Ice Sheet Modeling: Numerics and Visualization

Benjamin Sperisen

August 14, 2008

1 Introduction

My internship this summer has focused on two topics: learning about the math involved in ice sheet modeling and visualizing the results of PISM, a Parallel Ice Sheet Model. There are many hurdles involved in modeling ice sheets (and fluid dynamics), but this paper simply gives an introduction to the mathematical method used to solve the equations governing ice sheet flow. It then discusses how to create several different kinds of plots and animations using IDV, a visualization application used for viewing a wide variety of data.

Ice sheets are massive glaciers covering vast areas of land. There are two ice sheets on Earth: one covering Antarctica and another in Greenland. Together they cover 10% of Earth's surface (Alley, 2000). Ice exists in smaller glaciers in various mountain ranges, but they are dwarfed by the Antarctic and Greenland ice sheets, which contain 99% of the world's ice. This is enough ice to cause a 200 ft rise in sea level were it to completely melt, but such an extreme catastrophe is not anticipated by scientists over at least the next several hundred years (Alley, 2000). Sea level rise does concern scientists, but they also worry about other consequences of melting ice sheets. For example, as the Greenland ice sheet melts, cold fresh water enters the North Atlantic Ocean. A "conveyer belt" of current transports warm water from the equator to the North Atlantic along the coast of Western Europe, giving Europe a more temperate climate than land at similar lattitudes in North America. The arrival of this meltwater from Greenland could intefere with the conveyer, producing large changes in climate, especially for Europe.

This has motivated the development of several ice sheet models. Global climate models currently lack ice sheet models, and PISM aims to be accurate enough to be included in these models. PISM is unique among ice sheet models in being both parallel and publicly accessible, and it is particularly good at modeling "basal sliding," where ice slides over bedrock while lubricated by meltwater. This contrasts with ice flow where the ice sheet is frozen to the bedrock; that flow is governed by the more general thermocoupled version of the ice-sheet equation discussed in this paper.

The reason why ice sheet models have taken a while to mature is that they tackle a very hard problem. Section 2 attempts to give a taste of how hard this problem is. It begins with a simple example of a numerical solution to a partial differential equation. It goes on to show how a model of an isothermal ice sheet (i.e., all the ice is at the same temperature) can be constructed using similar principles. It also touches on the problem of making sure that errors in the model do not blow up and make results useless.

Section 3 tries to make the hard problem of ice sheet modeling slightly easier for PISM users by documenting how they can produce more sophisticated visualizations in IDV. Among the visualizations discussed are 2D plots, 3D plots (both simple and colored by a second variable), and isosurfaces. In particular, glaciologists have expressed interest in having a 3D plot of ice thickness colored by surface ice velocity. Isosurfaces of temperature are also helpful because knowing where ice that is at the melting temperature (which can change depending on pressure) can indicate where basal sliding is. Such tools can make it easier to identify bugs and problems with the model.

2 Numerical Solutions to Partial Differential Equations

The answers to many scientific problems, including ice sheet modeling, lie in the solutions of partial differential equations (PDEs). First, a numerical solution is found for the heat equation and the requirement for stability of the solution are discussed. The same is then done for the more complicated ice-sheet equation.

2.1 The Two-Dimensional Heat Equation

A canonical example of a PDE is the heat equation

$$\frac{\partial u}{\partial t} = \alpha \nabla^2 u,$$

where u is the temperature at a given point, t is time, and α is some constant dependent on the material of concern. For simplicity, consider the two dimensional heat equation where $\alpha = 1$:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}. \tag{1}$$

We may want to know the solution to this equation subject to certain boundary conditions, where the temperature at certain points is given. For example, consider a square plate of side length 1, one corner at the origin and the opposite corner at point $(1, 1)$, with the temperature held at 0 at the sides: $u(x, y, t) = 0$ for $x = 0$, $x = 1$, $y = 0$, or $y = 1$, and any t. Let us also use the initial condition

$$u(x, y, 0) = \delta(x - \tfrac{1}{2})\delta(y - \tfrac{1}{2}),$$

where δ is the Dirac delta.

For most practical problems involving PDEs, numerical solutions are a necessity. The difficulty with analytical solutions to PDEs is that they are hard to find for complicated initial or boundary conditions and for nonlinear PDEs (as is the case with real ice sheets). However, they allow us to verify the accuracy of numerical solutions through comparison with known exact solutions for simple problems. This helps us trust the results that numerical solutions give for problems which do not have analytical solutions.

2.1.1 Intuition

I usually find it helpful to qualitatively understand what an equation means before actually trying to solve it. This gives me an idea of what to expect in a solution. For the moment, let's simplify equation (1) into a one dimensional equation:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}. \tag{2}$$

The term $\partial u/\partial t$ tells us what u will do in the (near) future, and $\partial^2 u/\partial x^2$ tells us how u is "bent." We can read (2) as "the more u is bent, the faster it will flatten out." This effect is illustrated in Figure 1. Another way to think of this is that if a point has a higher or lower temperature than its neighbors, its temperature will move toward that of its neighbors (and vice versa); that is, heat flows from hot to cold. We will see that the numerical solution effectively is updating each point with a weighted average of the temperatures of its neighbors and itself.

2.1.2 Numerical Solution

A numerical solution involves a two dimensional grid I columns and J rows, with spacings Δx between the columns and Δy between the rows. Let u_{ij} be the temperature at the point in the ith column and jth row in the grid, where $u_{ij} = u(\Delta x(i-1), \Delta y(j-1))$. We increment time by a time step Δt and update the

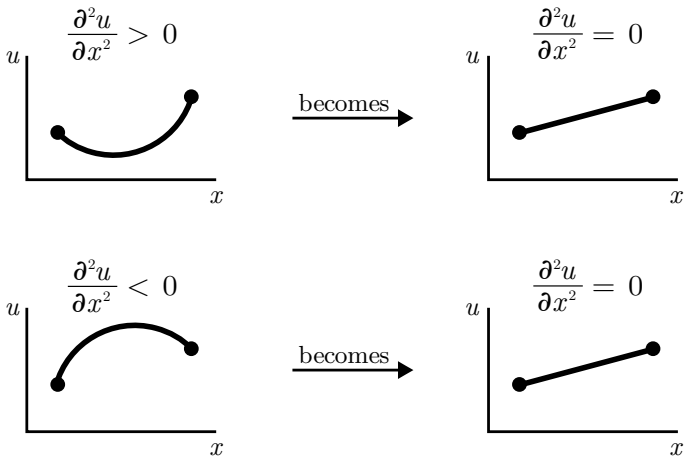

Figure 1: According to the heat equation, "bumps" in temperature will smoothen out.

grid's temperature according to an approximation of the heat equation. The partial derivative with respect to t can be approximated as

$$\frac{u(x,y,t+\Delta t)-u(x,y,t)}{\Delta t} = \frac{u_{ij,k+1}-u_{ijk}}{\Delta t}, \qquad x=(i-1)\Delta x,\ y=(j-1)\Delta y,\ t=k\Delta t,$$

which is called a "forward difference," as opposed to the "backward difference" $(u_{ijk}-u_{ij,k-1})/\Delta t$. To find an approximation of the second derivatives on the right hand side of (1), we begin with the "central difference"

$$\frac{u(x+\frac{1}{2}\Delta x,y)-u(x-\frac{1}{2}\Delta x,y)}{\Delta x}.$$

Applying the central difference twice we find

$$\begin{aligned}\frac{\partial^2 u}{\partial x^2} &\approx \frac{\frac{u(x+\Delta x,y)-u(x,y)}{\Delta x}-\frac{u(x,y)-u(x-\Delta x,y)}{\Delta x}}{\Delta x}\\ &= \frac{u(x+\Delta x,y)-2u(x,y)+u(x-\Delta x,y)}{(\Delta x)^2}\\ &= \frac{u_{i+1,j}-2u_{ij}+u_{i-1,j}}{(\Delta x)^2}.\end{aligned}$$

Thus, we can approximate the heat equation as

$$\frac{u_{ij,k+1}-u_{ijk}}{\Delta t} = \frac{u_{i+1,j}-2u_{ij}+u_{i-1,j}}{(\Delta x)^2}+\frac{u_{i,j+1}-2u_{ij}+u_{i,j-1}}{(\Delta y)^2},$$

where u is at the kth time step unless otherwise specified. If we let $\Delta x=\Delta y$ and solve for $u_{ij,k+1}$, we find

$$u_{ij,k+1} = \frac{\Delta t}{(\Delta x)^2}(u_{i+1,jk}+u_{i-1,jk}+u_{i,j+1,k}+u_{i,j-1,k})+\left(1-4\frac{\Delta t}{(\Delta x)^2}\right)u_{ijk}. \tag{3}$$

Equation (3) can easily be put into a for loop in a computer program and iterated over as needed. Note that, as long as $(1-4\Delta t/\Delta x^2)\geq 0$, $u_{ij,k+1}$ is a weighted average of u_{ij} and its neighbors.

2.1.3 Stability Requirements

When looking at (3) for the first time, one might think that Δt and $\Delta x=\Delta y$ can be set to some reasonably small values, and as Δt and Δx shrink, the numerical solution will come closer and closer to the exact solution.

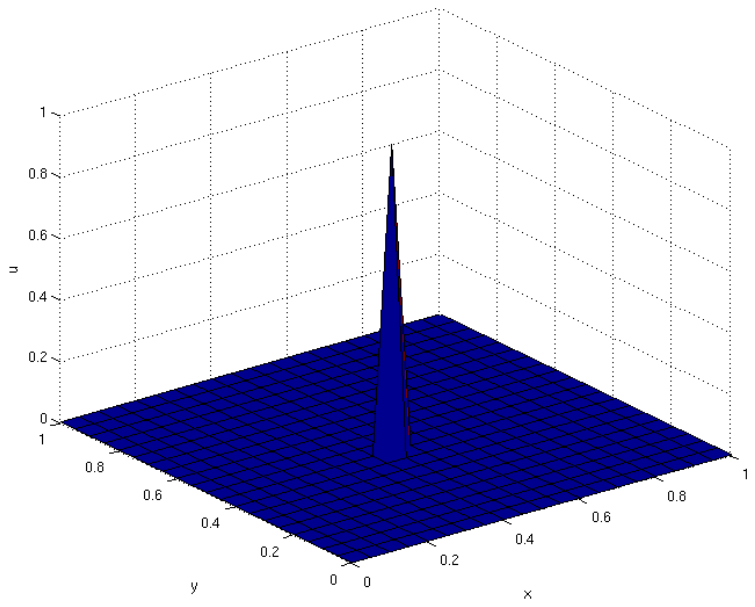

Figure 2: We approximate the initial conditions by setting all u at all points to 0 except for the center point, which is set to 1.

But what if the coefficient $(1 - 4\Delta t/(\Delta x)^2)$ is negative? We can see that the numerical solution is no longer a real average: the coefficients of the neighbor terms add up to more than one, so the coefficient for u_{ijk} subtracts to get to one. Suppose $\Delta t/(\Delta x)^2 = 1$. If $u_{ijk} = 1$ and $u_{i+1,jk} = u_{i-1,jk} = u_{i,j+1,k} = u_{i,j-1,k} = 0$, then $u_{i+1,j,k+1} = u_{i-1,j,k+1} = u_{i,j+1,k+1} = u_{i,j-1,k+1} = 1$ while $u_{ij,k+1} = -3$. We do have conservation of energy (the temperatures of the five points still add up to 1), but we see the unrealistic result of more heat flowing out of u_{ij} than it actually contains, so it has to become negative to conserve heat. We are forced to make sure our numerical solution meets the restriction

$$\frac{\Delta t}{(\Delta x)^2} \leq \tfrac{1}{4}. \tag{4}$$

Using the initial condition in Figure 2, we can see the difference between meeting condition (4) (in Figure (3)) and not meeting the condition (in Figure(4)).

2.2 The Isothermal Ice-Sheet Equation

The isothermal ice-sheet equation is also two-dimensional but more complicated:

$$\frac{\partial H}{\partial t} = M + \nabla \cdot (\Gamma H^{n+2} |\nabla H|^{n-1} \nabla H),$$

where H is the thickness of the ice, M is accumulation (snowfall), Γ is some constant, and n is a fixed exponent in the range $1.8 \leq n < 4$ (Bueler et al., 2005). In particular, the term $\nabla \cdot (\Gamma H^{n+2} |\nabla H|^{n-1} \nabla H)$ presents a challenge. One might first try writing it as

$$\Gamma \nabla \cdot \left(H^{n+2} \left| \tfrac{\partial H}{\partial x}\hat{\mathbf{x}} + \tfrac{\partial H}{\partial y}\hat{\mathbf{y}} \right|^{n-1} \left[\tfrac{\partial H}{\partial x}\hat{\mathbf{x}} + \tfrac{\partial H}{\partial y}\hat{\mathbf{y}} \right] \right)$$

$$= \Gamma \left[\tfrac{\partial}{\partial x} \left(H^{n+2} \left[\left(\tfrac{\partial H}{\partial x}\right)^2 + \left(\tfrac{\partial H}{\partial y}\right)^2 \right]^{\frac{1}{2}(n-1)} \tfrac{\partial H}{\partial x} \right) + \tfrac{\partial}{\partial y} \left(H^{n+2} \left[\left(\tfrac{\partial H}{\partial x}\right)^2 + \left(\tfrac{\partial H}{\partial y}\right)^2 \right]^{\frac{1}{2}(n-1)} \tfrac{\partial H}{\partial y} \right) \right],$$

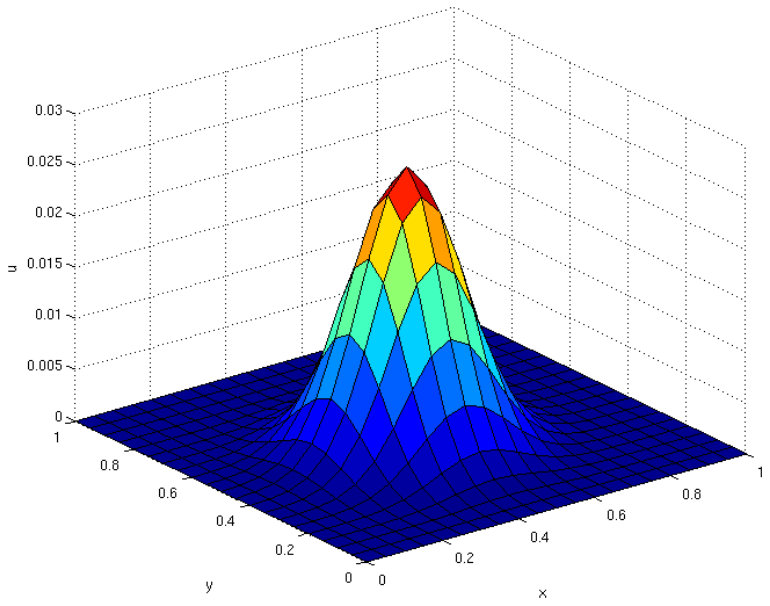

Figure 3: The result after 15 time steps for $\Delta t/(\Delta x)^2 = \frac{1}{5}$.

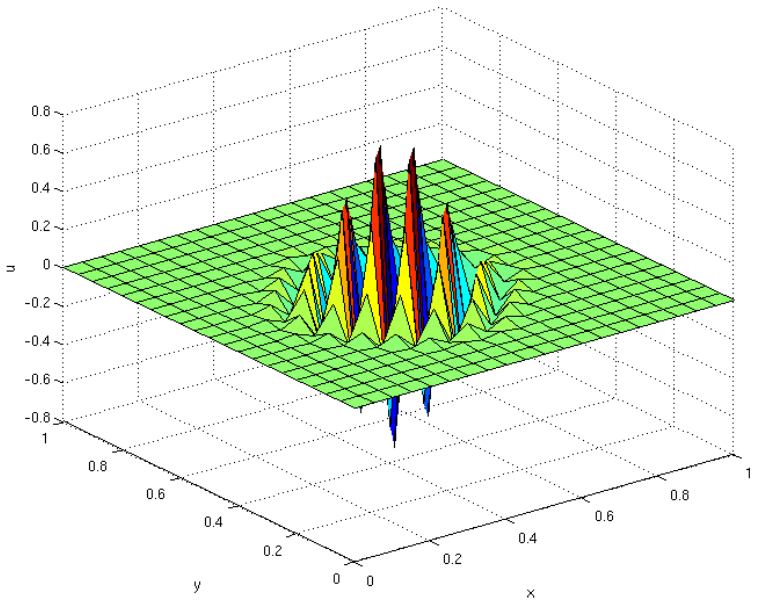

Figure 4: The result of 10 time steps for $\Delta t/(\Delta x)^2 = \frac{3}{10}$.

taking the partial derivatives with respect to x and y, and finally using the forward differences to approximate the first derivative terms and central differences for second derivatives. This tedious method produces a very complicated expression which we would like to avoid. Fortunately, there is a more elegant solution.

2.2.1 Intuition

The ice-sheet equation is quite a bit harder to understand intuitively than the heat equation. Like the heat equation, the left hand side tells us what the will happen to the height of the ice in the near future. The fact that M is on the right hand side makes perfect sense: more snowfall means higher ice.

The $\nabla \cdot (\Gamma H^{n+2} |\nabla H|^{n-1} \nabla H)$ term is much more mysterious. We can take a crack at it by lumping together several scalars: $D = \Gamma H(x,y)^{n+2} |\nabla H(x,y)|^{n-1}$. Here we see that D is big where the ice is tall (the H^{n+2} term) and where the ice is steep (the $|\nabla H(x,y)|^{n-1}$ term). If we consider this and the fact that ∇H is pointing in the steepest direction (and has the magnitude of that steepness), we can imagine that the $\nabla \cdot (D \nabla H)$ term is most positive in valleys next to steep slopes where the ice gets much thicker and most negative at tall, steep peaks. This makes sense because the thick ice means larger pressures within the ice. Ice is about as dense as water, so swimming to the bottom of a somewhat deep pool will give a feel for how quickly pressure increases down into the ice. Such high pressures will speed up the flow of the ice. So, we can summarize the ice-sheet equation as saying "ice will flow downhill, especially where the ice is very thick."

2.2.2 Numerical Solution

We can do a better job by writing $\nabla \cdot (\Gamma H^{n+2} |\nabla H|^{n-1} \nabla H) = \nabla \cdot (D \nabla H)$, where the scalar $D(x,y) = \Gamma H(x,y)^{n+2} |\nabla H(x,y)|^{n-1}$ (Hindmarsh and Payne, 1996). We then use the approximation

$$\begin{aligned} \frac{\partial (D(\partial H/\partial x))}{\partial x} &\approx D(x + \tfrac{1}{2}\Delta x, y)(H(x+\Delta x, y) - H(x,y)) - D(x - \tfrac{1}{2}\Delta x, y)(H(x,y) - H(x - \Delta x, y)) \\ &= D_{i+1/2,j}(H_{i+1,j} - H_{ij}) - D_{i-1/2,j}(H_{ij} - H_{i-1,j}), \end{aligned}$$

and a similar approximation for $\partial(D(\partial H/\partial y))/\partial y$ to write a full approximation for $\nabla \cdot (D \nabla H)$. The terms $D_{i+1/2,j}$, $D_{i-1/2,j}$, $D_{i,j+1/2}$, $D_{i,j-1/2}$ exist at points on a staggered grid, which means we must interpolate values from the main grid to find the staggered values, which we will do below. First, we write our full approximation of the ice sheet equation as

$$\begin{aligned} \frac{H_{ij,k+1} - H_{ijk}}{\Delta t} = M + \big[& D_{i+1/2,j}(H_{i+1,j} - H_{ij}) - D_{i-1/2,j}(H_{ij} - H_{i-1,j}) + \\ & D_{i,j+1/2}(H_{i,j+1} - H_{ij}) - D_{i,j-1/2}(H_{ij} - H_{i,j-1}) \big]. \end{aligned}$$

Solving for $H_{ij,k+1}$ yields

$$\begin{aligned} H_{ij,k+1} = H_{ijk} + \Delta t \big(M + \big[& D_{i+1/2,j}(H_{i+1,j} - H_{ij}) - D_{i-1/2,j}(H_{ij} - H_{i-1,j}) \\ & + D_{i,j+1/2}(H_{i,j+1} - H_{ij}) - D_{i,j-1/2}(H_{ij} - H_{i,j-1}) \big] \big). \end{aligned} \tag{5}$$

Before we can use (5) to solve anything, it is necessary to evaluate the values of D on the staggered grid. For example, consider

$$D_{i+1/2,j} = \Gamma H_{i+1/2,j}^{n+2} \left| \nabla H_{i+1/2,j} \right|^{n-1}.$$

We must rewrite $D_{i+1/2,j}$ in terms of values of H at points on the main grid. It is straightforward to use

$$H_{i+1/2,j}^{n+2} \approx \left(\frac{H_{ij} + H_{i+1,j}}{2} \right)^{n+2}, \tag{6}$$

but the $|\nabla H_{i+1/2,j}|^{n-1}$ term is a little bit more complicated. Writing it as

$$|\nabla H_{i+1/2,j}|^{n-1} = \left[\left(\frac{\partial H_{i+1/2,j}}{\partial x} \right)^2 + \left(\frac{\partial H_{i+1/2,j}}{\partial y} \right)^2 \right]^{\frac{1}{2}(n-1)}, \tag{7}$$

we can see that $\partial H_{i+1/2,j}/\partial x$ can be approximated using the forward difference (in fact, the forward difference is closer to $\partial H_{i+1/2,j}/\partial x$ than $\partial H_{ij}/\partial x$). The partial derivative with respect to y can be found using a sort of central difference:

$$\begin{aligned}\frac{\partial H_{i+1/2,j}}{\partial y} &\approx \frac{\frac{1}{2}(H_{i,j+1}+H_{i+1,j+1})-\frac{1}{2}(H_{i,j-1}+H_{i+1,j-1})}{2\Delta y} \\ &= \frac{H_{i,j+1}+H_{i+1,j+1}-H_{i,j-1}-H_{i+1,j-1}}{4\Delta y}. \end{aligned} \tag{8}$$

Substituting the forward difference in the x direction and (8) into equation (7) we get

$$|\nabla H_{i+1/2,j}|^{n-1} = \left[\left(\frac{H_{i+1,j}-H_{ij}}{\Delta x}\right)^2 + \left(\frac{H_{i,j+1}+H_{i+1,j+1}-H_{i,j-1}-H_{i+1,j-1}}{4\Delta y}\right)^2\right]^{\frac{1}{2}(n-1)}. \tag{9}$$

Substituting (9) along with (6) we can evaluate D at this staggered point with the formula

$$\begin{aligned} D_{i+1/2,j} &= \Gamma\left(\frac{H_{ij}+H_{i+1,j}}{2}\right)^{n+2} \\ &\quad \cdot\left[\left(\frac{H_{i+1,j}-H_{ij}}{\Delta x}\right)^2 + \left(\frac{H_{i,j+1}+H_{i+1,j+1}-H_{i,j-1}-H_{i+1,j-1}}{4\Delta y}\right)^2\right]^{\frac{1}{2}(n-1)}. \end{aligned} \tag{10}$$

Formulas for $D_{i-1/2,j}$, $D_{i,j+1/2}$, $D_{i,j-1/2}$ can be found in a similar way. Substituting these formulas into (5) gives us a full numerical approximation of the ice sheet equation.

2.2.3 Stability Requirements

As with the heat equation, the time steps must meet a certain condition for the numerical solution of the ice-sheet equation to be stable. We require that

$$\frac{\Delta t}{(\Delta x)^2} \le \frac{1}{4\max D},$$

using $\Delta x = \Delta y$ as with the heat equation (Bueler et al., 2005). The reason for this requirement will not be discussed here.

What's notable about this requirement is that it changes from time step to time step. When $\max D$ is big (when the ice is "rough" and has steep and high bumps), Δt will have to be very small relative to $(\Delta x)^2$. But once the ice has smoothed out and $\max D$ becomes small, the restriction becomes more lax and we can use larger time steps. This means we can improve performance by varying the time step as our model runs. When ice is bumpy we have to take it slow and use tiny time steps, but when it's smooth we can save lots of time by using big time steps. Even in solving the fully thermocoupled ice-sheet equation, PISM uses a variable time stepping scheme to improve performance.

3 Visualization of PISM Output in IDV

Thus far, PISM users have been using software like `ncview` to display their data. This section provides instructions for PISM users on how to use IDV to produce fancier visualizations. A comparison between `ncview` and IDV visualizations can be seen in Figure 5.

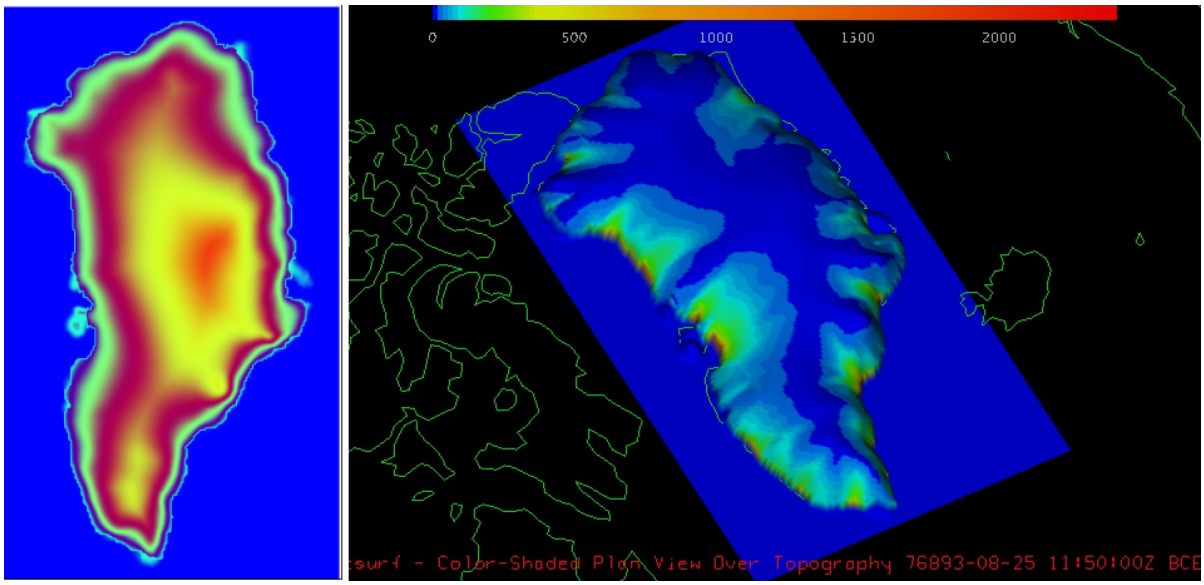

Figure 5: On the left is an `ncview` visualization of ice thickness (red is thicker, blue is thinner). On the right is a 3D plot of the same ice thickness colored by the surface ice speed from IDV.

3.1 Preparing Output for a Still Image

Before using IDV to view a NetCDF file `foo.nc` output by PISM, the x and y coordinates must be transposed (otherwise the visualization will appear rotated and flipped). This can be done automatically by the `transpose.sh` script, listed in the appendix:

```
$ transpose.sh foo.nc foo_transposed.nc
```

where `foo_transposed.nc` is the new file containing the same data as `foo.nc` with transposed x and y coordinates.

3.2 Preparing Output for an Animation

Creating an animation in IDV requires a file with data for multiple times. This can be done by using the `ncrcat` command from NCO. For example, if there are three output files `foo_1.nc`, `foo_2.nc`, and `foo_3.nc`, each containing data for a separate time, they can be combined into a single file `foo_all.nc` using the command:

```
$ ncrcat foo_1.nc foo_2.nc foo_3.nc foo_all.nc
```

Remember that the `transpose.sh` command above must be performed on `foo_all.nc` before it can properly be viewed in IDV:

```
$ transpose.sh foo_all.nc foo_all_transposed.nc
```

3.3 Loading PISM Data into IDV

Once we have an output file `foo.nc` ready for use in IDV, we start up IDV. Two windows will appear, one called the `Dashboard` and the other called the `Map View`. The `Map View` should, by default, show the coastline of North America. Select the `Dashboard` window and click the `Data Choosers` tab. In the pane to the left, there should be several tabs called `Files`, `URLs`, `Catalogs`, etc. It should be selected by default, but if it is not, click the `Files` tab. Navigate to the directory containing `foo.nc`, select it, and click the `Add Source` button. Now that the data is loaded, IDV should automatically switch to the `Field Selector` tab.

3.4 2D Plot

Producing a 2D graph is straightforward. Under the `Field Selector` tab in the `Dashboard`, select some 2D data, such as `2D grid` → `ice upper surface elevation`. In the `Displays` *pane* under the `Field Selector` tab (not the `Displays` *tab*), select `Plan Views` → `Color-Shaded Plan View`. Click the `Create Display` button. Now the `Map View` window should display a color plot somewhat like the plots produced by `ncview`, except that the coastlines of Greenland and other nearby land should also appear. The color scale for the plot can be changed under the `Displays` *tab* in the `Dashboard` window by clicking the button next to the `Color Table:` label.

3.5 Changing the View

To zoom in and out, hold `Shift` and right click and drag up and down the view, or use the scroll wheel. It's also possible to zoom in to a specific area by holding `Shift` and left clicking and dragging a white box to zoom into. To move the view horizontally, hold `Control` and right click and drag the view as desired. Simply right click and drag to change the view in 3D. Press `Control`+`R` to reset to the top-down view that first appeared.

3.6 3D Isosurface

Under the `Field Selector` tab in the `Dashboard`, select `3D grid` → `ice temperature`. (It should be possible to use `age of ice` also, but something about that data seems to make IDV run out of memory while calculating an isosurface (even with 2 GB available, which appears to be the most Java can handle).) In the `Displays` *pane* under the `Field Selector` tab, select `3D Surface` → `Isosurface`. Click the `Create Display` button. The isosurface will appear in the `Map View` window. The color and value for the isosurface can be changed under the `Displays` *tab* in the `Dashboard` window.

3.7 3D Plot

Creating 3D plots requires tricking IDV into thinking that the data you want to plot is topographic data. The easiest way to pull this off is using "formulas." Click on `Formulas` in the `Data Sources` sidepane under the `Field Selector` in the `Dashboard` window, and select `Miscellaneous` → `Any Field`. Select `3D Surface` → `Topography`, and click `Create Display`. A dialog should appear. Select *file name* → `2D grid` → `ice upper surface elevation`, where *file name* is the location of the NetCDF file you loaded, (any data under `2D grid` should work) and click `OK`. A 3D surface will appear in the `Map View` window (the fact it's 3D will be more apparent if you right click and drag in the view, as described in Subsection 3.5). As with 2D plots, the color scale can be changed under the `Displays` tab under the `Dashboard` window.

A 3D plot can also be colored by another variable; for example, the ice elevation surface can be colored by the surface ice speed. Delete the existing 3D plot by clicking the trash can icon in the `parameterWrapper - Topography` entry in the `Legend` in the `Map View` window (all existing displays can be quickly deleted by clicking the eraser icon in the top toolbar in both windows). Return to the `Field Selector` tab in the `Dashboard` window and click *file name* in the `Data Sources` sidepane. In the `Fields` pane, select `2D grid` → `magnitude of horizontal velocity ice at ice surface`. Select `Color-Shaded Plan View Over Topography` and click `Create Display`. A dialog will appear where you should select `Formulas` → `Miscellaneous` → `Any Field` and click `OK`. Another dialog will be displayed. Select *file name* → `2D grid` → `ice upper surface elevation`, (again, any 2D grid data should work) and click `OK`.

The `default` color scale for data like surface ice velocity will probably be unsatisfactory (almost all the ice will be roughly the same shade of blue). As usual, you can change the color scale under the `Displays` tab in the `Dashboard` window. However, I did not find an alternative color scale that really worked well (a logarithmic scale would be ideal). Unfortunately, it seems the only reasonable way to address this is to create a custom color scale by clicking the `default` button and then `Edit Color Table`.

3.8 Animation

Once you have a NetCDF file with data for multiple times, creating an animation in IDV is fairly simple. Just above the view in the `Map View` window there is a row of buttons that looks similar to buttons on a media-player. To the left of these buttons is a drop-down menu that should contain a time (in the format *year*-*month*-*date* *hour*:*minute*:*second*`Z`. Above this menu should be a row of green rectangles with one blue rectangle (if you have data for lots of points in time, it will look more like a long green bar with a thin vertical blue line). Each green rectangle represents a point in time. Clicking on one will display the data for that time, and they can also be navigated using the media-player-like buttons or the drop-down menu. You can "play" the animation by pressing the "play" button. If you want to change how quickly the animation plays, click on the circular "i" button to the right of the media-player-like buttons; it will bring up a dialog with which you can modify the delay between frames.

To save a movie, click on `View` → `Capture` → `Movie`, and a dialog will pop up. You can click the `Time Animation` button to begin saving the frames of the movie. *Make sure the view is not covered with anything else on the screen while recording.* The movie will contain whatever was on the screen in the location of the view, whether it's the view or not. You can then click `Save Movie` and choose what format (and frame rate) to use.

4 Conclusion

Ice sheet modelers face difficult challenges on many levels. This paper is intended to give an introduction to just one such challenge. Although the concept of approximating derivatives using finite differences would seem simple enough, the pitfall of unstable solutions is much less intuitive. Indeed, Morton and Mayers, among others, have written an entire book about these methods. Having worked through the process of numerically solving two relatively simple PDEs (in comparison to the fully thermocoupled case that PISM is tackling), I have gained a feel for how these kinds of models behave and how errors can cause problems.

This paper also documents the methods I have used to create visualizations of PISM data. It is hoped that this documentation will useful to PISM users who would like to visualize their own PISM data in the same way. Although my visualizations were focused on Greenland, it will be exciting to see the results of very recent improvements in modeling basal sliding in Antarctica, and perhaps such visualizations will aid in that research.

5 Acknowledgments

I am very grateful to my mentor, Ed Bueler, for help throughout the summer, patience as I learned the ropes of PDEs, funding through his grant for a glaciology course I took at UAF, and many interesting discussions. Patrick Webb also has my thanks for both teaching a seminar on IDV and for helping me individually with producing 3D plots. I also thank Greg Newby for all the work he did to make my time in Fairbanks (and that of the other interns) so enjoyable, as well as helping to let me enjoy both Denali National Park and the glaciology course while I was in Alaska.

Appendix: Exact Solutions of PDEs

An analytical solution to a heat equation boundary value problem can be found using Fourier series. Let us use the above example: a square plate with one corner at the origin and the opposite corner at $(1, 1)$, use the boundary condition of temperature 0 at the sides: $u(x, y, t) = 0$ for $x = 0$, $x = 1$, $y = 0$, or $y = 1$, and any t. The initial condition is

$$u(x, y, 0) = \delta(x - \tfrac{1}{2})\delta(y - \tfrac{1}{2}).$$

If we let $u(x, y, t) = X(x)Y(y)T(t)$, substitution into (1) gives

$$X(x)Y(y)T'(t) = X''(x)Y(y)T(t) + X(x)Y''(y)T(t).$$

```
#!/bin/sh
if [ $# -ne 2 ]
then
echo "usage:  $0 input_file.nc output_file.nc"
exit
fi
script_path=$(dirname "$0")
ncdump $1 > $1.asc
$script_path/transpose.py $1.asc
ncgen -o $2 $1.asc
rm $1.asc
```

Figure 6: `transpose.sh`

Separation of variables yields

$$\frac{T'(t)}{T(t)} = \frac{X''(x)}{X(x)} + \frac{Y''(y)}{Y(y)}, \tag{11}$$

and since the left-hand side of (11) is independent of x and y while the right-hand side is independent of t. Therefore, we know both sides are equal to some constant $-2(\beta^2 + \gamma^2)$, so

$$T(t) = ae^{-2(\beta^2+\gamma^2)t}$$

for some constant a. We also know that X''/X and Y''/Y are independent of each other and therefore constant; given the boundary conditions and symmetry in the x and y directions we can write

$$X(x) = \sin(\beta x), \quad Y(y) = \sin(\gamma y), \quad \beta = m\pi,\ \gamma = n\pi,$$

for positive integers m and n (any coefficients can be absorbed into a). Since the sum of solutions to (1) is a solution to (1), we find that

$$u(x,y,t) = \sum_{m=1}^{\infty}\sum_{n=1}^{\infty} a_{mn} e^{-2((m^2+n^2)\pi)^2 t} \sin(m\pi x)\sin(n\pi y),$$

where the coefficients a_{mn} are given by

$$\begin{aligned} a_{mn} &= 4\int_0^1\int_0^1 u(x,y,0)\sin(m\pi x)\sin(n\pi y)\,dx\,dy \\ &= 4\int_0^1\int_0^1 \delta(x-\tfrac{1}{2})\delta(y-\tfrac{1}{2})\sin(m\pi x)\sin(n\pi y)\,dx\,dy \\ &= 4\sin\left(\frac{m_1\pi}{2}\right)\sin\left(\frac{m_2\pi}{2}\right). \end{aligned}$$

Appendix: transpose.sh

The `transpose.sh` script works together with a Python script called `transpose.py`. All they do is swap the `x` and `y` variables in a NetCDF file (it should work for most any NetCDF file, but they are not really intended for use with non-PISM files).

Below are the contents of both files. They should be placed together in the same directory when used.

References

Alley, Richard B. *The Two-Mile Time Machine*. Princeton University Press, 2000.

```
#!  /usr/bin/env python
import sys
# Get the file's text.
oldfile = open(sys.argv[1], 'r')
text = oldfile.read()
oldfile.close()
index = 0 # index helps keep track of where we are in the file.
# Swap x and y in the dimensions section.
index = text.find('x = ') + 4
text = text.replace('x = ', 'y = ', 1)
text = text[:index] + text[index:].replace('y = ','x = ', 1)
# Swap x and y in the variables section.
text = text.replace('x, y', 'y, x')
# Swap x and y in the data section.
index = text.find('data:')
index = text.find('x =', index)
text = text[:index] + text[index:].replace('x =', 'y =', 1)
index += 3
text = text[:index] + text[index:].replace('y =', 'x =', 1)
newfile = open(sys.argv[1], 'w')
newfile.write(text)
newfile.close()
```

Figure 7: `transpose.py`

Bueler, Ed, Craig S. Lingle, Jed A. Kallen-Brown, David N. Covey, and Latrice N. Bowman. "Exact solutions and verification of numerical models for isothermal ice sheets." *Journal of Glaciology* 51, 173: (2005) 291–306.

Hindmarsh, Richard C. A., and Antony J. Payne. "Time-step limits for stable solutions of the ice-sheet equation." *Annals of Glaciology* 23: (1996) 74–85.

Morton, K. W., and D. F. Mayers. *Numerical Solution of Partial Differential Equations.* Cambridge University Press, 2005.