option PISM computes an extra flotation mask, available as the `gl_mask` output variable, which corresponds to the fraction of the cell that is grounded. Cells that are ice-free or fully floating are assigned the value of 0 while fully-grounded icy cells get the value of 1. Partially grounded cells, the ones which contain the grounding line, get a value between 0 and 1. The resulting field has two uses:

- It is used to scale the basal friction in cells containing the grounding line in order to avoid an abrupt change in the basal friction from the "last" grounded cell to the "first" floating cell. See the source code browser for the detailed description and section 11.3 for an application.

- It is used to adjust the basal melt rate in cells containing the grounding line: in such cells the basal melt rate is set to $M_{b,\text{adjusted}} = \lambda M_{b,\text{grounded}} + (1-\lambda) M_{b,\text{shelf-base}}$, where λ is the value of the flotation mask. Use `-no_subgl_basal_melt` to disable this.

8.2 Flotation criterion, mask, and sea level

The most basic decision about marine ice sheet dynamics made internally by PISM is whether a ice-filled grid cell is floating. That is, PISM applies the "flotation criterion" [115] at every time step and at every grid location to determine whether the ice is floating on the ocean or not. The result is stored in the `mask` variable. The `mask` variable has `pism_intent` = `diagnostic`, and thus it does *not* need to be included in the input file set using the `-i` option.

The possible values of the `mask` are given in Table 20. The mask does not *by itself* determine ice dynamics. For instance, even when ice is floating (mask value `MASK_FLOATING`), the user must turn on

the usual choice for ice shelf dynamics, namely the SSA stress balance, by using options `-stress_balance ssa` or `-stress_balance ssa+sia`.

Mask value	Meaning
0=`MASK_ICE_FREE_BEDROCK`	ice free bedrock
2=`MASK_GROUNDED`	ice is grounded
3=`MASK_FLOATING`	ice is floating (the SIA is never applied; the SSA is applied if the `ssa` or `ssa+sia` stress balance model is selected
4=`MASK_ICE_FREE_OCEAN`	ice-free ocean

Table 20: The PISM mask, in combination with user options, determines the dynamical model.

Assuming that the geometry of the ice is allowed to evolve (which can be turned off by option `-no_mass`), and assuming an ocean exists so that a sea level is used in the flotation criterion (which can be turned off by option `-dry`), then at each time step the mask will be updated.

8.3 Calving

PISM-PIK introduced a physically-based 2D-calving parameterization [67]. This calving parameterization is turned on in PISM by option `-calving eigen_calving`. Average calving rates, c, are proportional to the product of principal components of the horizontal strain rates, $\dot{\epsilon}_{\pm}$, derived from SSA-velocities

$$c = K\,\dot{\epsilon}_{+}\,\dot{\epsilon}_{-} \quad \text{and} \quad \dot{\epsilon}_{\pm} > 0\,. \tag{14}$$

The rate c is in $\mathrm{m\,s^{-1}}$, and the principal strain rates $\dot{\epsilon}_{\pm}$ have units $\mathrm{s^{-1}}$, so K has units $\mathrm{m\,s}$. The constant K incorporates material properties of the ice at the front. It can be set using the `-eigen_calving_K` option or a configuration parameter (`eigen_calving_K` in `src/pism_config.cdl`).

The actual strain rate pattern strongly depends on the geometry and boundary conditions along the confinements of an ice shelf (coast, ice rises, front position). The strain rate pattern provides information in which regions preexisting fractures are likely to propagate, forming rifts (in two directions). These rifts may ultimately intersect, leading to the release of icebergs. This (and other) ice shelf calving models are not intended to resolve individual rifts or calving events, but it produces structurally-stable calving front positions which agree well with observations. Calving rates balance calving-front ice flow velocities on average.

The partially-filled grid cell formulation (subsection 8.1.2) provides a framework suitable to relate the calving rate produced by `eigen_calving` to the mass transport scheme at the ice shelf terminus. Ice shelf front advance and retreat due to calving are limited to a maximum of one grid cell length per (adaptive) time step. The calving rate (velocity) from `eigen_calving` can be used to limit the overall timestep of PISM–thus slowing down all of PISM–by using `-cfl_eigen_calving`. This "CFL"-type time-step limitation is definitely recommended in high-resolution runs which attempt to model calving position accurately. Without this option, under certain conditions where PISM's adaptive time step happens

to be long enough, dendritic structures can appear at the calving front because the calving mechanism cannot "keep up" with the computed calving rate.

PISM also includes three more basic calving mechanisms (Table 21). The option `-calving thickness_calving` is based on the observation that ice shelf calving fronts are commonly thicker than about 150–250 m (even though the physical reasons are not clear yet). Accordingly, any floating ice thinner than H_{cr} is removed along the front, at a rate at most one grid cell per time step. The value of H_{cr} can be set using the `-thickness_calving_threshold` option or the `thickness_calving_threshold` configuration parameter.

Option `-calving float_kill` removes (calves), at each time step of the run, any ice that satisfies the flotation criterion. Use of this option implies that there are no ice shelves in the model at all.

Option `-calving ocean_kill` chooses the calving mechanism removing ice in the "open ocean". It requires the option `-ocean_kill_file` *filename*, which specifies the file containing the ice thickness field `thk`. (This can be the input file specified using `-i`.) Any locations which were ice-free (`thk`$= 0$) and which had bedrock elevation below sea level (`topg`< 0), in the provided data set, are marked as ice-free ocean. The resulting mask is not altered during the run, and is available as diagnostic field `ocean_kill_mask`. At these places any floating ice is removed at each step of the run. Ice shelves can exist in locations where a positive thickness was supplied in the provided data set.

To select several calving mechanisms, use a comma-separated list of keywords mentioned in Table 21:

```
-calving eigen_calving,thickness_calving,ocean_kill
```

8.4 Modeling melange back-pressure

Equation (13) above, describing the stress boundary condition for ice shelves, can be written in terms of velocity components:

$$\begin{aligned} 2\nu H(2u_x + u_y)\mathbf{n}_x &+ 2\nu H(u_y + v_x)\mathbf{n}_y &&= \int_b^h (p_{\text{ice}} - p_{\text{ocean}})dz\,\mathbf{n}_x, \\ 2\nu H(u_y + v_x)\mathbf{n}_x &+ 2\nu H(2v_y + u_x)\mathbf{n}_y &&= \int_b^h (p_{\text{ice}} - p_{\text{ocean}})dz\,\mathbf{n}_y. \end{aligned} \tag{15}$$

Here ν is the vertically-averaged ice viscosity, b is the ice base elevation, h is the ice top surface elevation, and p_{ocean} and p_{ice} are pressures of the column of sea water and ice, respectively.

We call the integral on the right hand side of (15) the "pressure imbalance term". To model the effect of melange [3] on the stress boundary condition, we assume that the melange back-pressure p_{melange} does not exceed $p_{\text{ice}} - p_{\text{ocean}}$. Therefore we introduce $\lambda \in [0, 1]$ (the melange back pressure fraction) such that

$$p_{\text{melange}} = \lambda(p_{\text{ice}} - p_{\text{ocean}}).$$

Then melange pressure is added to the ordinary ocean pressure so that the pressure imbalance term scales with λ:

$$\begin{aligned} \int_b^h (p_{\text{ice}} - (p_{\text{ocean}} + p_{\text{melange}}))\,dz &= \int_b^h (p_{\text{ice}} - (p_{\text{ocean}} + \lambda(p_{\text{ice}} - p_{\text{ocean}})))\,dz \\ &= (1 - \lambda)\int_b^h (p_{\text{ice}} - p_{\text{ocean}})\,dz. \end{aligned} \tag{16}$$

Option	Description
`-calving eigen_calving`	Physically-based calving parameterization [67, 115]. Whereever the product of principal strain rates is positive, the calving rate is proportional to this product.
`-cfl_eigen_calving`	Apply CFL-type criterion to reduce (limit) PISM's time step, according to for eigen-calving rate.
`-eigen_calving_K` (K)	Sets the proportionality parameter K in m s.
`-calving thickness_calving`	Calve all near-terminus ice which is thinner than ice threshold thickness H_{cr}.
`-thickness_calving_threshold` (m)	Sets the thickness threshold H_{cr} in meters.
`-calving float_kill`	All floating ice is calved off immediately.
`-calving ocean_kill`	All ice flowing into grid cells marked as "ice free ocean", according to the ice thickness in the provided file, is calved.
`-ocean_kill_file` *filename*	Sets the file with the `thk` field used to compute maximum ice extent.

Table 21: Options for the four calving models in PISM.

This formula replaces the right hand side of (15).

By default, λ is set to zero, but PISM implements a scalar time-dependent "melange back pressure fraction offset" forcing in which λ can be read from a file. Please see the *PISM's Climate Forcing Manual* for details.

9 Practical usage

9.1 Input and output

PISM is a program that reads NetCDF files and then outputs NetCDF files. Table 22 summarizes command-line options controlling the most basic ways to input and output NetCDF files when starting and ending PISM runs.

Option	Description
`-i` *filename*	Chooses a PISM output file (NetCDF format) to initialize or restart from. See section 4.
`-bootstrap`	Boostrap from the file set using `-i` using heuristics to "fill in" missing fields. See section 4.
`-dontreadSSAvels`	Turns off reading the `ubar_ssa` and `vbar_ssa` velocities saved by a previous run using the `ssa` or `ssa+sia` stress balance (see section 6.1).
`-o` *filename*	Chooses the output file name. Default name is `unnamed.nc`.
`-o_size` ⟨size⟩	Chooses the "size" of the output file to produce. Possible sizes are `none` (*no* output file at all), `small` (only variables necessary to restart PISM), `medium` (the default, includes a few diagnostic quantities) and `big` (writes all the variables mentioned in Tables 29, 30, 31, 32, 33, and 34). Configuration variables `output_medium` and `output_big` list the written variables for those sizes.

Table 22: Basic NetCDF input and output options

Table 23 lists the controls on what is printed to the standard output. Note the `-help` and `-usage` options for getting help at the command line.

The following sections describe more input and output options, especially related to saving quantities during a run, or adding to the "diagnostic" outputs of PISM.

Option	Description
`-help`	Brief descriptions of the many PISM and PETSc options. The run occurs as usual according to the other options. (The option documentation does not get listed if the run didn't get started properly.) Use with a pipe into `grep` to get usefully-filtered information on options, for example "`pisms -help | grep cold`".
`-info`	Gives information about PETSc operations during the run.
`-list_diagnostics`	Prints a list of all available diagnostic outputs (time series and spatial) for the run with the given options. Stops run after printing the list.
`-log_summary`	At the end of the run gives a performance summary and also a synopsis of the PETSc configuration in use.
`-options_left`	At the end of the run shows an options table which will indicate if a user option was not read or was misspelled.
`-usage`	Short summary of PISM executable usage, without listing all the options, and without doing the run.
`-verbose`	Increased verbosity of standard output. Usually given with an integer level; 0,1,2,3,4,5 are allowed. If given without argument then sets level 3, while "`-verbose 2`" is the default (i.e. equivalent to no option). At the extremes, `-verbose 0` produces no stdout at all, `-verbose 1` prints only warnings and a few high priority messages, and `-verbose 5` spews a lot of usually-undesirable stuff. `-verbose 3` output regarding initialization may be useful.
`-version`	Show version numbers of PETSc and PISM.

Table 23: Options controlling PISM's standard output

9.1.1 PISM file I/O performance

When working with fine grids[12], the time PISM spends writing output files, spatially-varying diagnostic files, or backup files can become significant.

It turns out that it is a lot faster to read and write files using the `t,x,y,z` storage order, as opposed to the more convenient (e.g. for NetCDF tools) `t,z,y,x` order. The reason is that PISM uses the `x,y,z` order internally,[13] and therefore writing an array in a different order is an inherently-expensive operation.

You can, however, choose any one of the three supported output orders using the `-o_order` option with one of `xyz`, `yxz`, and `zyx` as the argument.

To transpose dimensions in an existing file, use the `ncpdq` ("permute dimensions quickly") tool from the NCO (*NetCDF Operators*) suite. For example, run

```
$ ncpdq -a t,z,zb,y,x bad.nc good.nc
```

to turn `bad.nc` (with any inconvenient storage order) into `good.nc` using the `t,z,y,x` order.

PISM also supports NetCDF-4 parallel I/O, which gives better performance in high-resolution runs and avoids NetCDF-3 file format limitations. (In a NetCDF-3 file a variable record cannot exceed 4 gigabytes.) Build PISM with parallel NetCDF-4 and use `-o_format netcdf4_parallel` to enable this code.

In addition to `-o_format netcdf4_parallel` and `netcdf3` (default) modes, PISM can be built with PnetCDF for best I/O performance. The option `-o_format pnetcdf` turns "on" PnetCDF I/O code. (PnetCDF seems to be somewhat fragile, though, so use at your own risk.)

9.2 Saving time series of scalar diagnostic quantities

It is also possible to save time-series of certain scalar diagnostic quantities using a combination of the options `-ts_file`, `-ts_times`, and `-ts_vars`. For example,

```
$ pismr -i foo.nc -y 1e4 -o output.nc -ts_file time-series.nc \
        -ts_times 0:1:1e4 -ts_vars ivol,iareag
```

will run for 10000 years, saving total ice volume and grounded ice area to `time-series.nc` *yearly*. See tables 24 for the list of options and tables 25, 26, and 27 for the full list of supported time-series.

Note that, similarly to the snapshot-saving code (section 9.4), this mechanism does not affect adaptive time-stepping. Here, however, PISM will save exactly the number of time-series records requested, *linearly interpolated onto requested times.*

Omitting the `-ts_vars` option makes PISM save *all* available variables, as listed in tables 25, 26, and 27. Because scalar time-series take minimal storage space, compared to spatially-varying data, this is usually a reasonable choice. Run PISM with the `-list_diagnostics` option to see the list of all available time-series.

If the file `foo.nc`, specified by `-ts_file foo.nc`, already exists then by default the existing file will be moved to `foo.nc` and the new time series will go into `foo.nc`. To append the time series onto the end of the existing file, use option `-ts_append`.

[12] For example, resolutions of 2km and higher on the whole-Greenland scale.
[13] This is not likely to change.

PISM buffers time-series data and writes it at the end of the run, once 10000 values are stored, or when an `-extra_file` is saved, whichever comes first. Sending an `USR1` (or `USR2`) signal to a PISM process flushes these buffers, making it possible to monitor the run. (See section 9.8 for more about PISM's signal handling.)

Option	**Description**
`-ts_file` *filename*	Specifies the file to save to.
`-ts_times` *range or list*	Specifies times to save at as a MATLAB-style range $a : \Delta t : b$, a comma-separated list, or a keyword (`hourly`, `daily`, `monthly`, `yearly`). See section 9.3.
`-ts_vars` *comma-separated list*	Comma-separated list of variables, see tables 25, 26, and 27. Omitting this option is equivalent to listing the *all* variables.
`-ts_append`	Append time series to file if it already exists. No effect if file does not yet exist.

Table 24: Command-line options controlling saving scalar time-series

Besides the above information on usage, here are comments on the physical significance of several scalar diagnostics which appear in tables 25, 26, and 27:

- For each variable named ...`_flux`, positive values mean ice sheet mass gain.
- The `sub_shelf_ice_flux` may be non-zero even if `iareaf` (floating ice area) is zero. This is due to the fact that during time-stepping fluxes are computed before calving is applied, and the ice area is computed *after* calving. Hence ice that is calved off experiences top-surface and basal fluxes, but does not contribute to the reported area. This is a small error that approaches zero as the grid is refined. In this case `sub_shelf_ice_flux` should be added to the calving flux during post-processing.
- Ice volume and area are computed and then split among floating and grounded portions: `ivol` $\mapsto$ (`ivolf`, `ivolg`) while `iarea` $\mapsto$ (`iareaf`,`iareag`). The volumes have units m^3 and the areas have units m^2.
- The thermodynamic state of the ice sheet can be assessed, in part, by the amount of cold or temperate ("`temp`") ice. Thus there is another splitting: `ivol` $\mapsto$ (`ivolcold`, `ivoltemp`) and `iarea` $\mapsto$ (`iareacold`,`iareatemp`).
- If a PISM input file contains the `proj4` global attribute with a PROJ.4 string defining the projection then PISM computes corrected cell areas using this information, grid parameters, and the WGS84 reference ellipsoid. This yields areas and volumes with greater accuracy.

- The sea-level-relevant ice volume `slvol` is the total grounded ice volume minus the amount of ice, that, in liquid form, would fill up the regions with bedrock below sea level, if this ice were removed. That is, `slvol` is the sea level rise potential of the ice sheet at that time. The result is reported in sea-level equivalent, i.e. meters of sea level rise.
- Fields `max_diffusivity` and `max_hor_vel` relate to PISM time-stepping. These quantities appear in per-time-step form in the standard output from PISM (i.e. at default verbosity). `max_diffusivity` determines the length of the mass continuity sub-steps for the SIA stress balance (sub-)model. `max_hor_vel` determines the CFL-type restriction for mass continuity and conservation of energy contributions of the SSA stress balance (i.e. sliding) velocity.

Name	**Units**	**Description**
`grounded_basal_ice_flux_cumulative`	kg	cumulative total grounded basal ice flux
`discharge_flux_cumulative`	kg	cumulative discharge (calving etc.) flux
`nonneg_rule_flux_cumulative`	kg	cumulative *numerical* ice flux resulting from enforcing the thk ≥ 0 rule
`sub_shelf_ice_flux_cumulative`	kg	cumulative total sub-shelf ice flux
`surface_ice_flux_cumulative`	kg	cumulative total over ice domain of top surface ice mass flux
`dimassdt`	$kg \,/\, s$	total ice mass rate of change
`discharge_flux`	$kg \,/\, s$	discharge (calving and icebergs) flux
`divoldt`	$m^3 \,/\, s$	total ice volume rate of change
`dt`	*second*	mass continuity time step
`grounded_basal_ice_flux`	$kg \,/\, s$	total, over grounded ice, of basal ice flux
`iarea`	m^2	total ice area
`iareacold`	m^2	ice-covered area where basal ice is cold
`iareaf`	m^2	total floating ice area
`iareag`	m^2	total grounded ice area
`iareatemp`	m^2	ice-covered area where basal ice is temperate
`ienthalpy`	J	total ice enthalpy

Continued in Table 26

Table 25: Scalar time-series supported by PISM, part 1

Name	Units	Description
Continued from Table 25		
`imass`	kg	total ice mass
`ivol`	m^3	total ice volume
`ivolcold`	m^3	total volume of cold ice
`ivolf`	m^3	total floating ice volume
`ivolg`	m^3	total grounded ice volume
`ivoltemp`	m^3	total volume of temperate ice
`max_diffusivity`	$m^2 \,/\, s$	maximum diffusivity
`max_hor_vel`	$m \,/\, s$	maximum (absolute) component of horizontal ice velocity over the grid
`nonneg_rule_flux`	$kg \,/\, s$	*numerical* ice flux resulting from enforcing the thk ≥ 0 rule
`slvol`	m	total sea-level relevant ice **in sea-level equivalent**
`sub_shelf_ice_flux`	$kg \,/\, s$	total sub-shelf ice flux
`surface_ice_flux`	$kg \,/\, s$	total over ice domain of top surface ice mass flux
Continued in Table 27		

Table 26: Scalar time-series supported by PISM, part 2

Name	Units	Description
Continued from Table 26		
`hydro_ice_free_land_loss_cumulative`	kg	cumulative liquid water loss from subglacial hydrology into cells with mask as ice free land
`hydro_ice_free_land_loss`	$ks\ /\ s$	rate of liquid water loss from subglacial hydrology into cells with mask as ice free land
`hydro_ocean_loss_cumulative`	kg	cumulative liquid water loss from subglacial hydrology into cells with mask as ocean
`hydro_ocean_loss`	$kg\ /\ s$	rate of liquid water loss from subglacial hydrology into cells with mask as ocean
`hydro_negative_thickness_gain_cumulative`	kg	cumulative non-conserving liquid water gain from subglacial hydrology transportable water thickness coming out negative during time step, and being projected up to zero
`hydro_negative_thickness_gain`	$kg\ /\ s$	rate of non-conserving liquid water gain from subglacial hydrology transportable water thickness coming out negative during time step, and being projected up to zero
`hydro_null_strip_loss_cumulative`	kg	cumulative liquid water loss from subglacial hydrology into cells inside the null strip
`hydro_null_strip_loss`	$kg\ /\ s$	rate of liquid water loss from subglacial hydrology into cells inside the null strip

Table 27: Scalar time-series supported by PISM (with a mass-conserving subglacial hydrology model), part 3

9.3 Saving time series of spatially-varying diagnostic quantities

Sometimes it is useful to have PISM save a handful of diagnostic *maps* at some interval like every 10 years or even every month. One can use snapshots (section 9.4), but doing so can easily fill your hard-drive because snapshots are complete (i.e. re-startable) model states. Sometimes you want a *subset* of model variables saved frequently in an output file.

Use options `-extra_file`, `-extra_times`, and `-extra_vars` for this. For example,

```
$ pismr -i foo.nc -y 10000 -o output.nc -extra_file extras.nc \
        -extra_times 0:10:1e4 -extra_vars velsurf_mag,velbase_mag
```

will run for 10000 years, saving the magnitude of horizontal velocities at the ice surface and at the base of ice every 10 years. Times are specified using a comma-separated list or a MATLAB-style range. See Table 28 for all the options controlling this feature. Tables 29, 30, 31, 32, 33, and 34 list all the variable choices.

Note that options `-extra_times`, `-save_times`, `-ts_times` take *dates* if a non-trivial calendar is selected. For example,

```
pismr ... -extra_times 10       # every 10 years
pismr ... -extra_times 2days    # every 2 days
pismr ... -calendar gregorian -extra_times 1-1-1:daily:11-1-1 # daily for 10 years
pismr ... -calendar gregorian -extra_times daily -ys 1-1-1 -ye 11-1-1
pismr ... -calendar gregorian -extra_times 2hours -ys 1-1-1 -ye 1-2-1
```

The step in the range specification can have the form `Nunit`, for example `5days`. Units based on "months" and "years" are not supported if a non-trivial calendar is selected.

In addition to specifying a constant step in `-extra_times a:step:b` one can save every hour, day, month, or every year by using `hourly`, `daily`, `monthly` or `yearly` instead of a number; for example

```
$ pismr -i foo.nc -y 100 -o output.nc -extra_file extras.nc \
        -extra_times 0:monthly:100 -extra_vars dHdt
```

will save the rate of change of the ice thickness every month for 100 years. With `-calendar none` (the default), "monthly" means "every $\frac{1}{12}$ of the year", and "yearly" is "every $3.14\cdots \times 10^7$ seconds, otherwise PISM uses month lengths computed using the selected calendar.

It is frequently desirable to save diagnostic quantities at regular intervals for the whole duration of the run; options `-extra_times`, `-ts_times`, and `-save_times` provide a shortcut. For example, use `-extra_times yearly` to save at the end of every year.

This is especially useful when using a climate forcing file to set run duration:

```
$ pismr -i foo.nc -surface given -surface_given_file climate.nc \
        -calendar gregorian -time_file climate.nc \
        -extra_times monthly -extra_file ex.nc -extra_vars thk
```

will save ice thickness at the end of every month while running PISM for the duration of climate forcing data in `climate.nc`.

Times given using `-extra_times` describe the reporting intervals by giving the endpoints of these reporting intervals. The save itself occurs at the end of each interval. This implies, for example, that `0:1:10` will produce 10 records at times 1,...,10 and *not* 11 records.

If the file `foo.nc`, specified by `-extra_file foo.nc`, already exists then by default the existing file will be moved to `foo.nc~` and the new time series will go into `foo.nc`. To append the time series onto the end of the existing file, use option `-extra_append`.

The list of available diagnostic quantities depends on the model setup. For example, a run with only one vertical grid level in the bedrock thermal layer will not be able to save `litho_temp`, an SIA-only run does not use a basal yield stress model and so will not provide `tauc`, etc. To see which quantities are available in a particular setup, use the `-list_diagnostics` option, which prints the list of diagnostics and stops.

The `-extra_file` mechanism modifies PISM's adaptive time-stepping scheme so as to step to, and save at, *exactly* the times requested. By contrast, as noted in subsection 9.2, the `-ts_file` mechanism does not alter PISM's time-steps and instead uses linear interpolation to save at the requested times in between PISM's actual time-steps.

Option	**Description**
`-extra_file` *filename*	Specifies the file to save to; should be different from the output `o` file.
`-extra_times` *range or list*	Specifies times to save at either as a MATLAB-style range $a:\Delta t:b$ or a comma-separated list.
`-extra_vars` *comma-separated list*	Comma-separated list of variables, see tables 29, 30, 31, 32, 33, and 34
`-extra_split`	Save to separate files, similar to `-save_split`
`-extra_append`	Append variables to file if it already exists. No effect if file does not yet exist, and no effect if `-extra_split` is set.

Table 28: Command-line options controlling extra diagnostic output

Name	Units	Description
`age`	s	age of ice
`enthalpy`	$J\ \mathrm{kg}^{-1}$	ice enthalpy (includes sensible heat, latent heat, pressure)
`temp`	K	ice temperature
`cts`	*none*	cts $= E/E_s(p)$; cold-temperate transition surface is cts $= 1$
`liqfrac`	*none*	liquid water fraction in ice (between 0 and 1)
`litho_temp`	K	lithosphere (bedrock) temperature, in `PISMBedThermalUnit`
`pressure`	Pa	ice pressure (in hydrostatic approximation)
`strainheat`	$mW\ /\ m^3$	rate of strain heating in ice (dissipation heating)
`tauxz`	Pa	stress component τ_{xz} (in shallow ice approximation)
`tauyz`	Pa	stress component τ_{yz} (in shallow ice approximation)
`temp`	K	ice temperature
`temp_pa`	*degrees Celsius*	pressure-adjusted ice temperature (degrees above pressure-melting point)

Table 29: Scalar 3D diagnostic quantities

Name	Units	Description
`uvel`	$m\ /\ year$	horizontal velocity of ice in the X direction
`vvel`	$m\ /\ year$	horizontal velocity of ice in the Y direction
`wvel`	$m\ /\ year$	vertical velocity of ice, relative to geoid
`wvel_rel`	$m\ /\ year$	vertical velocity of ice, relative to base of ice directly below

Table 30: Vector 3D diagnostic quantities

Name	Units	Description
`bedtoptemp`	K	temperature of top of bedrock thermal layer
`beta`	$Pa\ s\ /\ m$	basal drag coefficient
`bfrict`	$W\ /\ m^2$	basal frictional heating
`bheatflx`	$W\ /\ m^2$	upward geothermal flux at bedrock surface
`bmelt`	$m\ /\ year$	ice basal melt rate in ice thickness per time
`bwat`	m	effective thickness of subglacial melt water
`bwprel`	1	pressure of transportable water in subglacial layer as fraction of the overburden pressure
`bwp`	Pa	subglacial (pore) water pressure
`cell_area`	m^2	cell areas
`climatic_ mass_balance_ cumulative`	$kg\ /\ m^2$	cumulative surface mass balance (accumulation/ablation)
`climatic_ mass_balance`	$kg\ /\ m^2\ /\ year$	surface mass balance (accumulation/ablation) rate
`dHdt`	$m\ /\ year$	ice thickness rate of change
`dbdt`	$m\ /\ year$	bedrock uplift rate
`deviatoric_ stresses`	Pa	2D deviatoric stresses (this saves 3 fields: in the X direction, in the Y direction, and the shear stress)
`diffusivity_ staggered`	$m^2\ /\ s$	diffusivity of SIA mass continuity equation, on the staggered grid
`diffusivity`	$m^2\ /\ s$	diffusivity of SIA mass continuity equation
`discharge_ flux_ cumulative`	Gt	cumulative ice discharge (calving) flux
`effbwp`	Pa	effective pressure of transportable water in subglacial layer (overburden pressure minus water pressure)
`enthalpybase`	$J\ /\ kg$	ice enthalpy at the base of ice
`enthalpysurf`	$J\ /\ kg$	ice enthalpy at 1m below the ice surface

Continued in Table 32

Table 31: Scalar 2D diagnostic quantities, part 1

Name	Units	Description
		Continued from Table 31
`floating_basal_flux_cumulative`	Gt	cumulative basal flux into the ice in floating areas
`flux_divergence`	m / $year$	flux divergence
`flux_mag`	m^2 / $year$	magnitude of vertically-integrated horizontal flux of ice
`grounded_basal_flux_cumulative`	Gt	cumulative basal flux into the ice in grounded areas
`hardav`	$Pa\,s^{1/n}$	vertical average of ice hardness
`hydrobmelt`	m / $year$	the version of `bmelt` seen by the hydrology model
`hydroinput`	m / $year$	total water input into subglacial hydrology layer
`hydrovelbase_mag`	m / s	the version of `velbase_mag` seen by the 'distributed' hydrology model
`ice_surface_temp`	K	annual average ice surface temperature, below firn processes
`lat`	*degrees north*	latitude
`lon`	*degrees east*	longitude
`mask`	*none*	grounded/dragging/floating integer mask
`nonneg_flux_cumulative`	Gt	cumulative numerical flux created by enforcing non-negativity of ice thickness
`nuH`	*kPa s m*	ice thickness times effective viscosity, on the staggered grid
`proc_ice_area`	*none*	number of cells containing ice in a processor's domain
`rank`	*none*	processor rank
`schoofs_theta`	*1*	multiplier θ in [98]
`shelfbmassflux`	m / $year$	ice mass flux from ice shelf base (positive flux is loss from ice shelf)
`shelfbtemp`	*Kelvin*	absolute temperature at ice shelf base
`strain_rates`	*1/s*	eigenvalues of the horizontal, vertically-integrated strain rate tensor
		Continued in Table 33

Table 32: Scalar 2D diagnostic quantities, part 2

Name	Units	Description
		Continued from Table 32
`taub_mag`	*Pa*	magnitude of the basal shear stress
`tauc`	*Pa*	yield stress for basal till (plastic or pseudo-plastic model)
`taud_mag`	*Pa*	magnitude of the driving stress at the base of ice
`tempbase`	*K*	ice temperature at the base of ice
`tempicethk_basal`	*m*	thickness of the basal layer of temperate ice
`tempicethk`	*m*	temperate ice thickness (total column content)
`temppabase`	*Celsius*	pressure-adjusted ice temperature at the base of ice
`tempsurf`	*K*	ice temperature at 1m below the ice surface
`thksmooth`	*m*	thickness relative to smoothed bed elevation in [98]
`thk`	*m*	land ice thickness
`tillphi`	*degrees*	friction angle for till under grounded ice sheet
`topgsmooth`	*m*	smoothed bed elevation in [98]
`topg`	*m*	bedrock surface elevation
`usurf`	*m*	ice upper surface elevation
`velbar_mag`	*m / year*	magnitude of vertically-integrated horizontal velocity of ice
`velbase_mag`	*m / year*	magnitude of horizontal velocity of ice at base of ice
`velsurf_mag`	*m / year*	magnitude of horizontal velocity of ice at ice surface
`wallmelt`	*m / year*	wall melt into subglacial hydrology layer from (turbulent) dissipation of energy in transportable water

Table 33: Scalar 2D diagnostic quantities, part 3

Name	Units	Description
`h_x`	*none*	the x-component of the surface gradient, on the staggered grid
`h_y`	*none*	the y-component of the surface gradient, on the staggered grid
`taub`	*Pa*	basal shear stress, X and Y components. See also `taub_mag`.
`taud`	*Pa*	driving stress at the base of ice, X and Y components. See also `taud_mag`.
`flux`	m^2 / *year*	vertically-integrated horizontal flux of ice, X and Y components. See also `flux_mag`.
`velbar`	*m* / *year*	vertical mean of horizontal ice velocity in the X and Y directions. See also `velbar_mag`.
`velbase`	*m* / *year*	horizontal velocity of ice at the base of ice in the X and Y directions. See also `velbase_mag`.
`velsurf`	*m* / *year*	horizontal velocity of ice at ice surface in the X and Y directions. See also `velsurf_mag`.
`wvelbase`	*m* / *year*	vertical velocity of ice at the base of ice, relative to the geoid
`wvelsurf`	*m* / *year*	vertical velocity of ice at ice surface, relative to the geoid
`bwatvel`	*m* / *year*	velocity of water in subglacial layer, on the staggered grid

Table 34: Vector 2D diagnostic quantities (vector)

9.4 Saving re-startable snapshots of the model state

Sometimes you want to check the model state every 1000 years, for example. One possible solution is to run PISM for a thousand years, have it save all the fields at the end of the run, then restart and run for another thousand, and etc. This forces the adaptive time-stepping mechanism to stop *exactly* at multiples of 1000 years, which may be desirable in some cases.

If saving exactly at specified times is not critical, then use the `-save_file` and `-save_times` options. For example,

```
$ pismr -i foo.nc -y 10000 -o output.nc -save_file snapshots.nc \
        -save_times 1000:1000:10000
```

starts a PISM evolution run, initializing from `foo.nc`, running for 10000 years and saving snapshots to `snapshots.nc` at the first time-step after each of the years 1000, 2000, ..., 10000.

We use a MATLAB-style range specification, $a : \Delta t : b$, where $a, \Delta t, b$ are in years. The time-stepping scheme is not affected, but as a consequence we do not guarantee producing the exact number of snapshots requested if the requested save times have spacing comparable to the model time-steps. This is not a problem in the typical case in which snapshot spacing is much greater than the length of a typical time step.

It is also possible to save snapshots at intervals that are not equally-spaced by giving the `-save_times` option a comma-separated list. For example,

```
$ pismr -i foo.nc -y 10000 -o output.nc -save_file snapshots.nc \
        -save_times 1000,1500,2000,5000
```

will save snapshots on the first time-step after years 1000, 1500, 2000 and 5000. The comma-separated list given to the `-save_times` option can be at most 200 numbers long.

If `snapshots.nc` was created by the command above, running

```
$ pismr -i snapshots.nc -y 1000 -o output_2.nc
```

will initialize using the last record in the file, at about 5000 years. By contrast, to restart from 1500 years (for example) it is necessary to extract the corresponding record using `ncks`

```
$ ncks -d t,1500years snapshots.nc foo.nc
```

and then restart from `foo.nc`. Note that `-d t,N` means "extract the N-th record" (counting from zero). So, this command is equivalent to

```
$ ncks -d t,1 snapshots.nc foo.nc
```

Also note that the second snapshot will probably be *around* 1500 years and `ncks` handles this correctly: it takes the record closest to 1500 years.

By default re-startable snapshots contain only the variables needed for restarting PISM. Use the command-line option `-save_size` to change what is saved.

Another possible use of snapshots is for restarting runs on a batch system which kills jobs which go over their allotted time. Running PISM with options `-y 1500 -save_times 1000:100:1400` would mean

that if the job is killed before completing the whole 1500 year run, we can restart from near the last multiple of 100 years. Restarting with option `-ye` would finish the run on the desired year.

When running PISM on such a batch system it is also possible to save re-startable snapshots at equal wall-clock time (as opposed to model time) intervals by adding the "`-backup_interval` hours" option.

A note of caution: if the wall-clock limit is equal to N times backup interval for a whole number N PISM will likely get killed while writing the last backup.

It is also possible to save snapshots to separate files using the `-save_split` option. For example, the run above can be changed to

```
$ pismr -i foo.nc -y 10000 -o output.nc -save_file snapshots \
        -save_times 1000,1500,2000,5000 -save_split
```

for this purpose. This will produce files called `snapshots-`*year*`.nc`. This option is generally faster if many snapshots are needed, apparently because of the time necessary to reopen a large file at each snapshot when `-save_split` is not used. Note that tools like NCO and `ncview` usually behave as desired with wildcards like "`snapshots-*.nc`".

Table 35 lists the options related to saving snapshots of the model state.

Option	Description
`-save_file` *filename*	Specifies the file to save to.
`-save_times` *range or list*	Specifies times at which to save snapshots, by either a MATLAB-style range $a : \Delta t : b$ or a comma-separated list.
`-save_split`	Separate the snapshot output into files named `snapshots-`*year*`.nc`. Faster if you are saving more than a dozen or so snapshots.
`-save_size` [none,small,medium,big]	similar to `o_size`, changes the "size" of the file (or files) written; the default is "small"

Table 35: Command-line options controlling saving snapshots of the model state.

9.5 Run-time diagnostic viewers

Basic graphical views of the changing state of a PISM ice model are available at the command line by using options listed in table 36. All the quantities listed in tables 29, 30, 31, 32, 33, and 34 are available. Additionally, a couple of diagnostic quantities are *only* available as run-time viewers; these are shown in table 37.

Note that (for performance and implementation reasons) map viewers are transposed.

The option `-view_map` shows map-plane views of 2D fields and surface and basal views of 3D fields (see tables 29, 30, 31, 32, 33, and 34); for example:

Option	Description
`-view_map` *comma-separated list*	Turns on map-plane views of one or several variables, see tables 29, 30, 31, 32, 33, and 34
`-view_size` number	desired viewer size, in pixels
`-display`	The option `-display :0` seems to frequently be needed to let PETSc use Xwindows when running multiple processes. *It must be given as a* final *option, after all the others.*

Table 36: Options controlling run-time diagnostic viewers

```
$ pismr -i input.nc -y 1000 -o output.nc -view_map thk,tempsurf
```

shows ice thickness and ice temperature at the surface.

One can also expose all the linear systems being solved in ice columns by using `-view_sys` along with the map-plane location specified using `-id XX` and `-jd YY`. For example, if the enthalpy formulation is in use then a MATLAB-format text file `enth_iXX_jYY.m` will be generated at each time step. This text file can be run as a script in MATLAB. The linear system will be stored in a matrix, a right-hand-side vector, and the solution in a solution vector. This way these linear systems can be checked for debugging purposes.

Variable name or an option	Description
`-ssa_view_nuh`	log base ten of `nuH`, only available if the finite-difference SSA solver is active. You can adjust the viewer size with `-ssa_nuh_viewer_size` *number*.
`-ksp_monitor_draw`	Iteration monitor for the Krylov subspace routines (KSP) in PETSc. Residual norm versus iteration number.

Table 37: Special run-time-only diagnostic viewers

9.6 PISM's configuration flags and parameters, and how to change them

PISM's behavior depends on values of many flags and physical parameters (see PISM Source Code Browser for details). Most of parameters have default values[14] which are read from the configuration file `pism_config.nc` in the `lib` sub-directory.

It is possible to run PISM with an alternate configuration file using the `-config` *filename* command-line option:

[14]For `pismr`, grid parameters Mx, My, Mz, Mbz, Lz, Lbz, that must be set at bootstrapping, are exceptions.

```
$ pismr -i foo.nc -y 1000 -config my_config.nc
```

The file `my_config.nc` has to contain *all* of the flags and parameters present in `pism_config.nc`.

The list of parameters is too long to include here; please see the PISM Source Code Browser for an automatically-generated table describing them.

Some command-line options *set* configuration parameters; some PISM executables have special parameter defaults. To examine what parameters were used in a particular run, look at the attributes of the `pism_config` variable in a PISM output file.

Managing parameter studies

Keeping all PISM output files in a parameter study straight can be a challenge. If the parameters of interest were controlled using command-line options then one can use `ncdump -h` and look at the `history` global attribute.

Alternatively, one can change parameter values by using an "overriding" configuration file. The `-config_override` *filename* command-line option provides this alternative. A file used with this option can have a subset of the configuration flags and parameters present in `pism_config.nc`. Moreover, PISM adds the `pism_config` variable with values used in a run to the output file, making it easy to see which parameters were used.

Here's an example. Suppose we want to compare the dynamics of an ice-sheet on Earth to the same ice-sheet on Mars, where the only physical change was to the value of the acceleration due to gravity. Running

```
$ pismr -i input.nc -y 1e5 -o earth.nc <other PISM options>
```

produces the "Earth" result, since PISM's defaults correspond to this planet. Next, we create `mars.cdl` containing the following:

```
netcdf mars {
    variables:
    byte pism_overrides;
    pism_overrides:standard_gravity = 3.728;
    pism_overrides:standard_gravity_doc = "m s-2; standard gravity on Mars";
}
```

Notice that the variable name is `pism_overrides` and not `pism_config` above. Now

```
$ ncgen -o mars_config.nc mars.cdl
$ pismr -i input.nc -y 1e5 -config_override mars_config.nc -o mars.nc <other PISM options>
```

will create `mars.nc`, the result of the "Mars" run. Then we can use `ncdump` to see what was different about `mars.nc`:

```
$ ncdump -h earth.nc | grep pism_config: > earth_config.txt
$ ncdump -h mars.nc | grep pism_config: > mars_config.txt
$ diff -U 1 earth_config.txt mars_config.txt
--- earth_config.txt 2015-05-08 12:44:43.000000000 -0800
+++ mars_config.txt 2015-05-08 12:44:51.000000000 -0800
```

```
@@ -734,3 +734,3 @@
                pism_config:ssafd_relative_convergence_units = "1" ;
-               pism_config:standard_gravity_doc = "acceleration due to gravity on Earth geoid" ;
+               pism_config:standard_gravity_doc = "m s-2; standard gravity on Mars" ;
                pism_config:standard_gravity_type = "scalar" ;
@@ -1057,3 +1057,3 @@
                pism_config:ssafd_relative_convergence = 0.0001 ;
-               pism_config:standard_gravity = 9.81 ;
+               pism_config:standard_gravity = 3.728 ;
                pism_config:start_year = 0. ;
```

Saving PISM's configuration for post-processing

In addition to saving `pism_config` in the output file, PISM automatically adds this variable to all files it writes (snap shots, time series of scalar and spatially-varying diagnostic quantities, and backups). This may be useful for post-processing and analysis of parameter sties as the user has easy access to all configuration options, model choices, etc., without the need to keep run scripts around.

9.7 Regridding

It is common to want to interpolate a coarse grid model state onto a finer grid or vice versa. For example, one might want to do the EISMINT II experiment on the default grid, producing output `foo.nc`, but then interpolate both the ice thickness and the temperature onto a finer grid. The basic idea of "regridding" in PISM is that one starts over from the beginning on the finer grid, but one extracts the desired variables stored in the coarse grid file and interpolates these onto the finer grid before proceeding with the actual computation.

The transfer from grid to grid is reasonably general—one can go from coarse to fine or vice versa in each dimension x, y, z—but the transfer must always be done by *interpolation* and never *extrapolation*. (An attempt to do the latter will always produce a PISM error.)

Such "regridding" is done using the `-regrid_file` *filename* and `-regrid_vars` *comma-separated list* commands as in this example:

```
$  pisms -eisII A -Mx 101 -My 101 -Mz 201 -y 1000 \
         -regrid_file foo.nc -regrid_vars thk,temp -o bar.nc
```

By specifying regridded variables "`thk,temp`", the ice thickness and temperature values from the old grid are interpolated onto the new grid. Here one doesn't need to regrid the bed elevation, which is set identically zero as part of the EISMINT II experiment A description, nor the ice surface elevation, which is computed as the bed elevation plus the ice thickness at each time step anyway.

A slightly different use of regridding occurs when "bootstrapping", as described in section 4 and illustrated by example in section 2.

See table 38 for the regriddable variables using `-regrid_file`. Only model state variables are regriddable, while climate and boundary data generally are not explicitly regriddable. (Bootstrapping, however, allows the same general interpolation as this explicit regrid.)

Name	Description
age	age of ice
bwat	effective thickness of subglacial melt water
bmelt	basal melt rate
dbdt	bedrock uplift rate
litho_temp	lithosphere (bedrock) temperature
mask	grounded/dragging/floating integer mask, see section 8.2
temp	ice temperature
thk	land ice thickness
topg	bedrock surface elevation
enthalpy	ice enthalpy

Table 38: Regriddable variables. Use `-regrid_vars` with these names.

Here is another example: suppose you have an output of a PISM run on a fairly coarse grid (stored in `foo.nc`) and you want to continue this run on a finer grid. This can be done using `-regrid_file` along with `-bootstrap`:

```
$ pismr -i foo.nc -bootstrap -Mx 201 -My 201 -Mz 21 -Lz 4000 \
        -regrid_file foo.nc -regrid_vars litho_temp,enthalpy -y 100 -o bar.nc \
        -surface constant
```

In this case all the model-state 2D variables present in `foo.nc` will be interpolated onto the new grid during bootstrapping, which happens first, while three-dimensional variables are filled using heuristics mentioned in section 4. Then temperature in bedrock (`litho_temp`) and ice enthalpy (`enthalpy`) will be interpolated from `foo.nc` onto the new grid during the regridding stage, overriding values set at the bootstrapping stage. All of this, bootstrapping and regridding, occurs before the first time step.

9.8 Signals, to control a running PISM model

Ice sheet model runs sometimes take a long time, so the state of a run may need checking. Sometimes the run needs to be stopped, but with the possibility of restarting. PISM implements these behaviors using "signals" from the POSIX standard, included in Linux and most flavors of Unix. Table 39 summarizes how PISM responds to signals. A convenient form of `kill`, for Linux users, is `pkill` which will find processes by executable name. Thus "`pkill -USR1 pismr`" might be used to send all PISM processes the same signal, avoiding an explicit list of *pids*.

Here is an example. Suppose we start a long verification run in the background, with standard out redirected into a file:

```
pismv -test G -Mz 101 -y 1e6 -o testGmillion.nc >> log.txt &
```

Command	Signal	PISM behavior
`kill -KILL` *pids*	`SIGKILL`	Terminate with extreme prejudice. PISM cannot catch it and no state is saved.
`kill -TERM` *pids*	`SIGTERM`	End process(es), but save the last model state in the output file, using `-o` name or default name as normal. Note that the `history` string in the output file will contain an "`EARLY EXIT caused by signal SIGTERM`" indication.
`kill -USR1` *pids*	`SIGUSR1`	Process(es) will continue after saving the model state at the end of the current time step, using name "`pismX-`*year*`.nc`". Time-stepping is not altered. Also flushes time-series output buffers.
`kill -USR2` *pids*	`SIGUSR2`	Just flush time-series output buffers.

Table 39: Signalling running PISM processes. "*pids*" stands for list of all identifiers of the PISM processes.

This run gets a Unix process id, which we assume is "8920". (Get it using `ps` or `pgrep`.) If we want to observe the run without stopping it we send the `USR1` signal:

```
kill -USR1 8920
```

(With `pkill` one can usually type "`pkill -usr1 pismv`".) Suppose it happens that we caught the run at year 31871.5. Then, for example, a NetCDF file `pismv-31871.495.nc` is produced. Note also that in the standard out log file `log.txt` the line

```
caught signal SIGUSR1:  Writing intermediate file ... and flushing time series.
```

appears around that time step. Suppose, on the other hand, that the run needs to be stopped. Then a graceful way is

```
kill -TERM 8920
```

because the model state is saved and can be inspected.

9.9 Understanding adaptive time-stepping

At each time step the PISM standard output includes "flags" and then a summary of the model state using a few numbers. A typical example is

```
v$Eh  diffusivity (dt=0.83945 in 2 substeps; av dt_sub_mass_cont=0.41972)
S -124791.571:  3.11640   2.25720      3.62041    18099.93737
y  SSA:     3 outer iterations, ~17.0 KSP iterations each
```

The characters "`v$Eh`" at the beginning of the flags line, the first line in the above example, give a very terse description of which physical processes were modeled in that time step. Here "`v`" means that a stress balance was solved to compute the velocity. Then the enthalpy was updated ("`E`") and the ice thickness and surface elevation were updated ("`h`"). The rest of the flags line looks like

"`diffusivity (dt=0.83945 in 2 substeps; av dt_sub_mass_cont=0.41972)`"

Recall that the PISM time step is determined by an adaptive mechanism. Stable mass conservation and conservation of energy solutions require such an adaptive time-stepping scheme [18]. The first character we see here, namely "`diffusivity`", is the adaptive-timestepping "reason" flag. See Table 40. We also see that there was a major time step of 0.83945 model years divided into 2 substeps of about 0.42 years. The `-skip` option enables this mechanism, while `-skip_max` sets the maximum number of such substeps. The adaptive mechanism may choose to take fewer substeps than `-skip_max` so as to satisfy certain numerical stability criteria, however.

The second line in the above, the line which starts with "`S`", is the summary. Its format, and the units for these numbers, is simple and is given by a couple of lines printed near the beginning of the standard output for the run:

```
P       YEAR:       ivol      iarea  max_diffusivity  max_hor_vel
U      years   10^6_km^3  10^6_km^2         m^2 s^-1       m/year
```

That is, in each summary we have the total ice volume, total ice area, maximum diffusivity (of the SIA mass conservation equation), and maximum horizontal velocity (i.e. $\max(\max(|u|), \max(|v|))$).

The third line of the above example shows that the SSA stress balance was solved. Information on the number of nonlinear (outer) and linear (inner) iterations is provided [17].

9.10 PETSc options for PISM users

All PETSc programs including PISM accept command line options which control how PETSc distributes jobs among parallel processors, how it solves linear systems, what additional information it provides, and so on. The PETSc manual [7] is the complete reference on these options. We list some here that are useful to PISM users. They can be mixed in any order with PISM options.

Both for PISM and PETSc options, there are ways of avoiding the inconvenience of long commands with many runtime options. Obviously, and as illustrated by examples in the previous sections, shell scripts can be set up to run PISM. But PETSc also provides two mechanisms to give runtime options without retyping at each run command.

First, the environment variable `PETSC_OPTIONS` can be set. For example, a sequence of runs might need the same refined grid, and you might want to know if other options are read, ignored, or misspelled. Set (in bash):

```
export PETSC_OPTIONS="-Mx 101 -My 101 -Mz 51 -options_left"
```

The runs

```
$ pismv -test F -y 100
$ pismv -test G -y 100
```

then have the same refined grid in each run, and the runs report on which options were read.

PISM output	Active adaptive constraint or PISM sub-system that limited time-step size
`3D CFL`	three-dimensional CFL for temperature/age advection [18]
`diffusivity`	diffusivity for SIA mass conservation [18, 48]
`end of the run`	end of prescribed run time
`fixed`	`-dt_force` set; generally option `-dt_force`, which overrides the adaptive scheme; *should not be used*
`max`	maximum allowed Δt applies; set with `-max_dt`
`internal (derived class)`	maximum Δt was temporarily set by a derived class
`2D CFL`	2D CFL for mass conservation in SSA regions (upwinded; [17])
`-ts_... reporting`	the `-ts_times` option and the `ts_force_output_times` configuration flag; see section 9.2
`-extra_... reporting`	the `-extra_times` option; see section 9.3
`surface`	a surface or an atmosphere model
`ocean`	an ocean model
`hydrology`	a hydrology model stability criterion, see section 7.2
`BTU`	time-the bedrock thermal layer model, see section 6.4
`eigencalving`	the eigen-calving model, see section 8.3

Table 40: Meaning of the adaptive time-stepping "reason" flag in the standard output flag line.

Alternatively, the file `.petscrc` is always read, if present, from the directory where PISM (i.e. the PETSc program) is started. It can have a list of options, one per line. In theory, these two PETSc mechanisms (`PETSC_OPTIONS` and `.petscrc`) can be used together.

Now we address controls on how PETSc solves systems of linear equations, which uses the PETSc "KSP" component (Krylov methods). Such linear solves are needed each time the nonlinear SSA stress balance equations are used (e.g. with the option `-stress_balance ssa -ssa_method fd`).

Especially for solving the SSA equations with high resolution on multiple processors, it is recommended that the option `-ssafd_ksp_rtol` be set lower than its default value of 10^{-5}. For example,

```
$  mpiexec -n 8 ssa_testi -Mx 3 -My 769 -ssa_method fd
```

may fail to converge on a certain machine, but adding "`-ssafd_ksp_rtol 1e-10`" works fine.

There is also the question of solver *type*, using option `-ssafd_ksp_type`. Based on one processor evidence from `ssa_testi`, the following are possible choices in the sense that they work and allow

Option	Description
`-adapt_ratio`	Adaptive time stepping ratio for the explicit scheme for the mass balance equation.
`-max_dt` (years)	The maximum time-step in years. The adaptive time-stepping scheme will make the time-step shorter than this as needed for stability, but not longer.
`-dt_force` (years)	The time step (in years) to take, overriding the adaptive scheme. *Not recommended.*
`-skip`	Enables time-step skipping, see below.
`-skip_max`	Number of mass-balance steps, including SIA diffusivity updates, to perform before temperature, age, and SSA stress balance computations are done. This is only effective if the time step is being limited by the diffusivity time step restriction associated to mass continuity using the SIA. The maximum recommended value for `-skip_max` is, unfortunately, dependent on the context. The temperature field should be updated when the surface changes significantly, and likewise the basal sliding velocity if it comes (as it should) from the SSA calculation.
`-timestep_hit_multiples` (years)	Hit multiples of the number of model years specified. For example, if stability criteria require a time-step of 11 years and the `-timestep_hit_multiples 3` option is set, PISM will take a 9 model year long time step. This can be useful to enforce consistent sampling of periodic climate data.

Table 41: Options controlling time-stepping

convergence at some reasonable rate: `cg`, `bicg`, `gmres`, `bcgs`, `cgs`, `tfqmr`, `tcqmr`, and `cr`. It appears `bicg`, `gmres`, `bcgs`, and `tfqmr`, at least, are all among the best. The default is `gmres`.

Actually the KSP uses preconditioning. This aspect of the solve is critical for parallel scalability, but it gives results which are dependent on the number of processors. The preconditioner type can be chosen with `-ssafd_pc_type`. Several choices are possible, but for solving the ice stream and shelf equations we recommend only `bjacobi`, `ilu`, and `asm`. Of these it is not currently clear which is fastest; they are all about the same for `ssa_testi` with high tolerances (e.g. `-ssa_rtol 1e-7 -ssafd_ksp_rtol 1e-12`). The default (as set by PISM) is `bjacobi`. To force no preconditioning, which removes processor-number-dependence of results but may make the solves fail, use `-ssafd_pc_type none`.

For the full list of PETSc options controlling the SSAFD solver, run

```
$ ssa_testi -ssa_method fd -help | grep ssafd_ | less
```

9.11 Utility and test scripts

In the `test/` and `util/` subdirectories of the PISM directory the user will find some python scripts and one Matlab script, listed in Table 42. The python scripts are all documented at the *Packages* tab on the PISM Source Code Browser. The python scripts all take option `-help`.

Script	Purpose
`test/vfnow.py`	Organizes the process of verifying PISM. Specifies standard refinement paths for each of the tests (section 10).
`test/vnreport.py`	Automates the creation of convergence graphs like figures 18– 22.
`util/fill_missing.py`	Uses an approximation to Laplace's equation $\nabla^2 u = 0$ to smoothly replace missing values in a two-dimensional NetCDF variable. The "hole" is filled with an average of the boundary non-missing values. Depends on `netcdf4-python` and `scipy` Python packages.
`util/flowline.py`	See subsection 9.12.
`util/flowlineslab.py`	See subsection 9.12.
`util/check_stationarity.py`	Evaluate stationarity of a variable in a PISM `-ts_file` output.
`util/nc2cdo.py`	Makes a netCDF file ready for Climate Data Operators (CDO).
`util/nc2mat.py`	Reads specified variables from a NetCDF file and writes them to an output file in the MATLAB binary data file format `.mat`, supported by MATLAB version 5 and later. Depends on `netcdf4-python` and `scipy` Python packages.
`util/nccmp.py`	A script comparing variables in a given pair of NetCDF files; used by PISM software tests.
`util/pism_config_editor.py`	Makes modifying or creating PISM configuration files easier.
`util/pism_matlab.m`	An example MATLAB script showing how to create a simple NetCDF file PISM can bootstrap from.
`util/PISMNC.py`	Used by many Python example scripts to generate a PISM-compatible file with the right dimensions and time-axis.

Table 42: Some scripts which help in using PISM.

9.12 Using PISM for flow-line modeling

As described in sections 5.1 and 5.2, PISM is a three-dimensional model. Moreover, parameters `Mx` and `My` have to be greater than or equal to three, so it is not possible to turn PISM into a 2D (flow-line) model by setting `Mx` or `My` to 1.

There is a way around this, though: by using the `-periodicity` option to tell PISM to make the computational grid y-periodic and providing initial and boundary conditions that are functions of x only one can ensure that there is no flow in the y-direction. (Option `-periodicity` takes an argument specifying the direction: `none`, `x`, `y` and `xy`— for "periodic in both X- and Y-directions".)

In this case `Mx` can be any number; we want to avoid unnecessary computations, though, so "`-Mx 3`" is the obvious choice.

One remaining problem is that PISM still expects input files to contain both `x` and `y` dimensions. To help with this, PISM comes with a Python script `flowline.py` that turns NetCDF files with N grid points along a flow line into files with 2D fields containing $N \times 3$ grid points.[15]

Here's an example which uses the script `util/flowlineslab.py` to create a minimal, and obviously unrealistic, dataset. A file `slab.nc` is created by `util/flowlineslab.py`, but it is not ready to use with PISM. Proceed as follows, after checking that `util/` is on your path:

```
$ flowlineslab.py                          # creates slab.nc with only an x-direction
$ flowline.py -o slab-in.nc --expand -d y slab.nc
```

produces a PISM-ready `slab-in.nc`. Specifically, `flowline.py` "expands" its input file in the y-direction. Now we can "bootstrap" from `slab-in.nc`:

```
$ mpiexec -n 2 pismr -surface given -i slab-in.nc -bootstrap -periodicity y \
    -Mx 201 -My 3 -Lx 1000 -Ly 4 -Lz 2000 -Mz 11 -y 10000 -o pism-out.nc
```

To make it easier to visualize data in the file created by PISM, "collapse" it:

```
$ flowline.py -o slab-out.nc --collapse -d y pism-out.nc
```

9.13 Managing source code modifications

"Practical usage" may include editing the source code to extend, fix or replace parts of PISM.

We provide both user-level (this manual) and developer-level documentation. Please see source code browsers at http://www.pism-docs.org for the latter.

- To use your (modified) version of PISM, you will need to follow the compilation from sources instructions in the *Installation Manual*
- We find it very useful to be able to check if a recent source code change broke something. PISM comes with "regression tests", which check if certain parts of PISM perform the way it should.[16]

  Run "`make test`" in the build directory to run PISM's regression tests.

[15] This script requires the `numpy` and `netCDF4` Python modules. Run `flowline.py -help` for a full list of options.
[16] This automates running verification tests described in section 10, for example.

Note, though, that while a test failure usually means that the new code needs more work, passing all the tests does not guarantee that everything works as it should. We are constantly adding new tests, but so far only a subset of PISM's functionality can be tested automatically.

- We strongly recommend using a version control system to manage code changes. Not only is it safer than the alternative, it is also more efficient.

10 Verification

> Two types of errors may be distinguished: modeling errors and numerical errors. Modeling errors arise from not solving the right equations. Numerical errors result from not solving the equations right. The assessment of modeling errors is *validation*, whereas the assessment of numerical errors is called *verification* ... Validation makes sense only after verification, otherwise agreement between measured and computed results may well be fortuitous.
>
> P. Wesseling, (2001) *Principles of Computational Fluid Dynamics*, pp. 560–561 [113]

Verification is the essentially mathematical task of checking that the predictions of the numerical code are close to the predictions of a continuum model, the one which the numerical code claims to approximate. It is a crucial task for a code as complicated as PISM. In verification there is no comparison between model output and observations of nature. Instead, one compares exact solutions of the continuum model, in circumstances in which they are available, to their numerical approximations.

Reference [92] gives a broad discussion of verification and validation in computational fluid dynamics. See [20] and [18] for discussion of verification issues for the isothermal and thermomechanically coupled shallow ice approximation (SIA), respectively, and for exact solutions to these models, and [17, 99] for verification using an exact solution to the SSA equations for ice streams.

In PISM there is a separate executable `pismv` which is used for SIA-related verification, and there are additional scripts for SSA-related verification. The source codes which are verified by `pismv` are, however, exactly the same source files as are run by the normal PISM executable `pismr`. In technical terms, `pismv` runs a derived class of the PISM base class.

Table 43 summarizes the many exact solutions currently available in PISM. Most of these exact solutions are solutions of *free boundary problems* for partial differential equations; only Tests A, E, J, K are fixed boundary value problems.

Table 44 shows how to run each of them on a coarse grids. Note that tests I and J require special executables `ssa_testi,ssa_testj` which are built with configuration flag `Pism_BUILD_EXTRA_EXECS` equal to `ON`. Table 45 gives the special verification-related options of the `pismv` executable.

Numerical errors are not, however, the dominant reasons why ice sheet models give imperfect results. The largest sources of errors include those from using the wrong (e.g. over-simplified or incorrectly-parameterized) continuum model, and from observational or pre-processing errors present in input data. Our focus here on numerical errors has a model-maintenance goal. It is *easier* to maintain code by quantitatively confirming that it produces small errors in cases where those can be measured, rather than "eyeballing" results to see that they are "right" according to human judgment.

The goal of verification is not generally to see that the error is zero at any particular resolution, or even to show that the error is small in a predetermined absolute sense. Rather the goals are

- to see that the error *is* decreasing,
- to measure the rate at which it decreases, and
- to develop a sense of the magnitude of numerical error before doing realistic ice sheet model runs.

Test	Continuum model tested	Comments	Reference
A	isothermal SIA, steady, flat bed, constant accumulation		[20]
B	isothermal SIA, flat bed, zero accum	similarity soln	[20, 45]
C	isothermal SIA, flat bed, growing accum	similarity soln	[20]
D	isothermal SIA, flat bed, oscillating accum	compensatory accum.	[20]
E	isothermal SIA; as A but with sliding in a sector	compensatory accum.	[20]
F	thermomechanically coupled SIA (mass and energy cons.), steady, flat bed	compensatory accum. and heating	[16, 18]
G	thermomechanically coupled SIA; as F but with oscillating accumulation	ditto	[16, 18]
H	bed deformation coupled with isothermal SIA	joined similarity	[19]
I	stream velocity computation using SSA (plastic till)		[99, 17]
J	shelf velocity computation using SSA		(source code)
K	pure conduction in ice and bedrock		[15]
L	isothermal SIA, steady, non-flat bed	numerical ODE soln	(source code)

Table 43: Exact solutions for verification.

Knowing the error decay rate may give a prediction of how fine a grid is necessary to achieve a desired smallness for the numerical error.

Therefore one must "go down" a grid refinement "path" and measure numerical error for each grid [92]. The refinement path is defined by a sequence of spatial grid cell sizes which decrease toward the refinement limit of zero size [77]. In PISM the timestep Δt is determined adaptively by a stability criterion (see subsection 9.9). In PISM one specifies the number of grid points, thus the grid cell sizes because the overall dimensions of the computational box are normally fixed; see subsection 5.1. By "measuring the error for each grid" we mean computing a norm (or norms) of the difference between the numerical solution and the exact solution.

For a grid refinement path example, in tests of the thermomechanically-coupled SIA model one refines in three dimensions, and these runs produced Figures 13, 14, and 15 of [18]:

```
pismv -test G -max_dt 10.0 -y 25000 -Mx 61 -My 61 -Mz 61 -z_spacing equal
pismv -test G -max_dt 10.0 -y 25000 -Mx 91 -My 91 -Mz 91 -z_spacing equal
pismv -test G -max_dt 10.0 -y 25000 -Mx 121 -My 121 -Mz 121 -z_spacing equal
pismv -test G -max_dt 10.0 -y 25000 -Mx 181 -My 181 -Mz 181 -z_spacing equal
pismv -test G -max_dt 10.0 -y 25000 -Mx 241 -My 241 -Mz 241 -z_spacing equal
pismv -test G -max_dt 10.0 -y 25000 -Mx 361 -My 361 -Mz 361 -z_spacing equal
```

Test	Example invocation
A	`pismv -test A -Mx 61 -My 61 -Mz 11 -y 25000`
B	`pismv -test B -Mx 61 -My 61 -Mz 11 -ys 422.45 -y 25000`
C	`pismv -test C -Mx 61 -My 61 -Mz 11 -y 15208.0`
D	`pismv -test D -Mx 61 -My 61 -Mz 11 -y 25000`
E	`pismv -test E -Mx 61 -My 61 -Mz 11 -y 25000`
F	`pismv -test F -Mx 61 -My 61 -Mz 61 -y 25000`
G	`pismv -test G -Mx 61 -My 61 -Mz 61 -y 25000`
H	`pismv -test H -Mx 61 -My 61 -Mz 11 -y 40034 -bed_def_iso`
I	`ssa_testi -ssa_method fd -Mx 5 -My 500 -ssa_rtol 1e-6 -ssafd_ksp_rtol 1e-11`
J	`ssa_testj -ssa_method fd -Mx 60 -My 60 -ssafd_ksp_rtol 1e-12`
K	`pismv -test K -Mx 6 -My 6 -Mz 401 -Mbz 101 -y 130000`
L	`pismv -test L -Mx 61 -My 61 -Mz 31 -y 25000`

Table 44: Canonical PISM verification runs using the exact solutions listed in Table 43.

Option	Description
`-test`	Choose verification test by single character name; see Table 43.
`-no_report`	Do not report errors at the end of a verification run.
`-eo`	Only evaluate the exact solution; no numerical approximation at all.

Table 45: pismv command-line options

The last two runs require a supercomputer! In fact the $361 \times 361 \times 361$ run involves more than 100 million unknowns, updated at each of millions of time steps. Appropriate use of parallelism (`mpiexec -n NN pismv`) and of the `-skip` modification to adaptive timestepping accelerates such fine-grid runs; see section 9.9.

Figures 18 through 22 show a sampling of the results of verifying PISM using the tests described above. These figures were produced automatically using Python scripts `test/vfnow.py` and `test/vnreport.py`. See subsection 9.11.

These figures *do not* show outstanding rates of convergence, relative to textbook partial differential equation examples. For the errors in tests B and G, see the discussion of free margin shape in [20]. For the errors in test I, the exact continuum solution is not very smooth at the free boundary [99].

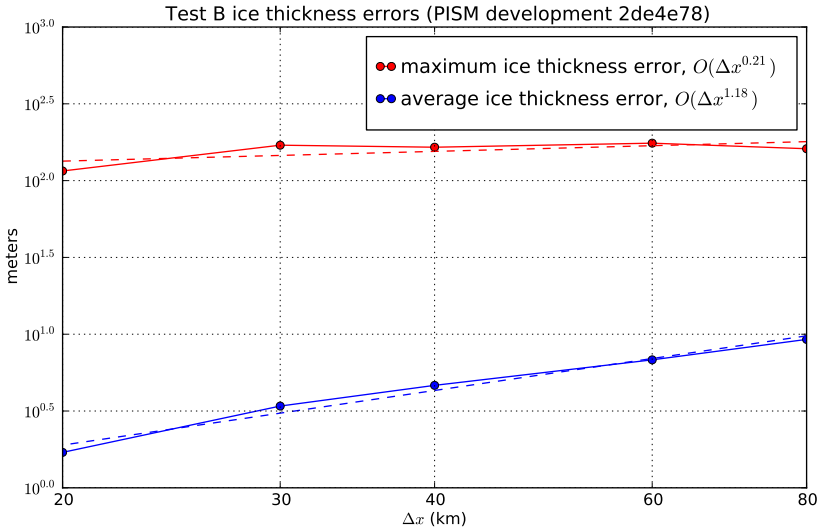


Figure 18: Numerical thickness errors in test B. See [20] for discussion.

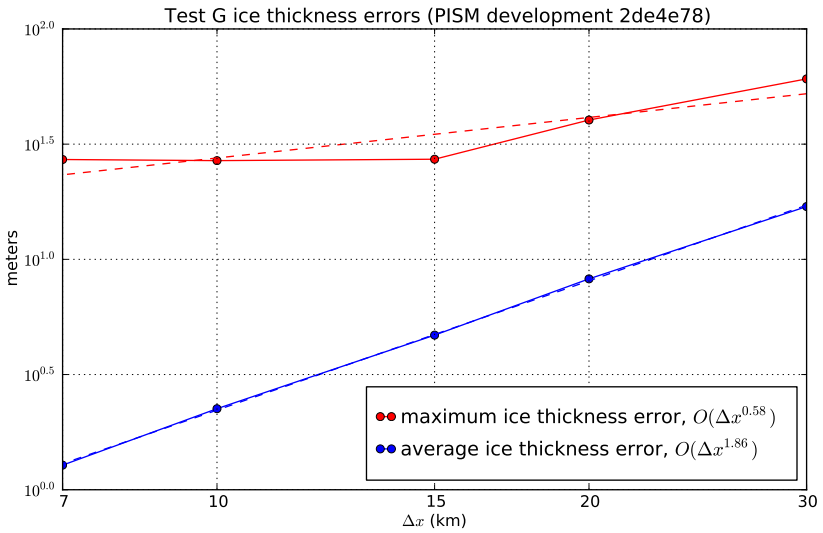


Figure 19: Numerical thickness errors in test G. See [18] and [20].

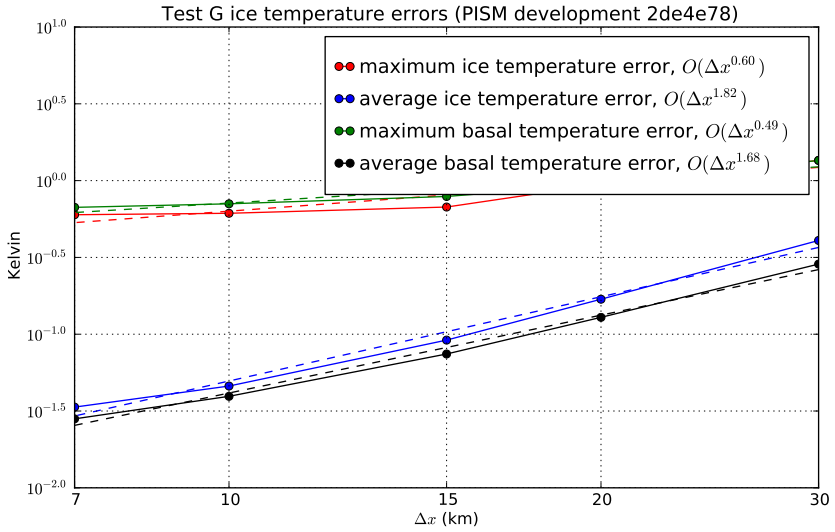


Figure 20: Numerical temperature errors in test G. See [18].

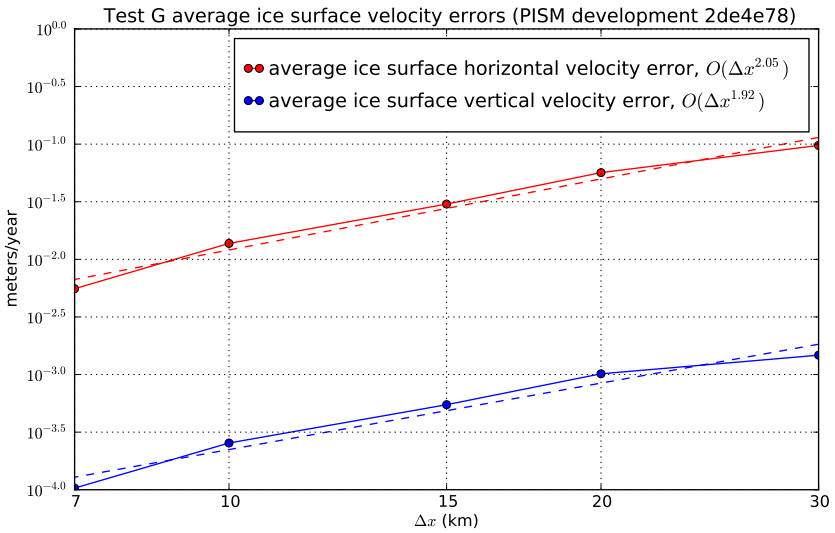


Figure 21: Numerical errors in computed surface velocities in test G.

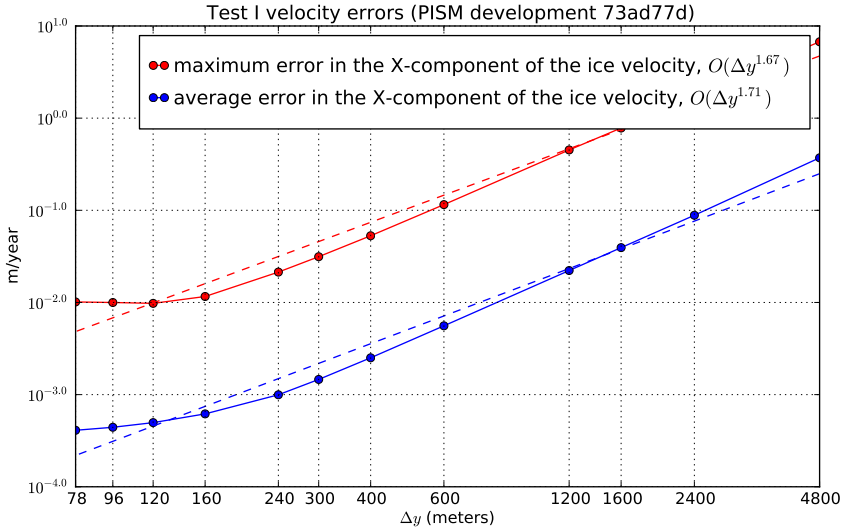


Figure 22: Numerical errors in horizontal velocities in test I, an ice stream. See [99, 17].

11 Simplified geometry experiments with PISM

There have been three stages of ice sheet model intercomparisons based on simplified geometry experiments since the early 1990s [21].

EISMINT I [56, European Ice Sheet Modeling INiTiative][17] was the first of these and involved only the isothermal shallow ice approximation (SIA). Both fixed margin and moving margin experiments were performed in EISMINT I, and various conclusions were drawn about the several numerical schemes used in the intercomparison. EISMINT I is superceded, however, by verification using the full variety of known exact solutions to the isothermal SIA [20]. The "rediscovery", since EISMINT I, of the Halfar similarity solution with zero accumulation [45], and verification runs using that solution, already suffices to measure the isothermal SIA performance of PISM more precisely than would be allowed by comparison to EISMINT I results.

EISMINT II [85] pointed out interesting and surprising properties of the thermocoupled SIA. References [18, 46, 47, 86, 94, 17] each interpret the EISMINT II experiments and/or describe attempts to add more complete physical models to "fix" the (perceived and real) shortfalls of ice sheet model behavior on EISMINT II experiments. We believe that the discussion in [87, 86, 18] adequately explains the "spokes" in EISMINT II experiment F as a genuine fluid instability, while [32] and Appendix B of [17] adequately cautions against the continuum model that generates the "spokes" in EISMINT II experiment H. Thus we can move on from that era of controversy. In any case, PISM has built-in support for all of the published and unpublished EISMINT II experiments; these are described in the next subsection.

The ISMIP (Ice Sheet Model Intercomparison Project)[18] round of intercomparisons covers 2008–2013 (at least). There are four components of ISMIP substantially completed, namely HOM = Higher Order Models [82, 33], HEINO = Heinrich Event INtercOmparison [43, 23], MISMIP (below), and MISMIP3d (also below).

PISM participated in HEINO, but this ability is unmaintained. We believe the continuum problem described by HEINO, also used in EISMINT II experiment H (above), is not meaningfully approximateable because of a required discontinuous jump in the basal velocity field. The continuum problem predicts infinite vertical velocity because of this jump [17, Appendix B]. Details of the numerical schemes and their results are irrelevant if the continuum model makes such a prediction. PISM offers the physical continuum model described in [17], an SIA+SSA hybrid, as an alternative to the continuum model used in ISMIP-HEINO and EISMINT II experiment H. Indeed the SIA+SSA hybrid is offered as a unified shallow model for real ice sheets (section 3).

There is no current plan to support ISMIP-HOM [82, 33], but comparison of shallow PISM results to exact Stokes solutions is a goal for PISM evaluation.

A third and fourth ISMIP parts are the two parts of the Marine Ice Sheet Model Intercomparison Project, MISMIP [81] and MISMIP3D [80]. These experiments are supported in PISM, as described in subsections 11.2 and 11.3 below.

[17] See http://homepages.vub.ac.be/%7Ephuybrec/eismint.html.
[18] See http://homepages.vub.ac.be/%7Ephuybrec/ismip.html.

11.1 EISMINT II

There are seven experiments described in the published EISMINT II writeup [85]. They are named A, B, C, D, F, G, and H. They have these common features:

- runs are for 2×10^5 years, with no prescribed time step;
- a 61×61 horizontal grid on a square domain (1500 km side length) is prescribed;
- surface inputs (temperature and mass balance) have angular symmetry around the grid center;
- the bed is flat and does not move (no isostasy);
- the temperature in the bedrock is not modeled;
- only the cold (not polythermal) thermomechanically-coupled SIA is used [85]; and
- basal melt rates do not affect the evolution of the ice sheet.

The experiments differ from each other in their various combinations of surface temperature and mass balance parameterizations. Experiments H and G involve basal sliding, under the physically-dubious SIA sliding rubric [17, Appendix B], while the others don't. Four experiments start with zero ice (A,F,G,H), while the other experiments (B,C,D) start from the final state of experiment A.

In addition to the seven experiments published in [85], there were an additional five experiments described in the EISMINT II intercomparison description [84], labeled E, I, J, K, and L. These experiments share most features listed above, but with the following differences. Experiment E is the same as experiment A except that the peak of the accumulation, and also the low point of the surface temperature, are shifted by 100 km in both x and y directions; also experiment E starts with the final state of experiment A. Experiments I and J are similar to experiment A but with non-flat "trough" bed topography. Experiments K and L are similar to experiment C but with non-flat "mound" bed topography.

See Table 46 for how to run all EISMINT II experiments in PISM. Experiments below the horizontal line are only documented in [84].

The vertical grid is not specified in EISMINT II, but a good simulation of the thermomechanically-coupled conditions near the base of the ice requires relatively-fine resolution there. We suggest using the default unequally-spaced grid. With 61 levels it gives a grid spacing of ∼20 m in the ice layer closest to the bed, but more vertical levels are generally better. Alternatively these experiments can be done with an equally-spaced grid; in this case we suggest using enough vertical levels to give 20 m spacing, for example. When there is sliding, even more vertical resolution is recommended (see Table 46). Also, the vertical extent must be sufficient so that when the ice thickness grows large, especially before thermo-softening brings it back down, the vertical grid is tall enough to include all the ice. Table 46 therefore includes suggested settings of `-Lz`; experiment F is different because ice thickness increases with colder temperatures.

These SIA-only simulations parallelize well. Very roughly, for the standard 61×61 horizontal grid, wall-clock-time speedups will occur up to about 30 processors. Runs on finer (horizontal) grids will benefit from even more processors. Also, the "skip" mechanism which avoids updating the temperature at each time step is effective, so options like `-skip -skip_max 5` are recommended.

Command: "pisms " +	**Relation to experiment A**
`-eisII A -Mx 61 -My 61 -Mz 61 -Lz 5000 -y 2e5 -o eisIIA.nc`	
`-eisII B -i eisIIA.nc -y 2e5 -o eisIIB.nc`	warmer
`-eisII C -i eisIIA.nc -y 2e5 -o eisIIC.nc`	less snow (lower accumulation)
`-eisII D -i eisIIA.nc -y 2e5 -o eisIID.nc`	smaller area of accumulation
`-eisII F -Mx 61 -My 61 -Mz 81 -Lz 6000 -y 2e5 -o eisIIF.nc`	colder; famous spokes [18]
`-eisII G -Mx 61 -My 61 -Mz 201 -Lz 5000 -y 2e5 -o eisIIG.nc`	sliding (regardless of temperature)
`-eisII H -Mx 61 -My 61 -Mz 201 -Lz 5000 -y 2e5 -o eisIIH.nc`	melt-temperature activated sliding
`-eisII E -i eisIIA.nc -y 2e5 -o eisIIE.nc`	shifted climate maps
`-eisII I -Mx 61 -My 61 -Mz 61 -Lz 5000 -y 2e5 -o eisIII.nc`	trough topography
`-eisII J -i eisIII.nc -y 2e5 -o eisIIJ.nc`	trough topography and less snow
`-eisII K -Mx 61 -My 61 -Mz 61 -Lz 5000 -y 2e5 -o eisIIK.nc`	mound topography
`-eisII L -i eisIIK.nc -y 2e5 -o eisIIL.nc`	mound topography and less snow

Table 46: Running the EISMINT II experiments in PISM. Use `-skip -skip_max 5`, on the 61×61 default grid, for significant speedup.

The EISMINT II experiments can be run with various modifications of the default settings. For instance, a twice-finer grid in the horizontal is "`-Mx 121 -My 121`". Table 47 lists some optional settings which are particular to the EISMINT II experiments.

See subdirectory `examples/eismintII/` for a simple helper script `runexp.sh`.

11.2 MISMIP

This intercomparison addresses grounding line dynamics by considering an idealized one-dimensional stream-shelf system. In summary, a flowline ice stream and ice shelf system is modeled, the reversibility of grounding line movement under changes in the ice softness is tested, different sliding laws are tested, and the behavior of grounding lines on reverse-slope beds is tested. The intercomparison process is described at the website

http://homepages.ulb.ac.be/~fpattyn/mismip/

Find a full text description there, along with the published report on the results [81]; that paper includes results from PISM version 0.1. These documents are essential reading for understanding MISMIP results generally, and for appreciating the brief discussion in this subsection.

PISM's version of MISMIP includes an attached ice shelf even though modeling the shelf is theoretically unnecessary in the flow line case. The analysis in [101] shows that the only effect of an ice shelf, in the flow line case, is to transfer the force imbalance at the calving front directly to the ice column at the grounding line. Such an analysis does not apply to ice shelves with two horizontal dimensions; real ice shelves have "buttressing" and "side drag" and other forces not present in the flow line [35]. See the next

Option	Default values [expers]	Units	Meaning
`-eisII`	A		Choose single character name of EISMINT II [85] simplified geometry experiment. See Table 46.
`-Mmax`	0.5 [ABDEFGHIK], 0.25 [CJL]	m/a	max value of accumulation rate
`-Rel`	450 [ABEFGHIK], 425 [CDJL]	km	radial distance to equilibrium line
`-Sb`	10^{-2} [*all*]	(m/a)/km	radial gradient of accumulation rate
`-ST`	1.67×10^{-2} [*all*]	K/km	radial gradient of surface temperature
`-Tmin`	238.15 [ACDEGHIJKL], 243.15[B], 223.15[F]	K	max of surface temperature
`-bmr_in_cont`			Include the basal melt rate in the mass continuity computation; overrides EISMINT II default.

Table 47: Changing the default settings for EISMINT II

subsection on MISMIP3d and the Ross ice shelf example in section 12.2, among other examples.

We must adapt the usual 3d PISM model to two horizontal dimensions, i.e. to do flow-line problems (see section 9.12). The flow direction for MISMIP is taken to be "x". We periodize the cross-flow direction "y", and use the minimum number of points in the y-direction. This number turns out to be "`-My 3`"; fewer points than this in the cross-flow direction confuses the finite difference scheme.

PISM can do MISMIP experiments with either of two applicable ice dynamics models. Model 1 is a pure SSA model; "category 2" in the MISMIP classification. Model 2 combines SIA and SSA velocities as described in [115]; "category 3" because it resolves "vertical" shear (i.e. using SIA flow).

There are many runs for a complete MISMIP intercomparison submission. Specifically, for a given model there are 62 runs for each grid choice, and three (suggested) grid choices, so a full suite is $3 \times 62 = 186$ runs.

The coarsest grid ("mode 1") has 12 km spacing. The finest grid, "mode 2" with 1.2 km spacing, accounts for all the compute time, however; in the MISMIP description it is 1500 grid spaces in the flow line direction (= 3001 grid *points* in PISM's doubled computational domain). In between is "mode 3", a mode interpretable by the intercomparison participant, and here we just use a 6 km grid.

The implementation of MISMIP in PISM conforms to the intercomparison description, but that document specifies

> ... we require that the rate of change of grounding line position be 0.1 m/a or less, while the rate of change of ice thickness at each grid point at which ice thickness is defined must be less than 10^{-4} m/a ...

as a standard for "steady state". The scripts here do not implement this stopping criterion. However, we report enough information, in PISM output files with scalar and spatially-variable time-series, to compute a grounding line rate or the time at which the thickness rate of change drops below 10^{-4} m/a.

See

`examples/mismip/mismip2d/README.md`

for usage of the scripts that run MISMIP experiments in PISM. For example, as described in this `README.md`, the commands

```
$ ./run.py -e 1a --mode=1 > experiment-1a-mode-1.sh
$ bash experiment-1a-mode-1.sh 2 >& out.1a-mode-1 &
$ ./plot.py ABC1_1a_M1_A7.nc -p -o profileA7.png
```

first generate a bash script, then use it to do a run which takes about 20 minutes, and then generate an image in `.png` format. Note that step 7 is in the middle of the experiment. It is shown in Figure 23 (left).

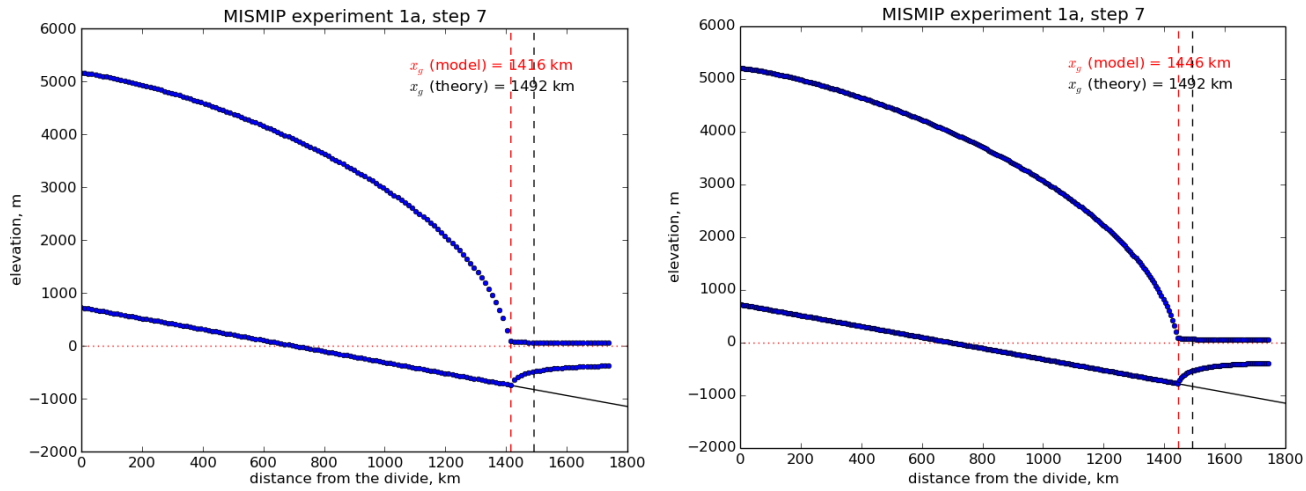


Figure 23: A marine ice sheet profile in the MISMIP intercomparison; PISM model 1, experiment 1a, at step 7. Left: grid mode 1 (12 km grid). Right: grid mode 3 (6 km grid).

The script `MISMIP.py` in `examples/mismip/mismip2d` has the ability to compute the profile from the Schoof's [101] asymptotic-matching boundary layer theory. This script is a Python translation, using `scipy` and `pylab`, of the MATLAB codes in `http://homepages.ulb.ac.be/~fpattyn/mismip/MISMIP_distribution.tar`. For example,

```
$ python MISMIP.py -o mismip_analytic.png
```

produces a `.png` image file with Figure 24. By default `run.py` uses the asymptotic-matching thickness result from the [101] theory to initialize the initial ice thickness, as allowed by the MISMIP specification.

Generally the PISM result does not put the grounding line in the same location as Schoof's boundary layer theory, and at least at coarser resolutions the problem is with PISM's numerical solution, not with Schoof's semi-analytic theory. The result improves under grid refinement, however. Results from grid mode 3 with 6 km spacing, instead of 12 km in mode 1, are the right part of Figure 23. The corresponding results from grid mode 2, with 1.2 km spacing, are in Figure 25. Note that the difference between the

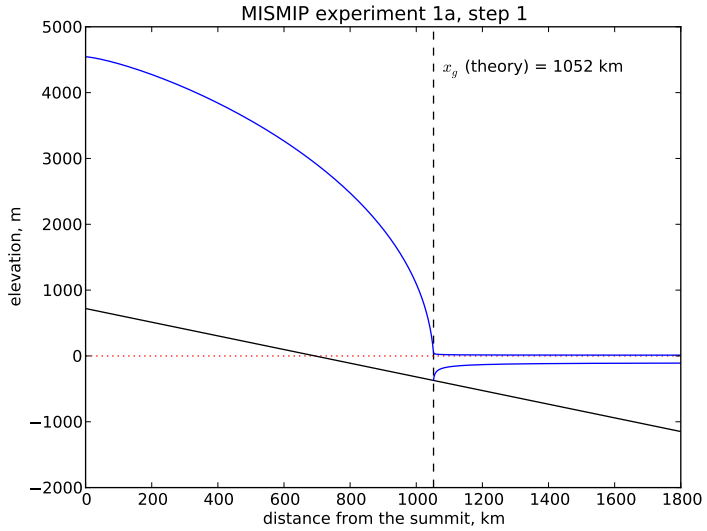


Figure 24: Analytical profile for steady state of experiment 1a, step 1, from theory in [101]. This is a boundary layer asymptotic matching result, but not the exact solution to the equations.

numerical grounding line location and the semi-analytical location has been reduced from 76 km for grid mode 1 to 16 km for grid mode 2 (a factor of about 5), by using a grid refinement from 12 km to 1.2 km (a factor of about 10).

11.3 MISMIP3d

The ice2sea MISMIP3d intercomparison is a two-horizontal-dimensional extension of the flowline case described above. As before, in MISMIP3d the grounding line position and its reversibility under changes of physical parameters is analyzed. Instead of changing the ice softness, however, the spatial distribution and magnitude of basal friction is adjusted between experiments. The applied basal friction perturbation of the basal friction is a localized gaussian "bump" and thus a curved grounding line is obtained. In contrast to the flowline experiments, no (semi-)analytical solutions are available to compare to the numerical results.

A full description of the MISMIP3d experiments can be found at

`http://homepages.ulb.ac.be/~fpattyn/mismip3d/`

and the results are published in [80].

A complete set of MISMIP3d experiments consists of three runs: Firstly, a flowline solution on a linearly-sloped bed, similar to the flowline MISMIP experiments of the previous section, is run into a steady state ("standard experiment `Stnd`"). Then the localized sliding perturbation is applied ("perturbation experiment") causing the grounding line to shift and lose symmetry. Two different amplitudes of the perturbation are considered ("`P10`" and "`P75`"). Finally, beginning from the final state of the perturbation experiment, the sliding perturbation is removed and the system is run again into steady state ("reversibility experiment"). The resulting geometry, in particular the grounding line position, is expected

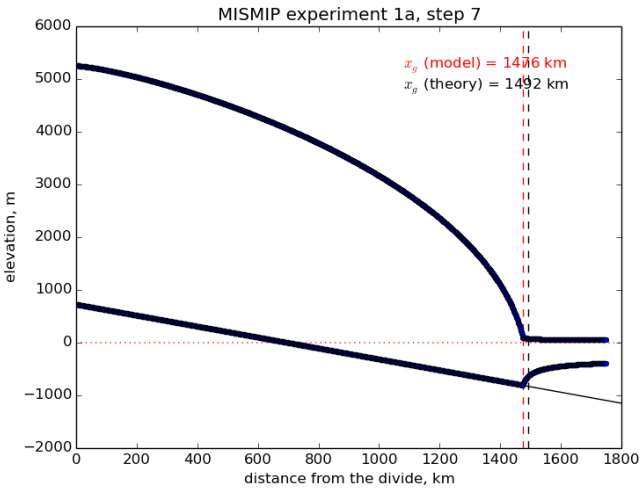


Figure 25: Results from MISMIP grid mode 2, with 1.2 km spacing, for steady state of experiment 1a: profile at step 7 (compare Figure 23).

to be close to that of the standard experiment. Expecting such reversibility assumes that a particular stationary ice geometry only depends on its physical parameters and boundary conditions and not on how it is dynamically reached.

For these experiments in PISM, a Python script generates a shell script which has the commands and options for running a MISMIP3d experiment. The python script is `createscript.py` in the folder `examples/mismip/mismip3d/`. Run

```
$ ./createscript.py -h
```

to see a usage message. A `README.md` gives a tutorial on how to use `createscript.py` and do the runs themselves.

For the flowline `Stnd` experiment, as in the MISMIP case, a computational domain with three grid points in the direction orthogonal to the ice flow (arbitrarily chosen as y-direction) is chosen by `createscript.py`. For the perturbation and reversibility experiments a domain is defined which is symmetric along the ice-divide (mirror symmetry) and along the center line of the ice flow, while the side boundaries are periodic, which corresponds to a free-slip condition for the flow in x-direction. Though this choice of the symmetric computational domain increases computational cost, it allows us to use standard PISM without fixing certain boundary conditions in the code. (That is, it avoids the issues addressed in the regional mode of PISM; see section 13.)

PISM participated in the MISMIP3d intercomparison project [80] using version pism0.5, and the exact results can be reproduced using that version. PISM's results, and the role of resolution and the new subgrid grounding line interpolation scheme are discussed in [30].

We observed a considerable improvement of the results with respect to the absolute grounding line positions compared to other models (e.g. the FE reference model Elmer/Ice) and to the reversibility when applying the subgrid grounding line interpolation method; see Figure 26. Furthermore, we observed that only using SSA yields almost the same results as the full hybrid SIA+SSA computation for the

MISMIP3D (and also the MISMIP) experiments, but, when not applying the SIA computation, after a considerably shorter computation time (about 10 times shorter). We explain the small and almost negligible SIA velocities for the MISMIP(3D) experiments with the comparably small ice surface gradients in the MISMIP3d ice geometries. See Fig. 27 for a comparison of SSA and SIA velocities in the MISMIP3D geometry. Note that both Figures 26 and 27 were generated with resolution of $\Delta x = \Delta y = 1$ km.

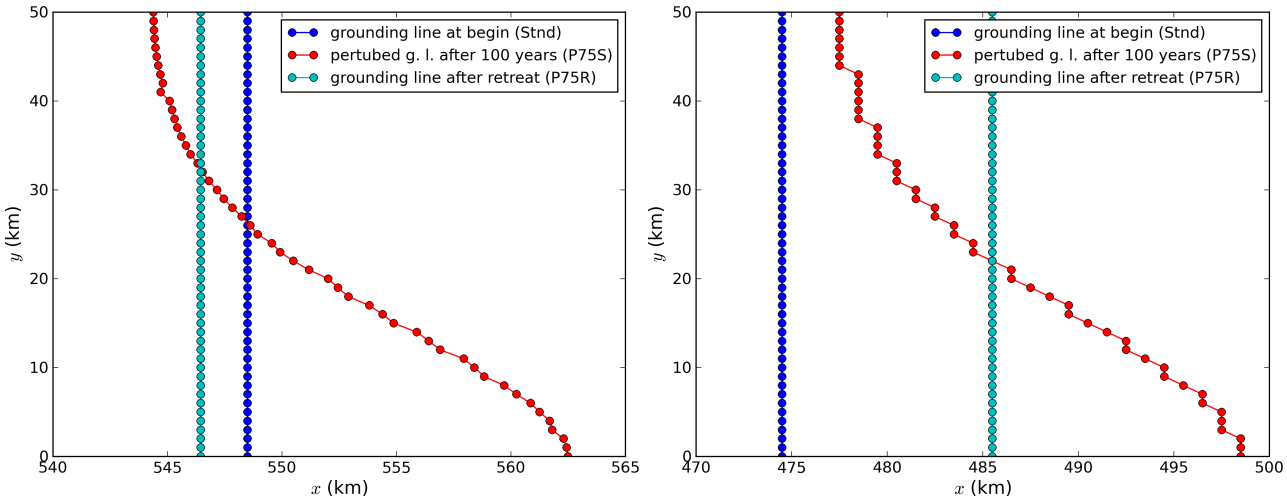


Figure 26: Comparison between the grounding lines of the higher-amplitude ("P75") MISMIP3d experiments performed with PISM when using the subgrid grounding line interpolation method (left) or not using it (right). In both cases the SIA+SSA hybrid is used.

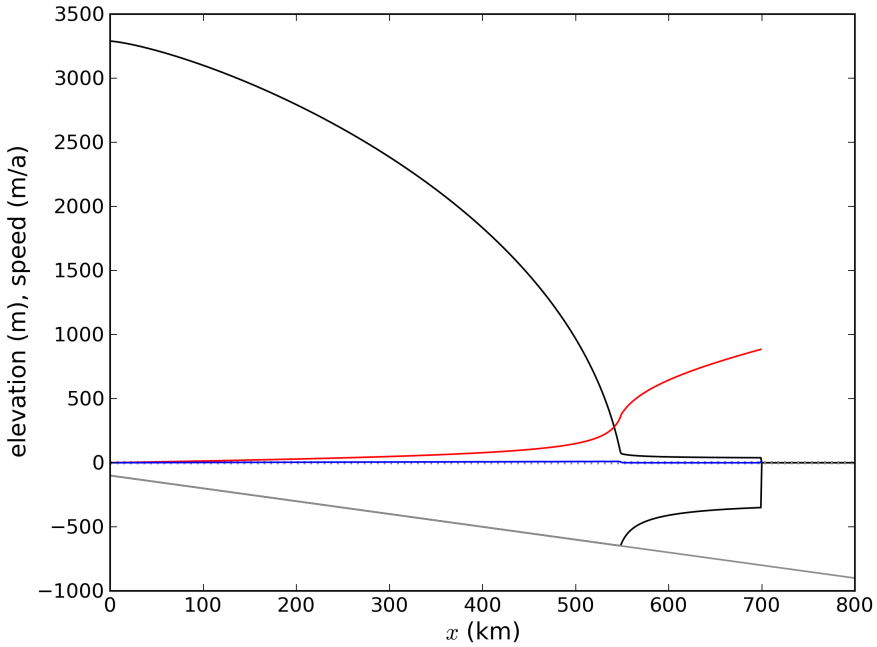


Figure 27: The SIA velocities are negligible in the MISMIP3d standard experiment ("`Stnd`"). The steady state ice geometry is plotted (black) together with the computed SSA velocity (red) and SIA velocity (blue). The SIA velocity reaches its maximum value of about 10 m/a at the grounding line, about two orders of magnitude less than the maximum of the SSA velocity.

12 Validation case studies

"Validation" describes the comparison of numerical model output with physical observations in cases where the observations are sufficiently-complete and of sufficient quality so that the performance of the numerical model can be assessed [92, 113]. Roughly speaking, validation can happen when the observations or data are better than the model, so the comparison measures the quality of the numerical model and not merely errors in, or incompleteness of, the data. Because of the difficulty of measuring boundary conditions for real ice flows, this situation is not automatic in glaciology, or even common.[19] Nonetheless we try two cases, first where PISM is applied on millimeter scale to model a laboratory experiment, and second for a large-scale ice flow in which all uncertainties of bedrock topography, basal sliding, and subglacial hydrology are removed, namely a present-day ice shelf.

12.1 An SIA flow model for a table-top laboratory experiment

Though there are additional complexities to the flow of real ice sheets, an ice sheet is a shear-thinning fluid with a free surface. PISM ought to be able to model such flows in some generality. We test that ability here by comparing PISM's isothermal SIA numerical model to a laboratory observations of a 1% Xanthan gum suspension in water in a table-top, moving-margin experiment by R. Sayag and M. Worster [97, 96]. The "gum" fluid is more shear-thinning than ice, and it has much lower absolute viscosity values, but it has the same density. This flow has total mass ~ 1 kg, compared to $\sim 10^{18}$ kg for the Greenland ice sheet.

We compare our numerical results to the "constant-flux" experiment from [97]. Figure 28 shows the experimental setup by reproducing Figures 2(c) and 2(d) from that reference. A pump pushes the translucent blue-dyed fluid through a round 8 mm hole in the middle of a clear table-top at a mass rate of about 3 gm/s. The downward-pointing camera, which produced the right-hand figure, allows measurement of the location of margin of the "ice cap", and in particular of its radius. The measured radii data are the black dots in Figure 29.

The closest glaciological analog would be an ice sheet on a flat bed fed by positive basal mass balance (i.e. "refreeze") underneath the dome, but with zero mass balance elsewhere on the lower and upper surfaces. However, noting that the mass-continuity equation is vertically-integrated, we may model the input flux (mass balance) as arriving at the *top* of the ice sheet, to use PISM's climate-input mechanisms. The flow though the input hole is simply modeled as constant across the hole, so the input "climate" uses `-surface given` with a field `climatic_mass_balance`, in the bootstrapping file, which is a positive constant in the hole and zero outside. While our replacement of flow into the base by mass balance at the top represents a very large change in the vertical component of the velocity field, we still see good agreement in the overall shape of the "ice sheet", and specifically in the rate of margin advance.

Sayag & Worster estimate Glen exponent $n = 5.9$ and a softness coefficient $A = 9.7 \times 10^{-9}\,\mathrm{Pa}^{-5.9}\,\mathrm{s}^{-1}$ for the flow law of their gum suspension, using regression of laboratory measurements of the radius. (Compare PISM defaults $n = 3$ and $A \approx 4 \times 10^{-25}\,\mathrm{Pa}^{-3}\,\mathrm{s}^{-1}$ for ice.) Setting the Sayag & Worster values is one of several changes to the configuration parameters, compared to PISM ice sheet defaults,

[19] Which explains the rise of "simplified geometry intercomparisons"; see section 11.

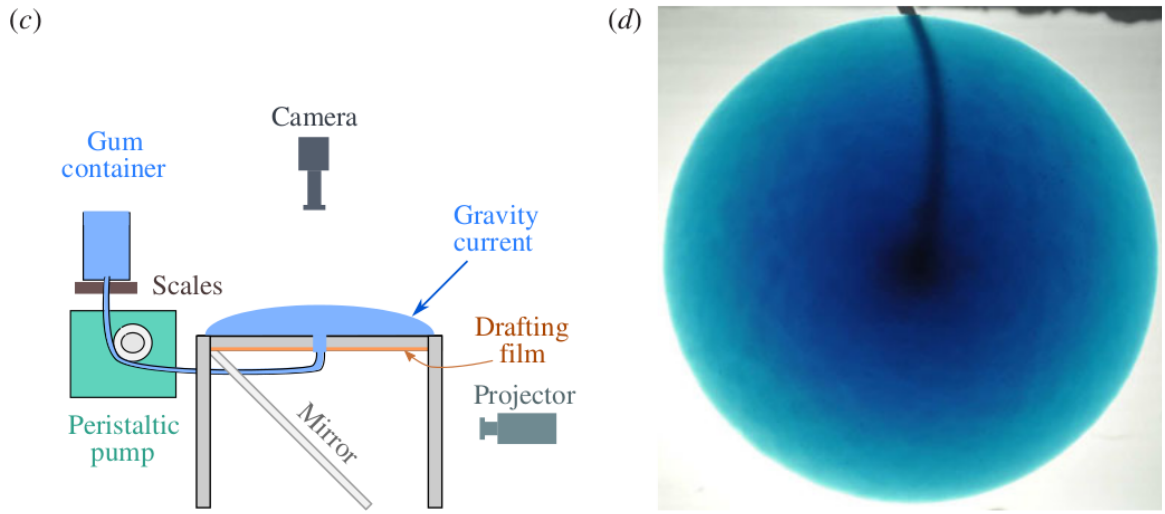


Figure 28: Reproduction of Figures 2(c) and 2(d) from [97]. Left: experimental apparatus used for "constant-flux release" experiment. Right: snapshot of constant-flux experiment (plan view), showing an axisymmetric front.

which are done in part by overriding parameters at run time by using the `-config_override` option. See `examples/labgum/preprocess.py` for the generation of a configuration `.nc` file with these settings.

To run the example on the default 10 mm grid, first do

```
$ python preprocess.py
```

and then do a run for 746 model seconds [97] on the 10 mm grid on a 520 mm × 520 mm square domain using 4 processors:

```
$ ./rungum.sh 4 52 &> out.lab52 &
```

This run generates text file `out.lab52`, diagnostic files `ts_lab52.nc` and `ex_lab52.nc`, and final output `lab52.nc`. This run took about 5 minutes on a 2013 laptop, thus roughly real time! When it is done, you can compare the modeled radius to the experimental data:

```
$ ./showradius.py -o r52.png -d constantflux3.txt ts_lab52.nc
```

You can also redo the whole thing on higher resolution grids (here: 5 and 2.5 mm), here using 6 MPI processes if the runs are done simultaneously, and when it is done after several hours, make a combined figure just like Figure 29:

```
$ ./preprocess.py -Mx 104 -o initlab104.nc
$ ./preprocess.py -Mx 208 -o initlab208.nc
$ ./rungum.sh 2 104 &> out.lab104 &
$ ./rungum.sh 4 208 &> out.lab208 &
$ ./showradius.py -o foo.png -d constantflux3.txt ts_lab*.nc
```

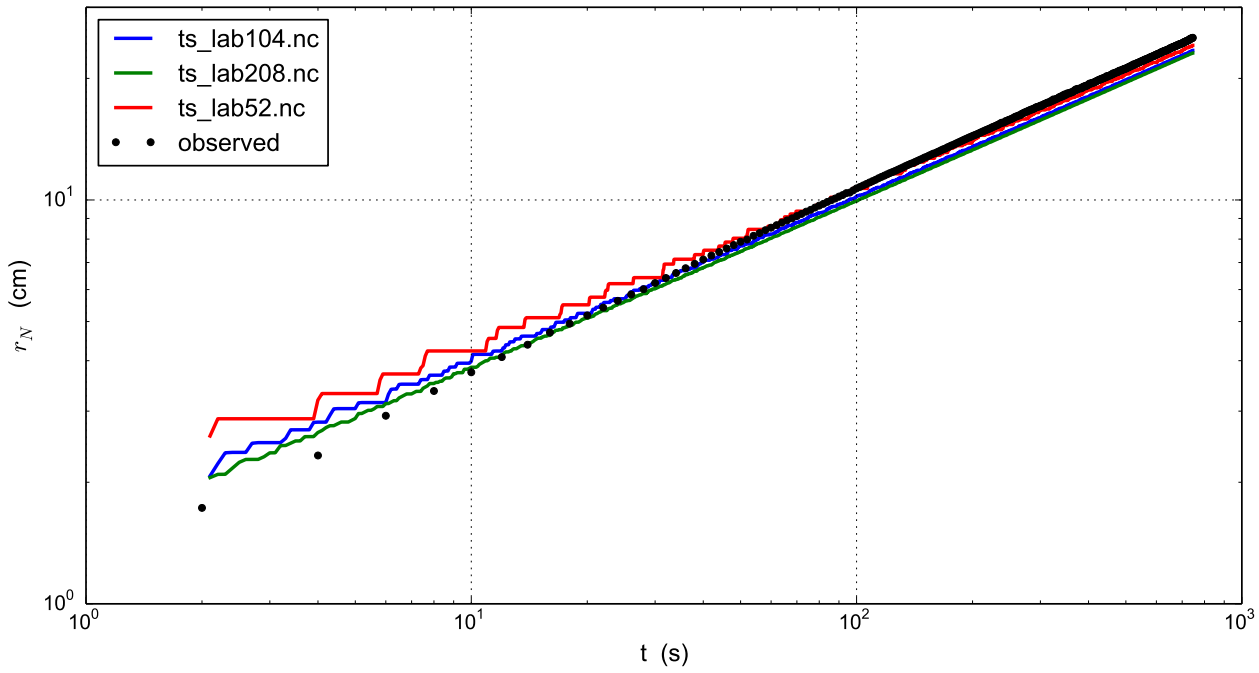


Figure 29: Radius $r_N(t)$ for runs with 10 mm (`ts_lab52.nc`), 5 mm (`ts_lab104.nc`), and 2.5 mm (`ts_lab208.nc`) grids, compared to observations from Sayag & Worster's [97] table-top "ice cap" (gravity current) made from a 1 % Xanthan gum suspension, as shown in Figure 28.

We see that on the coarsest grid the modeled volume has "steps" because the margin advances discretely. Note we are computing the radius by first computing the fluid-covered area a on the cartesian grid, and then using $a = \pi r^2$ to compute the radius.

Results are better on finer grids, especially at small times, because the input hole has radius of only 8 mm. Furthermore this "ice cap" has radius comparable to the hole for the first few model seconds. The early evolution is thus distinctly non-shallow, but we see that increasing the model resolution reduces most of the observation-model difference. In fact there is little need for "higher-order" stresses because the exact similarity solution of the shallow continuum equations, used by Sayag & Worster, closely-fits the data even for small radius and time [97, Figure 4].

In any case, the large-time observations are very closely-fit by the numerical results at all grid resolutions. We have used the Glen-law parameters n, A as calculated by Sayag & Worster, but one could do parameter-fitting to get the "best" values if desired. In particular, roughly speaking, n controls the slope of the results in Figure 29 and A controls their vertical displacement.

12.2 An SSA flow model for the Ross Ice Shelf in Antarctica

As part of the EISMINT series of intercomparisons, MacAyeal and others [72] successfully validated early-1990s ice shelf numerical models using velocity data for the Ross ice shelf. The data were from the RIGGS survey [11], acquired in the period 1973–1978 and measured at a few hundred locations in a grid across the shelf. Substantial modelling developments followed EISMINT-Ross, including inverse

modeling to recover depth-averaged viscosity [93] and parameter-sensitivity studies [52]. Previous PISM versions set up the EISMINT-Ross flow model and performed the diagnostic computation, with RIGGS data for validation.

However, availability of rich new data sets for ice sheet modeling, including the ALBMAP v1 [66] ice sheet geometry, bedrock, and climate data set, and the radar-derived (InSAR) MEaSUREs Antarctica Velocity Map [91], allows us to use more complete, recent, and higher-resolution data for the same basic job. Furthermore one can extend the diagnostic Ross ice shelf calculation both to other ice shelves around Antarctica and to time-evolving ("prognostic") cases using the eigencalving [67] mechanisms.

The scripts in this subsection are found in directory `examples/ross/`. In summary, the script `preprocess.py` downloads data and builds a NetCDF input file for PISM. For the diagnostic computation we document first, the script `run_diag.sh` (in subdirectory `examples/ross/diagnostic/`) runs PISM. The script `plot.py` shows a comparison of observations and model results, as in Figure 30.

Preprocessing the data

The script `preprocess.py` downloads ALBMAP and MEaSUREs NetCDF files using `wget`; these files total around 100 Mb. Then it uses NCO to cut out the relevant portion of the grid and CDO to conservatively-interpolate the high-resolution (500 m) velocity data onto the coarser (5 km) geometry grid used in ALBMAP. The script `nc2cdo.py` from directory `util/`, prepares the NetCDF file for the application of CDO, which requires complete projection information. Do

```
$ cd examples/ross/
$ ./preprocess.py
```

The NetCDF file `Ross_combined.nc` produced by `preprocess.py` contains ice thickness, bed elevations, surface temperature, net accumulation, as well as latitude and longitude values. All of these are typical of ice sheet modelling data, both in diagnostic runs and as needed to initialize and provide boundary conditions for prognostic (evolutionary) runs; see below for the prognostic case with these data. The `_combined` file also has variables `u_ssa_bc` and `v_ssa_bc` for the boundary values used in the (diagnostic and prognostic) computation of velocity. They are used at all grounded locations and at ice shelf cells that are immediate neighbors of grounded ice. The variable `bc_mask` specifies these locations. Finally the variables `u_ssa_bc,v_ssa_bc`, which contain observed values, are used after the run to compare to the computed interior velocities.

Diagnostic computation of ice shelf velocity

The diagnostic velocity computation bootstraps from `Ross_combined.nc` and does a zero-year run; in the 211×211 grid case we demonstrate below, the key parts of the PISM command are

```
pismr -i ../Ross_combined.nc -bootstrap -Mx 211 -My 211 -Mz 3 -Lz 3000 -z_spacing equal \
    -surface given -stress_balance ssa -energy none -yield_stress constant -tauc 1e6 \
    -pik -ssa_dirichlet_bc -y 0 -ssa_e 0.6 -ssafd_ksp_monitor
```

The computational grid here is the "native" 5 km data grid used in ALBMAP. Regarding the options,

- The maximum thickness of the ice is 2766 m so we choose a height for the computational box large enough to contain the ice (i.e. `-Lz 3000`). Vertical grid resolution is, however, unimportant in this case because we use the SSA stress balance only, and the temperature set at bootstrapping suffices to determine the ice softness; thus the options `-Mz 3 -z_spacing equal -energy none`.

- Option `-stress_balance ssa` selects the SSA stress balance and turns off the SIA stress balance computation, since our goal is to model the ice shelf. It also side-steps a technical issue: PISM uses periodic boundary conditions at domain boundaries and most fields in this setup are not periodic. Turning off SIA avoids operations such as differencing surface elevation across the domain edges. For a more complete solution to this technical issue see section 13 about a regional model using option `-no_model_strip` and executable `pismo`.

- Option `-y 0` chooses a diagnostic run.

- Option `-pik` is equivalent to `-cfbc -kill_icebergs` in this non-evolving example. Note that `-kill_icebergs` removes effectively-detached bits of ice, especially in McMurdo sound area, so that the SSA problem is well-posed for the grounded-ice-sheet-connected ice shelf.

- Option `-ssa_dirichlet_bc` forces the use of fields `u_ssa_bc,v_ssa_bc,bc_mask` described above. The field `bc_mask` is 1 at boundary condition locations, and 0 elsewhere. For the prognostic runs below, the ice thickness is also fixed at boundary condition locations, so as to prescribe ice flux as an ice shelf input.

- Options `-yield_stress constant -tauc 1e6` essentially just turn off the grounded-ice evolving yield stress mechanism, which is inactive anyway, and force a high resistance under grounded ice so it does not slide.

- Option `-ssa_e 0.6` is the single tuned parameter; this value gives good correlation between observed and modeled velocity magnitudes.

- Option `-ssafd_ksp_monitor` provides feedback on the linear solver iterations "underneath" the nonlinear (shear-thinning) SSA solver iteration.

There is no need to type in the above command; just do

```
$ cd diagnostic/
$ ./run_diag.sh 2 211 0.6
```

Note `run_diag.sh` accepts three arguments: `run_diag.sh N Mx E` does a run with `N` MPI processes, an `Mx` by `Mx` grid, and option `-ssa_e E`. The choices above give a run which only takes a few seconds, and it produces output file `diag_Mx211.nc`.

There are many reasonable choices for the effective softness of an ice shelf, as ice density, temperature, and the presence of fractures all influence the effective softness. Using an enhancement factor `-ssa_e 0.6` acknowledges that the physical justification for tuning the ice softness is uncertain. One could instead use

the temperature itself or the ice density[20] as tuning parameters, and these are worthwhile experiments for the interested PISM user.

The script `plot.py` takes PISM output such as `diag_Mx211.nc` to produce Figure 30. The run shown in the figure used an enhancement factor of 0.6 as above. The thin black line outlines the floating shelf, which is the actual modeling domain here. To generate this Figure yourself, do

```
$ ../plot.py diag_Mx211.nc
```

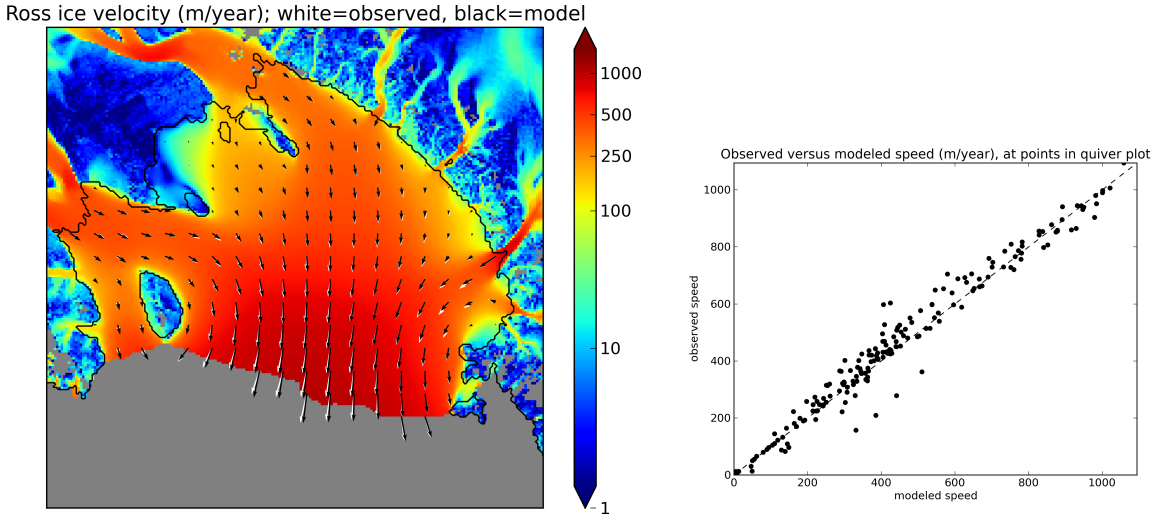


Figure 30: *Left*: Color is speed in m/a. Arrows are observed (white) and modeled (black) velocities. *Right*: Comparison between modeled and observed speeds at points plotted on the left.

Extending this example to other ice shelves

The SSA diagnostic solution described in this section can be easily applied to other ice shelves in Antarctica, such as the Filchner-Ronne Ice Shelf modeled using PISM in [1], for example.

Simply choose a different rectangular domain, within the area covered by the whole-Antarctic datasets used here, at the preprocessing stage. In particular you should modify the lines "`ncks -O -d x1,439,649 -d y1,250,460 ...`" (for ALBMAP data) and "`ncks -d x,2200,3700 -d y,3500,4700 ...`" (for MEaSUREs velocity data) in the script `examples/ross/preprocess.py`.

Prognostic modelling using eigencalving

Next we summarize how you can create an evolving-geometry model of the Ross ice shelf with constant-in-time inflow across the fixed grounding line. See `README.md` and `run_prog.sh` in `examples/ross/prognostic/`. This example also demonstrates the `-calving eigen_calving` model for a moving calving front [67].

[20] High accumulation rates, cold firn with minimal compression, and basal freeze-on of marine ice may all generate significant variation in shelf density.

Start by running `preprocess.py` in `examples/ross/` as described above. If you have already done the diagnostic example above, then this stage is complete.

Then change to the `prognostic/` directory and run the default example:

```
$ cd examples/ross/prognostic/
$ ./run_prog.sh 4 211 0.6 100
```

This 100 model year run on 4 processes and a 5 km grid took about twenty minutes on a 2013 laptop. It starts with a bootstrapping stage which does a `y 0` run, which generates `startfile_Mx211.nc`. It then re-initializes to start the prognostic run itself. See the `README.md` for a bit more on the arguments taken by `run_prog.sh` and on viewing the output files.

The PISM command done here is (essentially, and without showing diagnostic output choices)

```
pismr -i startfile_Mx211.nc -surface given -stress_balance ssa \
    -yield_stress constant -tauc 1e6 -pik -ssa_dirichlet_bc -ssa_e 0.6 \
    -y 100 -o prog_Mx211_yr100.nc -o_order zyx -o_size big \
    -calving eigen_calving,thickness_calving -eigen_calving_K 1e17 \
    -cfl_eigen_calving -thickness_calving_threshold 150.0 \
    -ssafd_ksp_type gmres -ssafd_ksp_norm_type unpreconditioned \
    -ssafd_ksp_pc_side right -ssafd_pc_type asm -ssafd_sub_pc_type lu
```

Several of these options are different from those used in the diagnostic case. First, while the command `-pik` is the same as before, now each part of its expansion, namely `-cfbc -kill_icebergs -part_grid -part_redist`, is important. As the calving front evolves (i.e. regardless of the calving law choices), option `-part_grid` moves the calving front by one grid cell only when the cell is full of the ice flowing into it; see [2]. The option `-kill_icebergs` is essential to maintain well-posedness of the SSA velocity problem at each time step [115]. See section 8.1.

Option combination

```
    -calving eigen_calving,thickness_calving -eigen_calving_K 1e17 \
    -cfl_eigen_calving -thickness_calving_threshold 150.0
```

specifies that ice at the calving front will be removed if either a criterion on the product of principal stresses is satisfied [67], namely `eigen_calving` with the given constant K, or if the ice thickness goes below the given threshold of 150 meters. See subsection 8.3.

There is also an extended option combination

```
    -ssafd_ksp_type gmres -ssafd_ksp_norm_type unpreconditioned \
    -ssafd_ksp_pc_side right -ssafd_pc_type asm -ssafd_sub_pc_type lu
```

which tells the PETSc KSP object used by the SSA solver to solve in the most robust, though not necessarily fastest, way. In particular, the linear problem is spread across processors using an additive Schwarz domain decomposition preconditioning method (`pc_type asm`) [106], along with the standard `gmres` KSP solver, and then on each processor the local part of the linear system is solved by a direct method by the preconditioner (`sub_pc_type lu`). These choices seem to be effective for solving SSA stress balances on the complicated-geometry domains which arise from nontrivial calving laws.

13 Example: A regional model of the Jakobshavn outlet glacier in Greenland

Jakobshavn Isbrae is a fast-flowing outlet glacier in western Greenland that drains approximately 7% of the area of the Greenland ice sheet. It experienced a large acceleration following the loss of its floating tongue in the 1990s [60], an event which seems to have been driven by warmer ocean temperatures [50]. Because it is thick, has a steep surface slope, has a deep trough in its bedrock topography (Figure 31), and has a thick layer of low-viscosity temperate ice at its base [70], this ice flow is different from the ice streams in West Antarctica or Northeast Greenland [107].

This section describes how to build a PISM regional model of this outlet glacier [27] using scripts from `examples/jako/`. The same strategy should work for other outlet glaciers. We also demonstrate the PISM executable `pismo` ("outlet-glacier mode"), and Python drainage-basin-delineation tools `regional-tools` which can be downloaded from the PISM source code website. Such regional models allow modest-size computers to run high resolution models[21] and large ensembles. Regional analysis is justified if detailed data is available for the region.

The geometric data used here is the SeaRISE [12] 1 km dataset for the whole Greenland ice sheet. It contains bedrock topography from recent CReSIS radar in the Jakobshavn area. We also use the SeaRISE 5 km data set which has climatic mass balance from the Greenland-region climate model RACMO [29].

A regional ice flow model generally needs ice flow and stress boundary conditions. For this we use a 5 km grid, whole ice sheet, spun-up model state from PISM, described in Section 2 of this *Manual*. You can download the large NetCDF result from the PISM website, or you can generate it by running a Section 2 script.

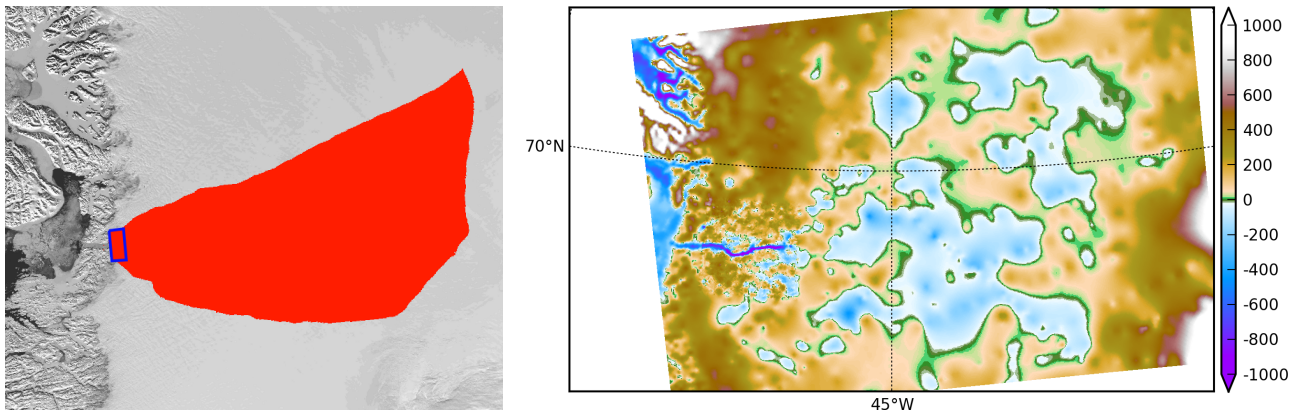


Figure 31: A `regional-tools` script computes a drainage basin mask from the surface DEM (left; Modis background) and from a user-identified terminus rectangle (blue). The regional model can exploit high-resolution bedrock elevations inland from Jakobshavn fjord (right; meters asl).

[21] PISM can also do 1 km runs for the whole Greenland ice sheet; see this news item.

Get the drainage basin delineation tool

The drainage basin tool `regional-tools` is at https://github.com/pism/regional-tools. Get it using `git` and set it up as directed in its `README.md`. Then come back to the `examples/jako/` directory and link the script. Here is the quick summary:

```
$ cd ~/usr/local/                                    # the location you want
$ git clone https://github.com/pism/regional-tools.git
$ cd regional-tools/
$ python setup.py install                            # may add "sudo" or "--user"
$ cd PISM/examples/jako/
$ ln -s ~/usr/local/regional-tools/pism_regional.py .  # symbolic link to tool
```

Preprocess the data and get the whole ice sheet model file

Script `preprocess.sh` downloads and cleans the 1 km SeaRISE data, an 80 Mb file called `Greenland1km.nc`.[22] The script also downloads the SeaRISE 5 km data set `Greenland_5km_v1.1.nc`, which contains the RACMO surface mass balance field (not present in the 1 km data set). If you have already run the example in Section 2 then you already have this file and you can link to it to avoid downloading:

```
$ ln -s ../std-greenland/Greenland_5km_v1.1.nc
```

The same script also preprocesses a pre-computed 5 km grid PISM model result `g5km_gridseq.nc` for the whole ice sheet. This provides the boundary conditions, and the thermodynamical initial condition, for the regional flow model we are building. If you have already generated it by running the script in subsection 2.8 then link to it,

```
$ ln -s ../std-greenland/g5km_gridseq.nc
```

Otherwise running `preprocess.sh` will download it. Because it is about 0.6 Gb this may take some time.

So now let's actual run the preprocessing script:

```
$ ./preprocess.sh
```

Files `gr1km.nc`, `g5km_climate.nc`, and `g5km_bc.nc` will appear. These can be examined in the usual ways, for example:

```
$ ncdump -h gr1km.nc | less          # read metadata
$ ncview gr1km.nc                    # view fields
```

The boundary condition file `g5km_bc.nc` contains thermodynamical spun-up variables (`enthalpy,bmelt,bwat`) and boundary values for the sliding velocity (`u_ssa_bc,v_ssa_bc`); these have been extracted from `g5km_gridseq.nc`.

None of the above actions is specific to Jakobshavn, though all are specific to Greenland. If your goal is to build a regional model of another outlet glacier in Greenland, then you may be able to use `preprocess.sh` as is. The SeaRISE 1 km data set has recent CReSIS bed topography data only for the vicinity of the Jakobshavn outlet, however, and it is otherwise just BEDMAP. Because outlet glacier flows are bed-topography-dominated, additional bed elevation data should be sought.

[22] If this file is already present then no actual download occurs, and preprocessing proceeds. Thus: Do not worry about download time if you need to preprocess again. The same comment applies to other downloaded files.

Identify the drainage basin for the modeled outlet glacier

Here we are going to extract a "drainage basin mask" from the surface elevation data (DEM) in `gr1km.nc`. The goal is to determine, in part, the locations outside of the drainage basin where boundary conditions taken from the precomputed whole ice sheet run can be applied to modeling the outlet glacier flow itself.

The basin mask is determined by the gradient flow of the surface elevation. Thus generating the mask uses a highly-simplified ice dynamics model (namely: ice flows down the surface gradient). Once we have the mask, we will apply the full PISM model in the basin interior marked by the mask. Outside the basin mask we will apply simplified models or use the whole ice sheet results as boundary conditions.

The script `pism_regional.py` computes the drainage basin mask based on a user choice of a "terminus rectangle"; see Figure 31. There are two ways to use this script:

- **To use the graphical user interface (GUI) mode.** Run

  ```
  $ python pism_regional.py
  ```

  Select `gr1km.nc` to open. Once the topographic map appears in the Figure window, you may zoom enough to see the general outlet glacier area. Then select the button "Select terminus rectangle". Use the mouse to select a small rectangle around the Jakobshavn terminus (calving front), or around the terminus of another glacier if you want to model that. Once you have a highlighted rectangle, select a "border width" of at least 50 cells.[23] Then click "Compute the drainage basin mask." Because this is a large data set there will be some delay. (Multi-core users will see that an automatic parallel computation is done.) Finally click "Save the drainage basin mask" and save with your preferred name; we will assume it is called `jakomask.nc`. Then quit.

- **To use the command-line interface.** The command-line interface of `pism_regional.py` allows one to re-create the mask without changing the terminus rectangle choice. (It also avoids the slowness of the GUI mode for large data sets.) In fact, for repeatability, we will assume you have used this command to calculate the drainage basin:

  ```
  $ python pism_regional.py -i gr1km.nc -o jakomask.nc -x 360,382 -y 1135,1176 -b 50
  ```

  This call generates the red region in Figure 31. Options `-x A,B -y C,D` identify the grid index ranges of the terminus rectangle, and option `-b` sets the border width. To see more script options, run with `-help`.

Cut out the computational domain for the regional model

We still need to "cut out" from the whole ice sheet geometry data `gr1km.nc` the computational domain for the regional model. The climate data file `g5km_climate.nc` and the boundary condition file `g5km_bc.nc` do not need this action because PISM's coupling and SSA boundary condition codes already handle interpolation and/or subsampling for such data.

You may have noticed that the text output from running `pism_regional.py` included a cutout command which uses `ncks` from the NCO tools. This command also appears as a global attribute of `jakomask.nc`:

[23] This recommendation is somewhat Jakobshavn-specific. We want our model to have an ice-free down flow (western) boundary on the resulting computational domain for the modeled region.

```
$ ncdump -h jakomask.nc | grep cutout
```

Copy and run the command that appears, something like

```
$ ncks -d x,299,918 -d y,970,1394 gr1km.nc jako.nc
```

This command is also applied to the mask file; note the option `-A` for "append":

```
$ ncks -A -d x,299,918 -d y,970,1394 jakomask.nc jako.nc
```

Now look at `jako.nc`, for example with "`ncview -minmax all jako.nc`". This file is the full geometry data ready for a regional model. The field `ftt_mask` identifies the drainage basin, outside of which we will use simplified time-independent boundary conditions. Specifically, outside of the `ftt_mask` area, but within the computational domain defined by the extent of `jako.nc`, we will essentially keep the initial thickness. Inside the `ftt_mask` area all fields will evolve normally.

Quick start

The previous steps starting with the command "`./preprocess.sh`" above, then using the command-line version of `pism_regional.py`, and then doing the `ncks` cut-out steps, are all accomplished in one script,

```
$ ./quickjakosetup.sh
```

Running this takes about a minute on a fast laptop, assuming data files are already downloaded.

Spinning-up the regional model on a 5 km grid

To run the PISM regional model we will need to know the number of grid points in the 1 km grid in `jako.nc`. Do this:

```
$ ncdump -h jako.nc |head
    netcdf jako {
    dimensions:
      y = 425 ;
      x = 620 ;
    ...
```

The grid has spacing of 1 km, so our computational domain is a 620 km by 425 km rectangle. A 2 km resolution, century-scale model run is easily achievable on a desktop or laptop computer, and that is our goal below. A lower 5 km resolution spin-up run, matching the resolution of the 5 km whole ice sheet state computed earlier, is also achievable on a small computer; we do that first.

The boundary condition fields in `g5km_bc.nc`, from the whole ice sheet model result `g5km_gridseq.nc`, may or may not, depending on modeller intent, be spun-up adequately for the purposes of the regional model. For instance, the intention may be to study equilibrium states with model settings special to the region. Here, however we assume that some regional spin-up is needed, if for no other reason that the geometry used here (from the SeaRISE 1km data set) differs from that in the whole ice sheet model state.

We will get first an equilibrium 5 km regional model, and then do a century run of a 2 km model based on that. While determining "equilibrium" requires a decision, of course, a standard satisfied here is that

the ice volume in the region changes by less than 0.1 percent in the final 100 model years. See `ivol` in `ts_spunjako_0.nc` below.

The 5 km grid[24] uses `-Mx 125 -My 86`. So now we do a basic run using 4 MPI processes:

```
$ ./spinup.sh 4 125 86 &> out.spin5km &
```

You can read the `stdout` log file while it runs: "`less out.spin5km`". The run takes about 5 processor-hours on a 2013 laptop. It produces three files which can be viewed (e.g. with `ncview`): `spunjako_0.nc`, `ts_spunjako_0.nc`, and `ex_spunjako_0.nc`. Some more comments on this run are appropriate:

- Generally the regridding techniques used at the start of this spin-up run are recommended for regional modeling. Read the actual run command by

  ```
  $ PISM_DO=echo ./spinup.sh 4 125 86 | less
  ```

- We use `-i jako.nc -bootstrap`, so we get to choose our grid, and (as usual in PISM with `-bootstrap`) the fields are interpolated to our grid.

- A modestly-fine vertical grid with 20 m spacing is chosen, but even finer is recommended, especially to resolve the temperate ice layer in these outlet glaciers.

- There is an option `-no_model_strip 10` asking `pismo` to put a 10 km strip around edge of the computational domain. This strip is entirely outside of the drainage basin defined by `ftt_mask`. In this strip the thermodynamical spun-up variables `bmelt,tillwat,enthalpy,litho_temp` from `g5km_bc.nc` are held fixed and used as boundary conditions for the conservation of energy model. A key part of putting these boundary conditions into the model strip are the options

  ```
  -regrid_file g5km_bc.nc -regrid_vars bmelt,tillwat,enthalpy,litho_temp,vel_ssa_bc
  ```

- Dirichlet boundary conditions `u_ssa_bc,v_ssa_bc` are also regridded from `g5km_bc.nc` for the sliding SSA stress balance, and the option `-ssa_dirichlet_bc` then uses them during the run. The SSA equations are solved as usual except in the `no_model_strip` where these Dirichlet boundary conditions are used. Note that the velocity tangent to the north and south edges of the computational domain is significantly nonzero, which motivates this usage.

- The calving front of the glacier is handled by the following option combination:

  ```
  -calving ocean_kill -ocean_kill_file jako.nc -pik
  ```

  This choice uses the present-day ice extent, defined by SeaRISE data in `Greenland1km.nc`, to determine the location of the calving front. Recalling that `-pik` includes `-cfbc`, we are applying a PIK mechanism for the stress boundary condition at the calving front. The other PIK mechanisms are largely inactive because of `-calving ocean_kill`, but they should do no harm (see section 8.1).

[24] Calculate `620/5 + 1` and `425/5 + 1`, for example.

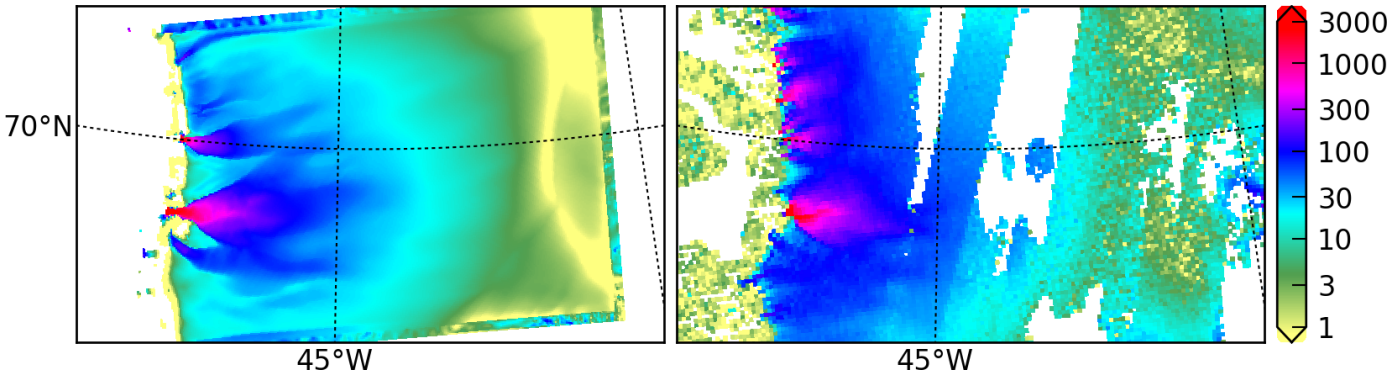


Figure 32: Left: modeled surface speed at the end of a 2 km grid, 100 model year, steady present-day climate run. Right: observed surface speed, an average of four winter velocity maps (2000,2006–2008) derived from RADARSAT data, as included in the SeaRISE 5 km data set [63], for the same region. Scales are in meters per year.

Century run on a 2 km grid

Now that we have a spun-up state, here is a 100 model year run on a 2 km grid with a 10 m grid in the vertical:

```
$ ./century.sh 4 311 213 spunjako_0.nc &> out.2km_100a &
```

This run requires at least 6 GB of memory, and it takes about 16 processor-hours.

It produces a file `jakofine_short.nc` almost immediately and then restarts from it because we need to regrid fields from the end of the previous 5 km regional run (in `spunjako_0.nc`) and then to "go back" and regrid the SSA boundary conditions from the 5 km whole ice sheet results `g5km_bc.nc`. At the end of the run the final file `jakofine.nc` is produced. Also there is a time-series file `ts_jakofine.nc` with monthly scalar time-series and a spatial time-dependent file `ex_jakofine.nc`. The surface speed at the end of this run is shown in Figure 32, with a comparison to observations.

Over this 100 year period the flow appears to be relatively steady state. Though this is not surprising because the climate forcing and boundary conditions are time-independent, a longer run reveals ongoing speed variability associated to subglacially-driven sliding cyclicity; compare [110].

The ice dynamics parameters chosen in `spinup.sh` and `century.sh`, especially the combination

```
-topg_to_phi 15.0,40.0,-300.0,700.0 -till_effective_fraction_overburden 0.02 \
  -pseudo_plastic -pseudo_plastic_q 0.25 -tauc_slippery_grounding_lines
```

are a topic for a parameter study (compare [4]) or a study of their relation to inverse modeling results (e.g. [44]).

References

[1] T. Albrecht and A. Levermann. Fracture field for large-scale ice dynamics. *J. Glaciol.*, 58(207):165–176, 2012.

[2] T. Albrecht, M. Martin, M. Haseloff, R. Winkelmann, and A. Levermann. Parameterization for subgrid-scale motion of ice-shelf calving fronts. *The Cryosphere*, 5:35–44, 2011.

[3] J. M. Amundson, M. Fahnestock, M. Truffer, J. Brown, M. P. Lüthi, and R. J. Motyka. Ice mélange dynamics and implications for terminus stability, Jakobshavn Isbrae, Greenland. *J. Geophys. Res.*, 115, 2010. F01005.

[4] A. Aschwanden, G. Adalgeirsdóttir, and C. Khroulev. Hindcasting to measure ice sheet model sensitivity to initial states. *The Cryosphere*, 7:1083–1093, 2013.

[5] A. Aschwanden and H. Blatter. Mathematical modeling and numerical simulation of polythermal glaciers. *J. Geophys. Res.*, 114, 2009. F01027.

[6] A. Aschwanden, E. Bueler, C. Khroulev, and H. Blatter. An enthalpy formulation for glaciers and ice sheets. *J. Glaciol.*, 58(209):441–457, 2012.

[7] S. Balay et al. PETSc Users Manual. Technical Report ANL-95/11 - Revision 3.5, Argonne National Laboratory, 2014.

[8] R.C. Bales, J.R. McConnell, E. Mosley-Thompson, and G. Lamorey. Accumulation map for the Greenland Ice Sheet: 1971-1990. *Geophys. Res. Lett*, 28(15):2967–2970, 2001. URL: http://zero.eng.ucmerced.edu/rcbales/PARCA/.

[9] J. L. Bamber, D. G. Vaughan, and I. Joughin. Widespread complex flow in the interior of the Antarctic ice sheet. *Science*, 287:1248–1250, 2000.

[10] J.L. Bamber, R.L. Layberry, and S.P. Gogenini. A new ice thickness and bed data set for the Greenland ice sheet 1: Measurement, data reduction, and errors. *J. Geophys. Res.*, 106 (D24):33,773–33,780, 2001.

[11] C. R. Bentley. Glaciological studies on the Ross Ice Shelf, Antarctica, 1973–1978. *Antarctic Research Series*, 42(2):21–53, 1984.

[12] R. Bindschadler and twenty-seven others. Ice-sheet model sensitivities to environmental forcing and their use in projecting future sea-level (The SeaRISE Project). *J. Glaciol*, 59(214):195–224, 2013.

[13] H. Blatter. Velocity and stress fields in grounded glaciers: a simple algorithm for including deviatoric stress gradients. *J. Glaciol.*, 41(138):333–344, 1995.

[14] Jed Brown, Barry Smith, and Aron Ahmadia. Achieving textbook multigrid efficiency for hydrostatic ice sheet flow. *SIAM J. Sci. Comp.*, 35(2):B359–B375, 2013.

[15] E. Bueler. An exact solution to the temperature equation in a column of ice and bedrock. preprint `arXiv:0710.1314`, 2007.

[16] E. Bueler and J. Brown. On exact solutions and numerics for cold, shallow, and thermocoupled ice sheets. preprint `arXiv:physics/0610106`, 2006.

[17] E. Bueler and J. Brown. Shallow shelf approximation as a "sliding law" in a thermodynamically coupled ice sheet model. *J. Geophys. Res.*, 114, 2009. F03008, doi:10.1029/2008JF001179.

[18] E. Bueler, J. Brown, and C. Lingle. Exact solutions to the thermomechanically coupled shallow ice approximation: effective tools for verification. *J. Glaciol.*, 53(182):499–516, 2007.

[19] E. Bueler, C. S. Lingle, and J. A. Kallen-Brown. Fast computation of a viscoelastic deformable Earth model for ice sheet simulation. *Ann. Glaciol.*, 46:97–105, 2007.

[20] E. Bueler, C. S. Lingle, J. A. Kallen-Brown, D. N. Covey, and L. N. Bowman. Exact solutions and numerical verification for isothermal ice sheets. *J. Glaciol.*, 51(173):291–306, 2005.

[21] Ed Bueler. Lessons from the short history of ice sheet model intercomparison. *The Cryosphere Discussions*, 2:399–412, 2008. http://www.the-cryosphere-discuss.net/2/399/2008/.

[22] Ed Bueler, Constantine Khroulev, Andy Aschwanden, Ian Joughin, and Ben E. Smith. Modeled and observed fast flow in the Greenland ice sheet. submitted, 2009.

[23] R. Calov, R. Greve, A. Abe-Ouchi, E. Bueler, P. Huybrechts, J. V. Johnson, F. Pattyn, D. Pollard, C. Ritz, F. Saito, and L. Tarasov. Results from the ice sheet model intercomparison project—Heinrich event intercomparison (ISMIP HEINO). *J. Glaciol*, 56(197):371–383, 2010.

[24] G. K. C. Clarke. Subglacial processes. *Annu. Rev. Earth Planet. Sci.*, 33:247–276, 2005.

[25] G. Cogley et al. Glossary of Mass-Balance and Related Terms. IACS Working Group on Mass-balance Terminology and Methods, Draft 3, 10 July, 2009.

[26] K. M. Cuffey and W. S. B. Paterson. *The Physics of Glaciers*. Elsevier, 4th edition, 2010.

[27] D. DellaGiustina. Regional modeling of Greenland's outlet glaciers with the Parallel Ice Sheet Model. Master's thesis, University of Alaska, Fairbanks, 2011. M.S. Computational Physics.

[28] P. Dickens and T. Morey. Increasing the scalability of PISM for high resolution ice sheet models. In *Proceedings of the 14th IEEE International Workshop on Parallel and Distributed Scientific and Engineering Computing, May 2013, Boston*, 2013.

[29] J. Ettema, M. R. van den Broeke, E. van Meijgaard, W. J. van de Berg, J. L. Bamber, J. E. Box, and R. C. Bales. Higher surface mass balance of the Greenland ice sheet revealed by high-resolution climate modeling. *Geophys. Res. Let.*, 36(L12501), 2009.

[30] J. Feldmann, T. Albrecht, C. Khroulev, F. Pattyn, and A. Levermann. Resolution-dependent performance of grounding line motion in a shallow model compared to a full-Stokes model according to the MISMIP3d intercomparison. *J. Glaciol.*, 60(220):353–360, 2014.

[31] A. C. Fowler. *Mathematical Models in the Applied Sciences*. Cambridge Univ. Press, 1997.

[32] Andrew C. Fowler. Modelling the flow of glaciers and ice sheets. In Brian Straughan et al., editors, *Continuum Mechanics and Applications in Geophysics and the Environment*, pages 201–221. Springer, 2001.

[33] O. Gagliardini and T. Zwinger. The ISMIP-HOM benchmark experiments performed using the Finite-Element code Elmer. *The Cryosphere*, 2(1):67–76, 2008. www.the-cryosphere.net/2/67/2008/.

[34] R. M. Gladstone, A. J. Payne, and S. L. Cornford. Parameterising the grounding line in flow-line ice sheet models. *The Cryosphere*, 4:605–619, 2010.

[35] D. Goldberg, D. M. Holland, and C. Schoof. Grounding line movement and ice shelf buttressing in marine ice sheets. *J. Geophys. Res.*, 114(F04026), 2009.

[36] D. L. Goldsby and D. L. Kohlstedt. Superplastic deformation of ice: experimental observations. *J. Geophys. Res.*, 106(M6):11017–11030, 2001.

[37] N. Golledge, C. Fogwill, A. Mackintosh, and K. Buckley. Dynamics of the Last Glacial Maximum Antarctic ice-sheet and its response to ocean forcing. *Proc. Nat. Acad. Sci.*, 109(40):16052–16056, 2012.

[38] N. Golledge, A. Mackintosh, and 8 others. Last Glacial Maximum climate in New Zealand inferred from a modelled Southern Alps icefield. *Quaternary Science Reviews*, 46:30–45, 2012.

[39] N. Golledge and twelve others. Glaciology and geological signature of the Last Glacial Maximum Antarctic ice sheet. *Quaternary Sci. Rev.*, 78(0):225–247, 2013.

[40] R. Greve. A continuum–mechanical formulation for shallow polythermal ice sheets. *Phil. Trans. Royal Soc. London A*, 355:921–974, 1997.

[41] R. Greve and H. Blatter. *Dynamics of Ice Sheets and Glaciers*. Advances in Geophysical and Environmental Mechanics and Mathematics. Springer, 2009.

[42] Ralf Greve. Glacial isostasy: Models for the response of the Earth to varying ice loads. In Brian Straughan et al., editors, *Continuum Mechanics and Applications in Geophysics and the Environment*, pages 307–325. Springer, 2001.

[43] Ralf Greve, Ryoji Takahama, and Reinhard Calov. Simulation of large-scale ice-sheet surges: The ISMIP-HEINO experiments. *Polar Meteorol. Glaciol.*, 20:1–15, 2006.

[44] M. Habermann, M. Truffer, and D. Maxwell. Changing basal conditions during the speed-up of Jakobshavn Isbrae, Greenland. *The Cryosphere*, 7(6):1679–1692, 2013.

[45] P. Halfar. On the dynamics of the ice sheets 2. *J. Geophys. Res.*, 88(C10):6043–6051, 1983.

[46] R. C. A. Hindmarsh. Thermoviscous stability of ice-sheet flows. *J. Fluid Mech.*, 502:17–40, 2004.

[47] R. C. A. Hindmarsh. Stress gradient damping of thermoviscous ice flow instabilities. *J. Geophys. Res.*, 111(B12409), 2006.

[48] R. C. A. Hindmarsh and A. J. Payne. Time–step limits for stable solutions of the ice–sheet equation. *Ann. Glaciol.*, 23:74–85, 1996.

[49] R. Hock. Glacier melt: a review of processes and their modelling. *Prog. Phys. Geog.*, 29(3):362–391, 2005.

[50] D. M. Holland, R. H. Thomas, B. de Young, M. H. Ribergaard, and B. Lyberth. Acceleration of Jakobshavn Isbræ triggered by warm subsurface ocean waters. *Nature Geoscience*, 1:659–664, 2008.

[51] R. Hooke. Flow law for polycrystalline ice in glaciers: comparison of theoretical predictions, laboratory data, and field measurements. *Rev. Geophys. Space. Phys.*, 19(4):664–672, 1981.

[52] A. Humbert, R. Greve, and K. Hutter. Parameter sensitivity studies for the ice flow of the Ross Ice Shelf, Antarctica. *J. Geophys. Res.*, 110(F04022), 2005.

[53] K. Hutter. *Theoretical Glaciology.* D. Reidel, 1983.

[54] 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.*, 21:203–231, 2002.

[55] P. Huybrechts and J. de Wolde. The dynamic response of the Greenland and Antarctic ice sheets to multiple-century climatic warming. *J. Climate*, 12:2169–2188, 1999.

[56] P. Huybrechts et al. The EISMINT benchmarks for testing ice-sheet models. *Ann. Glaciol.*, 23:1–12, 1996.

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

[58] 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:624–629, 1995.

[59] I. Joughin. Ice-sheet velocity mapping: a combined interferometric and speckle-tracking approach. *Ann. Glaciol.*, 34:195–201, 2002.

[60] I. Joughin, W. Abdalati, and M. Fahnestock. Large fluctuations in speed on Greenland's Jakobshavn Isbræ glacier. *Nature*, 432(23):608–610, 2004.

[61] I. Joughin, M. Fahnestock, S. Ekholm, and R. Kwok. Balance velocities of the Greenland ice sheet. *Geophysical Research Letters*, 24(23):3045–3048, 1997.

[62] I. Joughin, M. Fahnestock, D. MacAyeal, J. L. Bamber, and P. Gogineni. Observation and analysis of ice flow in the largest Greenland ice stream. *J. Geophys. Res.*, 106(D24):34021–34034, 2001.

[63] I. Joughin, B. E. Smith, I. M. Howat, T. Scambos, and T. Moon. Greenland flow variability from ice-sheet-wide velocity mapping. *J. Glaciol.*, 56(197):415–430, 2010.

[64] Ian Joughin, Sarah B. Das, Matt A. King, Ben E. Smith, Ian M. Howat, and Twila Moon. Seasonal Speedup Along the Western Flank of the Greenland Ice Sheet. *Science*, 320(5877):781–783, 2008. http://www.sciencemag.org/cgi/content/abstract/320/5877/781.

[65] E. Larour, H. Seroussi, M. Morlighem, and E. Rignot. Continental scale, high order, high spatial resolution, ice sheet modeling using the Ice Sheet System Model (ISSM). *J. Geophys. Res.*, 117(F01022), 2012.

[66] A. M. Le Brocq, A. J. Payne, and A. Vieli. An improved Antarctic dataset for high resolution numerical ice sheet models (ALBMAP v1). *Earth System Science Data*, 2(2):247–260, 2010.

[67] A. Levermann, T. Albrecht, R. Winkelmann, M. A. Martin, M. Haseloff, and I. Joughin. Kinematic first-order calving law implies potential for abrupt ice-shelf retreat. *The Cryosphere*, 6:273–286, 2012.

[68] C. S. Lingle and J. A. Clark. A numerical model of interactions between a marine ice sheet and the solid earth: Application to a West Antarctic ice stream. *J. Geophys. Res.*, 90(C1):1100–1114, 1985.

[69] L. A. Lliboutry and P. Duval. Various isotropic and anisotropic ices found in glaciers and polar ice caps and their corresponding rheologies. *Annales Geophys.*, 3:207–224, 1985.

[70] M. Lüthi, M. Fahnestock, and M. Truffer. Correspondence: Calving icebergs indicate a thick layer of temperate ice at the base of Jakobshavn Isbrae, Greenland. *J. Glaciol.*, 55(191):563 – 566, 2009.

[71] D. R. MacAyeal. Large-scale ice flow over a viscous basal sediment: theory and application to ice stream B, Antarctica. *J. Geophys. Res.*, 94(B4):4071–4087, 1989.

[72] D. R. MacAyeal, V. Rommelaere, Ph. Huybrechts, C.L. Hulbe, J. Determann, and C. Ritz. An ice-shelf model test based on the Ross ice shelf. *Ann. Glaciol.*, 23:46–51, 1996.

[73] M. W. Mahaffy. A three–dimensional numerical model of ice sheets: tests on the Barnes Ice Cap, Northwest Territories. *J. Geophys. Res.*, 81(6):1059–1066, 1976.

[74] 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*, 5:727–740, 2011.

[75] L. W. Morland. Unconfined ice-shelf flow. In C. J. van der Veen and J. Oerlemans, editors, *Dynamics of the West Antarctic ice sheet*, pages 99–116. Kluwer Academic Publishers, 1987.

[76] L. W. Morland and R. Zainuddin. Plane and radial ice-shelf flow with prescribed temperature profile. In C. J. van der Veen and J. Oerlemans, editors, *Dynamics of the West Antarctic ice sheet*, pages 117–140. Kluwer Academic Publishers, 1987.

[77] K. W. Morton and D. F. Mayers. *Numerical Solutions of Partial Differential Equations: An Introduction*. Cambridge University Press, 2nd edition, 2005.

[78] W. S. B. Paterson. *The Physics of Glaciers*. Pergamon, 3rd edition, 1994.

[79] W. S. B. Paterson and W. F. Budd. Flow parameters for ice sheet modeling. *Cold Reg. Sci. Technol.*, 6(2):175–177, 1982.

[80] F. Pattyn, L. Perichon, G. Durand, and 25 others. Grounding-line migration in plan-view marine ice-sheet models: results of the ice2sea MISMIP3d intercomparison. *J. Glaciol.*, 59(215):410–422, 2013.

[81] F. Pattyn, C. Schoof, L. Perichon, and 15 others. Results of the Marine Ice Sheet Model Intercomparison Project, MISMIP. *The Cryosphere*, 6:573–588, 2012.

[82] F. Pattyn and twenty others. Benchmark experiments for higher-order and full Stokes ice sheet models (ISMIP-HOM). *The Cryosphere*, 2:95–108, 2008.

[83] Frank Pattyn. A new three-dimensional higher-order thermomechanical ice sheet model: Basic sensitivity, ice stream development, and ice flow across subglacial lakes. *J. Geophys. Res.*, 108(B8), 2003.

[84] A. Payne. EISMINT: Ice sheet model intercomparison exercise phase two. Proposed simplified geometry experiments. homepages.vub.ac.be/∼phuybrec/eismint/thermo-descr.pdf, 1997.

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

[86] A. J. Payne and D. J. Baldwin. Analysis of ice–flow instabilities identified in the EISMINT intercomparison exercise. *Ann. Glaciol.*, 30:204–210, 2000.

[87] A. J. Payne and P. W. Dongelmans. Self–organization in the thermomechanical flow of ice sheets. *J. Geophys. Res.*, 102(B6):12219–12233, 1997.

[88] W. T. Pfeffer, J. T. Harper, and S. O'Neel. Kinematic constraints on glacier contributions to 21st-century sea-level rise. *Science*, 321:1340–1343, 2008.

[89] D. Pollard and R. M. DeConto. A coupled ice-sheet/ice-shelf/sediment model applied to a marine-margin flowline: Forced and unforced variations. In M. J. Hambrey et al., editors, *Glacial Sedimentary Processes and Products*. Blackwell Publishing Ltd., 2007. Special Publication Number 39 of the International Association of Sedimentologists.

[90] S. Price, A. Payne, I. Howat, and B. Smith. Committed sea-level rise for the next century from Greenland ice sheet dynamics during the past decade. *Proc. Nat. Acad. Sci.*, 108(22):8978–8983, 2011.

[91] E. Rignot, J. Mouginot, and B. Scheuchl. Ice flow of the Antarctic Ice Sheet. *Science*, 333(6048):1427–1430, 2011.

[92] P.J. Roache. *Verification and Validation in Computational Science and Engineering.* Hermosa Publishers, Albuquerque, New Mexico, 1998.

[93] V. Rommelaere and D. R. MacAyeal. Large-scale rheology of the Ross Ice Shelf, Antarctica, computed by a control method. *Ann. Glaciol.*, 24:43–48, 1997.

[94] F. Saito, A. Abe-Ouchi, and H. Blatter. European Ice Sheet Modelling Initiative (EISMINT) model intercomparison experiments with first-order mechanics. *J. Geophys. Res.*, 111(F02012), 2006.

[95] F. Saito, A. Abe-Ouchi, and H. Blatter. An improved numerical scheme to compute horizontal gradients at the ice-sheet margin: its effect on the simulated ice thickness and temperature. *Ann. Glaciol.*, 46:87–96, 2007.

[96] R. Sayag, S. S. Pegler, and M. G. Worster. Floating extensional flows. *Physics of Fluids*, 24(9), 2012.

[97] R. Sayag and M. G. Worster. Axisymmetric gravity currents of power-law fluids over a rigid horizontal surface. *J. Fluid Mech.*, 716, 2013.

[98] C. Schoof. The effect of basal topography on ice sheet dynamics. *Continuum Mech. Thermodyn.*, 15:295–307, 2003.

[99] C. Schoof. A variational approach to ice stream flow. *J. Fluid Mech.*, 556:227–251, 2006.

[100] C. Schoof. Variational methods for glacier flow over plastic till. *J. Fluid Mech.*, 555:299–320, 2006.

[101] C. Schoof. Marine ice-sheet dynamics. Part 1. The case of rapid sliding. *J. Fluid Mech.*, 573:27–55, 2007.

[102] C. Schoof. Coulomb friction and other sliding laws in a higher order glacier flow model. *Math. Models Methods Appl. Sci. (M3AS)*, 20:157–189, 2010.

[103] C. Schoof, I. J. Hewitt, and M. A. Werder. Flotation and free surface flow in a model for subglacial drainage. Part I: Distributed drainage. *J. Fluid Mech.*, 702:126–156, 2012.

[104] C. Schoof and R. Hindmarsh. Thin-film flows with wall slip: an asymptotic analysis of higher order glacier flow models. *Quart. J. Mech. Appl. Math.*, 63(1):73–114, 2010.

[105] M. Siegert, A. Le Brocq, and A. Payne. *Hydrological connections between Antarctic subglacial lakes, the flow of water beneath the East Antarctic Ice Sheet and implications for sedimentary processes*, pages 3–10. Wiley-Blackwell, 2009.

[106] B. Smith, P. Bjorstad, and W. Gropp. *Domain decomposition: parallel multilevel methods for elliptic partial differential equations.* Cambridge University Press, 1996.

[107] M. Truffer and K. Echelmeyer. Of isbrae and ice streams. *Ann. Glaciol.*, 36(1):66–72, 2003.

[108] S. Tulaczyk, W. B. Kamb, and H. F. Engelhardt. Basal mechanics of Ice Stream B, West Antarctica 1. Till mechanics. *J. Geophys. Res.*, 105(B1):463–481, 2000.

[109] S. Tulaczyk, W. B. Kamb, and H. F. Engelhardt. Basal mechanics of Ice Stream B, West Antarctica 2. Undrained plastic bed model. *J. Geophys. Res.*, 105(B1):483–494, 2000.

[110] W. J. J. van Pelt and J. Oerlemans. Numerical simulations of cyclic behaviour in the parallel ice sheet model (pism). *Journal of Glaciology*, 58(208):347–360, 2012.

[111] W. J. J. van Pelt, J. Oerlemans, C. H. Reijmer, R. Pettersson, V. A. Pohjola, E. Isaksson, and D. Divine. An iterative inverse method to estimate basal topography and initialize ice flow models. *The Cryosphere*, 7(3):987–1006, 2013.

[112] M. Weis, R. Greve, and K. Hutter. Theory of shallow ice shelves. *Continuum Mech. Thermodyn.*, 11(1):15–50, 1999.

[113] Pieter Wesseling. *Principles of Computational Fluid Dynamics*. Springer-Verlag, 2001.

[114] R. Winkelmann, A. Levermann, K. Frieler, and M.A. Martin. Increased future ice discharge from antarctica owing to higher snowfall. *Nature*, 492:239–242, 2012.

[115] R. Winkelmann, M. A. Martin, M. Haseloff, T. Albrecht, E. Bueler, C. Khroulev, and A. Levermann. The Potsdam Parallel Ice Sheet Model (PISM-PIK) Part 1: Model description. *The Cryosphere*, 5:715–726, 2011.

General Index

bootstrapping, 35
  preparing data using MATLAB, 93

Configuration flags and parameters
  `calendar`, 40
  `reference_date`, 40
CReSIS bedrock topography for Jakobshavn, 118

earth deformation, 61
EISMINT, 103
  defined, 102
  unpublished additional EISMINT II experiments, 103
enhancement factor, 50
executables
  `pisms`, 34, 87
  `pismv`, 88
    options to run verification tests, 98
  python scripts
    `vfnow.py`, 98
    `vnreport.py`, 98

flow law, 48, 49
flow law exponent, 50

GPL (*GNU Public License*), 3

Ice Sheets
  Antarctic ice sheet, 27
    Ross ice shelf, 113
  Greenland ice sheet, 27
initialization
  by bootstrapping, 35
  from saved model state, 34
ISMIP
  defined, 102
  interpretation of HEINO results, 102
  MISMIP, 102, 104
  MISMIP3d, 102, 107

Jakobshavn, 118

mask, 65, 66

NCO (NetCDF Operators), 26
  ncks, 83
  wildcards, 84
NetCDF, 34
  tools
    `ncdump`, 26
    `ncgen`, 26
    `ncview`, 26
    CDO, 26
    IDV, 26
    NCL, 26
    NCO, 26
    PyNGL, 26
    work with wildcards, 84
NetCDF variables
  `time`, 40

Organizations
  Arctic Region Supercomputing Center (ARSC), 3
  Geophysical Institute
    Snow, Ice, and Permafrost group, 3
  NASA
    Cryospheric Sciences program, 3
    Modeling, Analysis, and Prediction program, 3

parallelization
  relative ease of, between SIA and SSA, 27
Parameterization of bed roughness, 62
PISM
  Source Code Browser (HTML), 6
  `pism_intent` attribute, 35
  adaptive time stepping scheme, 89
  catches signals -TERM,-USR1,-USR2, 88
  computational box, 37
  diagnostic quantities, 76

diagnostic Ross ice shelf model, 113
diagnostic run, 29
download source code, 6
earth deformation models, using, 61
evolution run, 29
getting started, 7
grid, 37
hierarchy of simplifying assumptions, 28
install, 6
marine ice sheet modeling, 63
modeling the age of the ice, 52
NetCDF file format, 34
physical meaning of scalar diagnostics, 72
PISM-PIK, 63
pismo executable for outlet glaciers, 118
regional model example, 118
running the EISMINT II experiments in, 104
saving diagnostic quantities regularly, 76
saving snapshots of the model state, 83
saving time-series, 71
SIA, 27
SSA, 27
uses UDUNITS when reading NetCDF files, 36
validation of SIA flow model, 111
validation of SSA ice shelf model, 113
verification of, 96
verification results and reporting, 98
warning, 6
python scripts, 93

refinement path
definition, 97
example, 97
refining the grid, 88
regional-tools, 118
regrid, 88
rheology, 48, 49
Ross ice shelf, 113

SeaRISE
data, 7
group, 3
SIA (shallow ice approximation), 27
applicability, 27
sliding laws, 27, 44
signals, 88
pid (process id), 89
term, 89
USR1, 89
USR2, 89
snapshots of the model state, 83
SSA (shallow shelf approximation), 27, 44
applicability, 27

Time
calendars, 39
time-series, 71

UDUNITS-2, 36

validation
SIA flow model, 111
SSA ice shelf model, 113
validation versus verification, 96
verification tests, 98

PISM Command-line options

Adaptive time-stepping
- `-adapt_ratio`, 92
- `-dt_force` (years), 92
- `-max_dt` (years), 92
- `-skip_max`, 90, 92
- `-skip`, 90, 92
- `-timestep_hit_multiples` (years), 92

Basal strength and sliding
- `-plastic_phi` (degrees), 56
- `-plastic_reg` (m/a), 54
- `-pseudo_plastic_q`, 54
- `-pseudo_plastic_uthreshold` (m/a), 54
- `-pseudo_plastic`, 54
- `-stress_balance ssa+sia`, 53
- `-tauc_add_transportable_water`, 57
- `-tauc_slippery_grounding_lines`, 56
- `-tauc`, 56
- `-till_cohesion`, 55, 56
- `-till_compressibility_coefficient`, 58
- `-till_effective_fraction_overburden`, 58
- `-till_reference_void_ratio`, 58
- `-topg_to_phi` *list of 4 numbers*, 56
- `-yield_stress constant`, 56
- `-yield_stress mohr_coulomb`, 56

Bootstrapping, *see* Grid, *see* Input and output
- `-boot_temperature_heuristic quartic_guess`, 36
- `-bootstrap`, 35
- `-x_range`, 35
- `-y_range`, 35

Computational box, *see* Grid
- `-Lbz` (meters), 38
- `-Lx` (km), 38
- `-Ly` (km), 38
- `-Lz` (meters), 38

Computing ice age
- `-age`, 52

Disabling PISM components
- `-energy cold`, 42
- `-energy none`, 42
- `-no_mass`, 42
- `-stress_balance none`, 42

Driving stress computation
- `-gradient`, 51

Earth deformation models
- `-bed_def` [`iso, lc`], 61

EISMINT II
- `-Mmax`, 105
- `-Rel`, 105
- `-ST`, 105
- `-Sb`, 105
- `-Tmin`, 105
- `-bmr_in_cont`, 105
- `-eisII`, 105

Energy conservation
- `-energy cold`, 52
- `-energy none`, 52

Flow-line modeling
- `-periodicity`, 94

Grid
  - domain distribution
    - `-Nx`, 39
    - `-Ny`, 39
    - `-procs_x`, 39
    - `-procs_y`, 39
  - space
    - `-Mbz`, 38
    - `-Mx`, 38
    - `-My`, 38
    - `-Mz`, 38
    - `-z_spacing` [`quadratic`, `equal`], 37
  - time
    - `-calendar`, 40

-time_file_continue_run, 41
-time_file *filename*, 41
-ye, 41
-ye (years), 39
-ys, 41
-ys (years), 39
-y, 41
-y (years), 39

Ice rheology
-sia_e, 48, 50
-sia_flow_law, 47
-sia_n, 48, 50
-ssa_e, 48, 50
-ssa_flow_law, 47
-ssa_n, 48, 50

Initialization, *see* Input and output
-bootstrap, 34
-dontreadSSAvels, 34
-i *filename*, 34

Input and output, *see* Bootstrapping, *see* Regridding
-bootstrap, 69
-dontreadSSAvels, 69
-help, 70
-info, 70
-i *filename*, 69
-list_diagnostics, 70
-log_summary, 70
-o_format, 71
-o_order, 71
-o_size ⟨size⟩, 69
-options_left, 70
-o *filename*, 69
-usage, 70
-verbose, 70
-version, 70

Jakobshavn
-no_model_strip, 122

Marine ice sheets
-cfbc, 63
-kill_icebergs, 63
-no_subgl_basal_melt, 63
-part_grid, 63
-part_redist, 63
-pik, 63, 64
-ssa_method, 64
-subgl, 63
Calving
-calving eigen_calving, 66, 68
-calving float_kill, 67, 68
-calving ocean_kill, 67, 68
-calving thickness_calving, 67, 68
-cfl_eigen_calving, 66, 68
-eigen_calving_K, 66
-eigen_calving_K (K), 68
-ocean_kill_file *filename*, 67, 68
-thickness_calving_threshold, 67
-thickness_calving_threshold (m), 68
Calving front motion
-part_grid, 64
-part_redist, 64
Calving front stresses
-cfbc, 64
-ssa_method, 64
Grounding line
-no_subgl_basal_melt, 65
-subgl, 65
Icebergs
-kill_icebergs, 65
Mask
-dry, 66
-stress_balance ssa+sia, 66
-stress_balance ssa, 66

Parameterization of bed roughness
-bed_smoother_range, 62

PETSc options for PISM users
-ssafd_ksp_rtol, 91
-ssafd_ksp_type, 91
-ssafd_pc_type, 92

PISM configuration file
-config_override *filename*, 86

-config *filename*, 85

Regridding, *see* Bootstrapping
-regrid_file *filename*, 87
-regrid_vars *comma-separated list*, 87
Ross ice shelf
-calving eigen_calving, 116
-ssa_dirichlet_bc, 115
Run-time diagnostic viewers
-display, 85
-id XX, 85
-jd YY, 85
-ksp_monitor_draw, 85
-ssa_nuh_viewer_size *number*, 85
-ssa_view_nuh, 85
-view_map *comma-separated list*, 85
-view_size number, 85

Saving 2D and 3D time-series
-extra_append, 77
-extra_file *filename*, 77
-extra_split, 77
-extra_times, 76
-extra_times *range or list*, 77
-extra_vars *comma-separated list*, 77
-list_diagnostics, 77
-save_times, 76
-ts_times, 76
Saving scalar time-series
-list_diagnostics, 71
-ts_append, 72
-ts_file *filename*, 72
-ts_times *range or list*, 72
-ts_vars *comma-separated list*, 72
Saving snapshots
-backup_interval hours, 84
-save_file *filename*, 84
-save_size [none,small,medium,big], 84
-save_split, 84
-save_times *range or list*, 84
Stress balance
-prescribed_sliding_file *filename*, 45
-ssa_eps, 45
-ssa_maxi, 46
-ssa_method, 45
-ssa_rtol, 46
-ssa_view_nuh, 45
-ssafd_ksp_rtol, 46
-stress_balance prescribed_sliding+sia, 45
-stress_balance prescribed_sliding, 45
-stress_balance none, 45
-stress_balance sia, 45
-stress_balance ssa+sia, 45
-stress_balance ssa, 45
Subglacial hydrology
-hydrology null, 58
-hydrology routing, 58
-hydrology_bmelt_file, 58
-hydrology_bmelt_file *filename*, 59
-hydrology_const_bmelt (m/a), 59
-hydrology_gradient_power_in_flux β, 61
-hydrology_hydraulic_conductivity k, 61
-hydrology_input_to_bed_file, 58
-hydrology_input_to_bed_file *filename*, 59
-hydrology_input_to_bed_period (a), 59
-hydrology_input_to_bed_reference_year (a), 59
-hydrology_null_strip (km), 61
-hydrology_thickness_power_in_flux α, 61
-hydrology_tillwat_decay_rate (m/a), 59
-hydrology_tillwat_max, 60
-hydrology_tillwat_max (m), 59
-hydrology_use_const_bmelt, 59
-report_mass_accounting, 61

Verification
-eo, 98
-no_report, 98
-test, 98