Numerical modelling of ice sheets, streams, and shelves

Ed Bueler

Dept of Mathematics and Statistics
and Geophysical Institute
University of Alaska, Fairbanks

September 2014
Karthaus Summer School

what are *you* trying to do with this thing?

slogans & scope

slogans: I will

- focus on approximating ice flow
- provide numerical codes that actually work
- always care about the continuum model
- provide multiple entry points

scope: I will cover these

- models
  - shallow ice approximation (SIA) in 2D
  - shallow shelf approximation (SSA) in 1D
  - mass continuity & surface kinematical equations
- numerical ideas
  - finite difference schemes
  - solving algebraic systems from stress balances
  - verification

notation

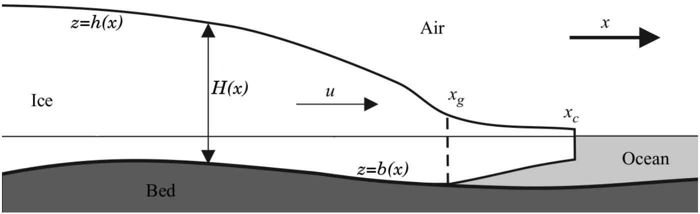


figure modified from Schoof (2007)

- coordinates t, x, y, z (with z vertical, positive upward)
- subscripts for partial derivatives $u_x = \partial u/\partial x$
- $H =$ ice thickness
- $h =$ ice surface elevation
- $b =$ bedrock surface elevation
- $T =$ temperature
- $\mathbf{u} = (u, v, w) =$ ice velocity
- $\rho =$ density of ice
- $g =$ acceleration of gravity
- $A = A(T) =$ ice softness in Glen law ($\mathbf{D}_{ij} = A(T)\tau^{n-1}\tau_{ij}$)
- see printed notes for notation
- please ask about notation! (stupid questions impossible)

Matlab/Octave codes

- slides and notes are structured around 18 Matlab/Octave codes
- ...though only 5 appear in these slides
- each about 1/2 page of code
- with rich comments and help files
- please give them a try!
    - `.zip` and `.tar.gz` forms available from memory stick
    - and online:

`https://github.com/bueler/karthaus`

Outline

ice in glaciers is a *fluid*

- what's a fluid?

ice in glaciers is a *fluid*

- what's a fluid?

- at minimum, we describe fluids by
    - a *density field* $\rho(t, x, y, z)$
    - a vector *velocity field* $\mathbf{u}(t, x, y, z)$

- we will assume ice is constant density
  - so ice flow is incompressible
- if ice fluid were also
  - faster-moving than it actually is, and
  - linearly-viscous

  then ice flow would be a "typical" fluid like liquid water
- ... and one would use the Navier-Stokes equations:

$$
\begin{aligned}
\nabla \cdot \mathbf{u} &= 0 && \textit{incompressibility} \\
\rho \left(\mathbf{u}_t + \mathbf{u} \cdot \nabla \mathbf{u} \right) &= -\nabla p + \nabla \cdot \tau_{ij} + \rho \mathbf{g} && \textit{stress balance} \\
2\nu \mathbf{D}_{ij} &= \tau_{ij} && \textit{linear flow law}
\end{aligned}
$$

- this is the main continuum model for fluid dynamics
- stress balance equation is "$ma = F$"

hmmm . . . does not sound like glaciology to me!

- yes, numerical ice sheet flow modelling is "computational fluid dynamics"
  - it's large-scale like atmosphere and ocean
  - . . . but it is a weird one
- consider what makes ocean flow modeling exciting:
  - coriolis force
  - density/salinity stratification
  - convection
  - turbulence
- . . . none of these are relevant to ice flow
- so what could be interesting about the flow of slow, cold, laminar, old ice?
  - it's *ice dynamics!*

ice is a slow, shear-thinning fluid

- but *our* fluid is
  "*slow*": $\rho\,(\mathbf{u}_t + \mathbf{u}\cdot\nabla\mathbf{u}) \approx 0$
  non-Newtonian: viscosity ν is not constant
- slow:

$$\rho\,(\mathbf{u}_t + \mathbf{u}\cdot\nabla\mathbf{u}) \approx 0 \quad\iff\quad \begin{pmatrix}\text{forces of inertia}\\ \text{are neglected}\end{pmatrix}$$

- non-Newtonian in a "shear-thinning" way
  - higher strain rates means lower viscosity
- so the standard "full" model is power-law ($n = 3$) Stokes:

$$\begin{aligned} \nabla\cdot\mathbf{u} &= 0 && \textit{incompressibility}\\ 0 &= -\nabla p + \nabla\cdot\tau_{ij} + \rho\,\mathbf{g} && \textit{stress balance}\\ \mathbf{D}_{ij} &= A\tau^2\tau_{ij} && \textit{Glen flow law} \end{aligned}$$

“slow” means no memory of velocity (i.e. momentum)

- a time-stepping ice sheet code . . .
  - recomputes the full velocity field at every time step, and
  - does not require velocity from the previous step
- because the model needs no memory of previous velocity, *velocity is a “diagnostic” output of ice flow models*
- with BIG apologies to Dylan: *to be a weatherman you’ve got to know which way the wind blows . . . but don’t expect that from a glaciologist*

plane flow Stokes

- recall the power-law Stokes model:

$$\begin{aligned} \nabla \cdot \mathbf{u} &= 0 && \textit{incompressibility} \\ 0 &= -\nabla p + \nabla \cdot \tau_{ij} + \rho \, \mathbf{g} && \textit{stress balance} \\ \mathbf{D}_{ij} &= A\tau^2 \tau_{ij} && \textit{Glen flow law} \end{aligned}$$

- now work in the x, z plane
  - like the centerline of a glacier
  - or in a cross-flow plane
- notation on next slide:
  - subscripts "x" and "z" are partial derivatives
  - τ_{13} is the "vertical" shear stress
  - τ_{11} and $\tau_{33} = -\tau_{11}$ are (deviatoric) longitudinal stresses

plane flow Stokes 2

- in the x, z plane flow case the Stokes equations say

$$\begin{aligned} u_x + w_z &= 0 && \textit{incompressibility} \\ p_x &= \tau_{11,x} + \tau_{13,z} && \textit{stress balance } (x) \\ p_z &= \tau_{13,x} - \tau_{11,z} - \rho g && \textit{stress balance } (z) \\ u_x &= A\tau^2\tau_{11} && \textit{flow law } \text{(diagonal)} \\ u_z + w_x &= 2A\tau^2\tau_{13} && \textit{flow law } \text{(off-diagonal)} \end{aligned}$$

- we have five equations in five unknowns $(u, w, p, \tau_{11}, \tau_{13})$
- complicated enough . . . what about in a simplified situation?

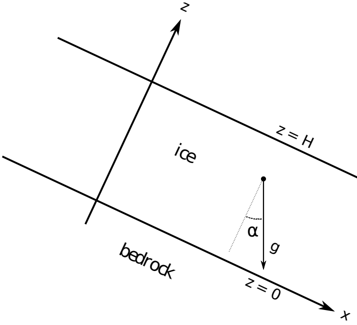
slab-on-a-slope

- suppose we have
  - constant thickness and
  - tilted bed
- rotated coordinates give:

$$\mathbf{g} = g \sin\theta\,\hat{x} - g\cos\theta\,\hat{z}$$

- so p_x, p_z equations are now:

$$p_x = \tau_{11,x} + \tau_{13,z} + \rho g \sin\theta$$
$$p_z = \tau_{13,x} - \tau_{11,z} - \rho g \cos\theta$$



- for an infinite slab-on-a-slope there is *no variation in x*
- the equations simplify:

$$w_z \overset{1}{=} 0 \qquad 0 = \tau_{11}$$
$$p_z \overset{2}{=} -\rho g \cos\theta \qquad u_z = 2A\tau^2\tau_{13}$$
$$\tau_{13,z} \overset{3}{=} -\rho g \sin\theta$$

slab-on-a-slope 2

- add some boundary conditions:

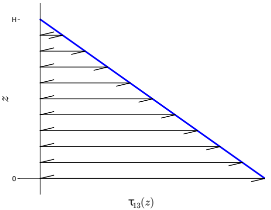
$$u(\text{base}) = u_0, \qquad w(\text{base}) = 0, \qquad p(\text{surface}) = 0$$

- by integrating 1,2,3 vertically, get (see figure):

$$w = 0, \qquad p = \rho g \cos\theta (H - z), \qquad \tau_{13} = \rho g \sin\theta (H - z)$$

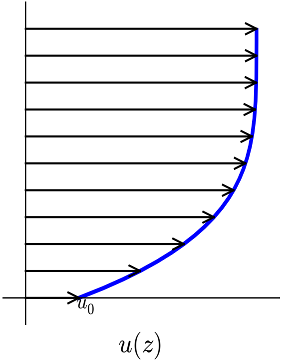
- and from "$u_z = 2A\tau^2\tau_{13}$" get velocity formula:

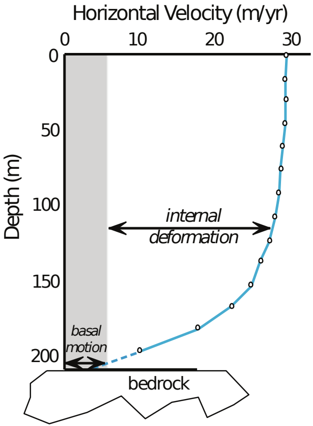
$$\begin{aligned} u(z) &= u_0 + 2A(\rho g \sin\theta)^3 \int_0^z (H - z')^3 \, dz' \\ &= u_0 + \frac{1}{2} A (\rho g \sin\theta)^3 \left(H^4 - (H - z)^4\right) \end{aligned}$$



slab-on-a-slope 3

- do we believe these equations?
- velocity formula on last slide gives figure below
- compare to observations at right





Velocity profile of the Athabasca Glacier, Canada, derived from inclinometry (Savage and Paterson, 1963)

mass continuity

- now we know the velocity u . . . so what?
- suppose, instead of slab-on-a-slope, that our ice flow has variable thickness $H(t, x)$
- compute the vertical average of velocity:

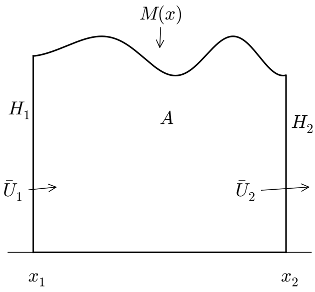
$$\bar{U}(t, x) = \frac{1}{H} \int_0^H u(t, x, z)\, dz$$

mass continuity 2

- $M(x)$ is climatic mass balance
- consider change of area A in figure:

$$\frac{dA}{dt} \stackrel{*}{=} \int_{x_1}^{x_2} M(x)\,dx + \bar{U}_1 H_1 - \bar{U}_2 H_2$$

- assume width $\Delta x = x_2 - x_1$ is small so $A \approx \Delta x\, H$



- divide eqn $*$ by Δx and get

$$H_t = M - \left(\bar{U} H\right)_x$$

- this is a *mass continuity equation*
- "area" in 2D becomes "volume" in 3D

combine equations so far . . .

- from slab-on-slope velocity formula in $u_0 = 0$ case,

$$\begin{aligned}\bar{U}H &= \int_0^H \frac{1}{2} A(\rho g \sin\theta)^3 \left(H^4 - (H-z)^4\right)\,dz \\ &= \frac{2}{5} A(\rho g \sin\theta)^3 H^5\end{aligned}$$

- note $\sin\theta \approx \tan\theta = -h_x$
- combine with mass continuity $H_t = M - \left(\bar{U}H\right)_x$ to get:

$$H_t = M + \left(\frac{2}{5}(\rho g)^5 A H^5 |h_x|^2 h_x\right)_x$$

- this is rough explanation of "shallow ice approximation"
- . . . which will be our first numerically-solved ice flow model!

Outline

slow, non-Newtonian, shallow, and sliding

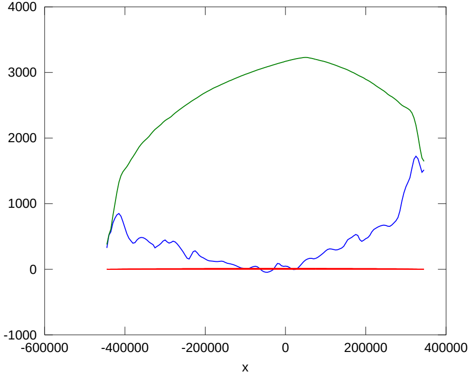
- ice sheets have four outstanding properties *as fluids*:
  1. slow
  2. non-Newtonian
  3. shallow (at big scale)
  4. contact slip (some places, sometimes)

slow, non-Newtonian, shallow, and sliding

- ice sheets have four outstanding properties *as fluids*:
  1. slow
  2. non-Newtonian
  3. shallow (at big scale)
- our first model captures the first three properties

regarding "shallow"

- below in red is a no-vertical-exaggeration cross section of Greenland at 71°
- green and blue: standard vertically-exaggerated cross section



- *note*: you can scale Stokes equation using smallness of $\epsilon = [H]/[L]$, where $[H]$ is a typical thickness of an ice sheet and $[L]$ is a typical horizontal dimension, ... (see Chapter 18 of Fowler, 1997)

flow model I: non-sliding, isothermal shallow ice approximation (= SIA)

a model which applies to

- small depth-to-width ratio ("shallow") grounded ice sheets
- on not-too-rough bed topography,
- whose flow is not dominated by sliding and/or liquid water at the base or margin

"Polaris Glacier," northwest Greenland, photo 122, Post & LaChapelle (2000)

SIA velocity equation

- instead of an $\epsilon = [H]/[L]$ argument, here we take the simple slogan:

  the SIA uses the formulas from slab-on-a-slope

- shear stress approximation:

$$(\tau_{13}, \tau_{23}) = -\rho g(h - z)\nabla h$$

- let $\mathbf{U} = (u, v)$ be the horizontal velocity
- we further approximate

$$\begin{aligned}\mathbf{U}_z &= 2A|(\tau_{13}, \tau_{23})|^{n-1}(\tau_{13}, \tau_{23})\\ &= -2A(\rho g)^n(h - z)^n|\nabla h|^{n-1}\nabla h\end{aligned}$$

- by integrating vertically, in the non-sliding case,

$$\mathbf{U} = -\frac{2A(\rho g)^n}{n+1}\left[H^{n+1} - (h - z)^{n+1}\right]|\nabla h|^{n-1}\nabla h$$

SIA thickness equation

- but mass continuity remains:

$$H_t = M - \nabla \cdot (\bar{\mathbf{U}} H)$$

- from velocity formula for **U** on last slide, combine to get the non-sliding, isothermal shallow ice approximation:

$$H_t = M + \nabla \cdot \left(\Gamma H^{n+2} |\nabla h|^{n-1} \nabla h\right)$$

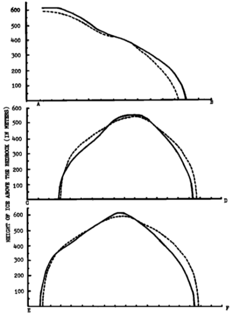
  - where H is ice thickness, h is ice surface elevation, b is bed elevation ($h = H + b$)
  - M combines surface and basal mass (im)balance: accumulation if $M > 0$, ablation if $M < 0$
  - n is the exponent in the Glen flow law
  - $\Gamma = 2A(\rho g)^n/(n+2)$ is a positive constant

the SIA model

- the SIA equation from last slide:

$$\boxed{H_t = M + \nabla \cdot \left(\Gamma H^{n+2} |\nabla h|^{n-1} \nabla h\right)} \qquad (1)$$

- equation (1) describes how ice thickness varies in time and space, as a function of climate M and bed elevation b
- numerically solve (1) and you've got a usable model for . . . *the Barnes ice cap* (Mahaffy, 1976; model-observation fit below)



good questions:

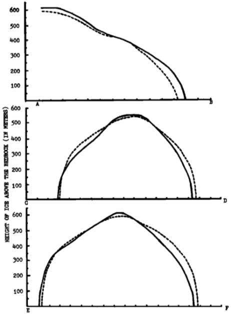
1. where does equation (1) come from?
2. how to solve it numerically?
3. how to *think* about it?

the SIA model

- the SIA equation from last slide:

$$\boxed{H_t = M + \nabla \cdot \left(\Gamma H^{n+2} |\nabla h|^{n-1} \nabla h\right)} \qquad (1)$$

- equation (1) describes how ice thickness varies in time and space, as a function of climate M and bed elevation b
- numerically solve (1) and you've got a usable model for . . . *the Barnes ice cap* (Mahaffy, 1976; model-observation fit below)



good questions:

1. where does equation (1) come from? ✓
2. how to solve it numerically?
3. how to *think* about it?

heat equation (as analogy for SIA)

- recall Newton's law of cooling

$$\frac{dT}{dt} = -K(T - T_{\text{ambient}})$$

where

  - where T is object temperature and
  - K relates to material and geometry of object

- e.g. hot cup of coffee in lecture room

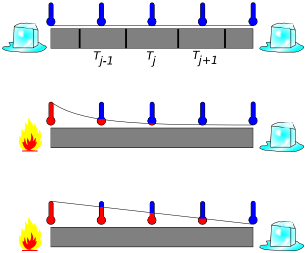
- solution:

  exponential decay of $T(t)$ to T_{ambient}

heat equation (as analogy for SIA)

- Newton's law for segments of a rod:

$$\frac{dT_j}{dt} = -K\left(T_j - \frac{1}{2}(T_{j-1} + T_{j+1})\right) = \frac{K}{2}\left(T_{j-1} - 2T_j + T_{j+1}\right)$$



- ...because average of neighbor's temperatures *is* ambient
- has this limit as segments shrink:

$$T_t = DT_{xx}$$

  where
  - D is "diffusivity"
  - D includes material properties: conductivity, density, heat capacity

- see also: "finite difference approximations"

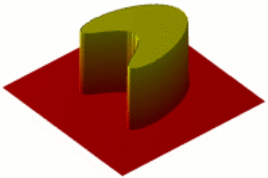
2D heat equation

- $T(t, x, y)$ is temperature in object at position x, y and time t
- Fourier rewrote Newton's law as a rule for heat flux: $\mathbf{q} = -k\nabla T$
- also: ρ is density, c is specific heat, k is conductivity, f is heat source
- by conservation of energy:

$$\rho c T_t = f + \nabla \cdot (k \nabla T)$$

- define $D = k/(\rho c)$ and $F = f/(\rho c)$
- get heat equation:

$$T_t = F + \nabla \cdot (D \nabla T) \qquad (2)$$



analogy: SIA versus 2D heat equation

- side-by-side comparison of (1) and (2):

SIA: $H(t,x,y)$ is ice thickness

$$\boxed{H_t = M + \nabla \cdot \left(\Gamma H^{n+2} |\nabla h|^{n-1}\, \nabla h\right)}$$

heat: $T(t,x,y)$ is temperature

$$\boxed{T_t = F + \nabla \cdot (D \nabla T)}$$

- we identify the diffusivity in the SIA:

$$D = \Gamma H^{n+2} |\nabla h|^{n-1}$$

- *non-sliding shallow ice flow diffuses the ice sheet*
- some issues with this analogy:
  - H and h are different ... but at least $H = h + b$
  - D depends on H, the solution
  - $D \to 0$ at margin, where $H \to 0$
  - $D \to 0$ at divides/domes, where $|\nabla h| \to 0$

analogy: SIA versus 2D heat equation

- side-by-side comparison of (1) and (2):

SIA: $H(t,x,y)$ is ice thickness	heat: $T(t,x,y)$ is temperature
$\boxed{H_t = M + \nabla\cdot\left(\Gamma H^{n+2}\lvert\nabla h\rvert^{n-1}\,\nabla h\right)}$	$\boxed{T_t = F + \nabla\cdot(D\,\nabla T)}$

- we identify the diffusivity in the SIA:

$$D = \Gamma H^{n+2}\lvert\nabla h\rvert^{n-1}$$

- *non-sliding shallow ice flow diffuses the ice sheet*
- some issues with this analogy:
  - H and h are different …but at least $H = h + b$
  - D depends on H, the solution
  - $D \to 0$ at margin, where $H \to 0$
  - $D \to 0$ at divides/domes, where $\lvert\nabla h\rvert \to 0$
- *we are finally ready for numerics!*

numerics for heat equation: basic ideas of finite differences

- numerical schemes for heat equation are good start for SIA
- for differentiable $f(x)$ and any h, *Taylor's theorem* says

$$f(x+h) = f(x) + f'(x)h + \frac{1}{2}f''(x)h^2 + \frac{1}{3!}f'''(x)h^3 + \dots$$

- you can replace "h" by multiples of Δx, e.g.:

$$f(x-\Delta x) = f(x) - f'(x)\Delta x + \frac{1}{2}f''(x)\Delta x^2 - \frac{1}{3!}f'''(x)\Delta x^3 + \dots$$

$$f(x+2\Delta x) = f(x) + 2f'(x)\Delta x + 2f''(x)\Delta x^2 + \frac{4}{3}f'''(x)\Delta x^3 + \dots$$

- *combine expressions like these to give approximations of derivatives, from values on a grid*

finite differences for partial derivatives

- we want partial derivative expressions, for example with any function $u = u(t,x)$:

$$u_t(t,x) = \frac{u(t+\Delta t,x) - u(t,x)}{\Delta t} + O(\Delta t),$$
$$u_t(t,x) = \frac{u(t+\Delta t,x) - u(t-\Delta t,x)}{2\Delta t} + O(\Delta t^2),$$
$$u_x(t,x) = \frac{u(t,x+\Delta x) - u(t,x-\Delta x)}{2\Delta x} + O(\Delta x^2),$$
$$u_{xx}(t,x) = \frac{u(t,x+\Delta x) - 2u(t,x) + u(t,x-\Delta x)}{\Delta x^2} + O(\Delta x^2)$$

  and so on
- sometimes we want a derivative in-between grid points:

$$u_x(t,x+(\Delta x/2)) = \frac{u(t,x+\Delta x) - u(t,x)}{\Delta x} + O(\Delta x^2)$$

- "$+O(h^2)$" is better than "$+O(h)$" if h is a small number

explicit scheme for heat equation

- consider 1D heat equation $T_t = DT_{xx}$
- an *explicit* scheme comes from:

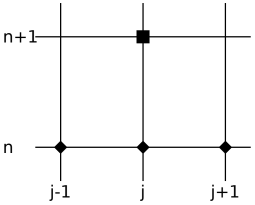
$$\frac{T(t+\Delta t, x) - T(t,x)}{\Delta t} \approx D\,\frac{T(t, x+\Delta x) - 2T(t,x) + T(t, x-\Delta x)}{\Delta x^2}$$

- more notation:
  - (t_n, x_j) denotes a point in a time-space grid
  - $T_j^n \approx T(t_n, x_j)$
  - $\nu = D\Delta t/(\Delta x)^2$

- then scheme is

$$T_j^{n+1} = \nu T_{j+1}^n + (1-2\nu)T_j^n + \nu T_{j-1}^n$$

- stencil $\longrightarrow$

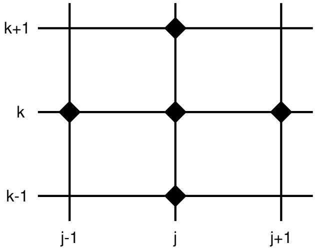


explicit scheme in two space dimensions

- a heat equation in 2D: $T_t = D(T_{xx} + T_{yy})$
- we write $T^n_{jk} \approx T(t_n, x_j, y_k)$
- the 2D explicit scheme is

$$\frac{T^{n+1}_{jk} - T^n_{jk}}{\Delta t} = D\left(\frac{T^n_{j+1,k} - 2T^n_{jk} + T^n_{j-1,k}}{\Delta x^2} + \frac{T^n_{j,k+1} - 2T^n_{jk} + T^n_{j,k-1}}{\Delta y^2}\right)$$

- space-only stencil below



implementation

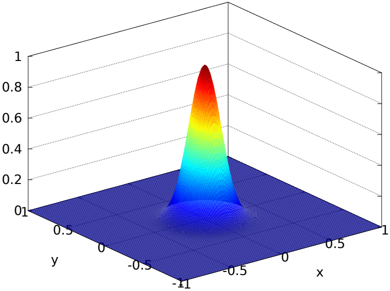
```
function T = heat(D,J,K,dt,N)

dx = 2 / J;     dy = 2 / K;
[x,y] = meshgrid(-1:dx:1, -1:dy:1);
T = exp(-30*(x.*x + y.*y));

mu_x = dt * D / (dx*dx);
mu_y = dt * D / (dy*dy);
for n=1:N
   T(2:J,2:K) = T(2:J,2:K) + ...
       mu_x * ( T(3:J+1,2:K) - 2 * T(2:J,2:K) + T(1:J-1,2:K) ) + ...
       mu_y * ( T(2:J,3:K+1) - 2 * T(2:J,2:K) + T(2:J,1:K-1) );
end

surf(x,y,T),  shading('interp'),  xlabel x,  ylabel y
```

heat.m

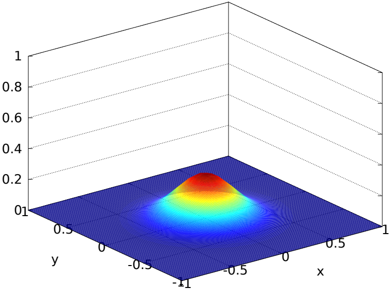
- solves $T_t = D(T_{xx} + T_{yy})$ on square $-1 < x < 1$, $-1 < y < 1$
- example uses gaussian initial condition $T(0,x,y) = e^{-30r^2}$
- uses “colon notation” to remove loops
- » `heat(1.0,30,30,0.001,20)`
  approximates T on 30×30 spatial grid, with $D = 1$; take $N = 20$ steps of $\Delta t = 0.001$ to final time $t = 0.02$

the look of success

initial condition $T(0, x, y)$

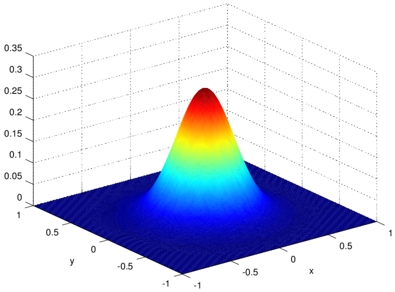


approximate solution $T(t, x, y)$ at $t = 0.02$

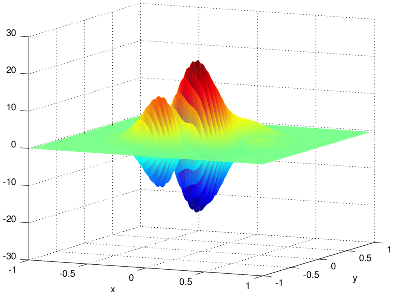


the look of instability

- both figures are from solving $T_t = D(T_{xx} + T_{yy})$ on the *same* space grid and the *same* final time
- ... but with slightly different time steps



$\Delta t = 0.0001$ and $\dfrac{D\Delta t}{\Delta x^2} = 0.16$



$\Delta t = 0.0002$ and $\dfrac{D\Delta t}{\Delta x^2} = 0.32$

understand and avoid the instability

- recall 1D explicit scheme had the form

$$T_j^{n+1} = \nu T_{j+1}^n + (1-2\nu) T_j^n + \nu T_{j-1}^n$$

- thus the new value T_j^{n+1} is an *average* of the old values, *if the middle coefficient is positive*:

$$1 - 2\nu \geq 0 \quad \iff \quad \frac{D\Delta t}{\Delta x^2} \leq \frac{1}{2} \quad \iff \quad \Delta t \leq \frac{\Delta x^2}{2D}$$

- averaging is stable because averaged wiggles are smaller than the original wiggles
- *the result was unstable because the time step was too big*
- in 2D case with $\Delta x = \Delta y$ the condition is

$$\frac{D\Delta t}{\Delta x^2} \leq \frac{1}{4}$$

- this condition is a sufficient stability criterion

adaptive implementation: guaranteed stability

```
function T = heatadapt(D,J,K,tf)

dx = 2 / J;     dy = 2 / K;
[x,y] = ndgrid(-1:dx:1, -1:dy:1);
T = exp(-30*(x.*x + y.*y));

t = 0.0;     count = 0;
while t < tf
   dt0 = 0.25 * min(dx,dy)^2 / D;
   dt = min(dt0, tf - t);
   mu_x = dt * D / (dx*dx);    mu_y = dt * D / (dy*dy);
   T(2:J,2:K) = T(2:J,2:K) + ...
       mu_x * ( T(3:J+1,2:K) - 2 * T(2:J,2:K) + T(1:J-1,2:K) ) + ...
       mu_y * ( T(2:J,3:K+1) - 2 * T(2:J,2:K) + T(2:J,1:K-1) );
   t = t + dt;
   count = count + 1;
end

surf(x,y,T),  shading('interp'),  xlabel x,  ylabel y
```

`heatadapt.m`

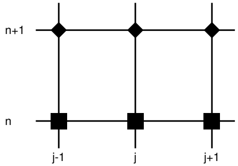
- same as `heat.m` except

time step computed from stability criterion

alternative instability fix: implicitness

- implicit methods can be stable for *any* positive time step Δt

- an implicit scheme is *Crank-Nicolson* $\longrightarrow$
- Crank-Nicolson has smaller error too



- *but* you have to solve a linear (for heat equation) system of equations to take each time step
  - becomes a *nonlinear* system of equations at each time step for ice flow
  - implementation of nonlinear solver is an “opportunity cost”
  - . . . never address the other processes you really care about?

- Donald Knuth has advice for ice sheet modelers:

  We should forget about small efficiencies . . . : premature optimization is the root of all evil.

variable diffusivity and staggered grid

- recall the analogy: (SIA) $\leftrightarrow$ (heat eqn)
- the SIA has a diffusivity which varies in space, so consider a more general heat equation:

$$T_t = F + \nabla \cdot (D(x,y)\nabla T)$$

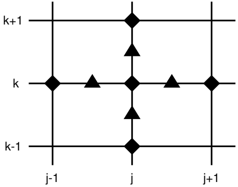
- the explicit method is conditionally stable with the same time step restriction if we evaluate diffusivity $D(x,y)$ at staggered grid points:

$$\nabla \cdot (D(x,y)\nabla T) \approx \frac{D_{j+1/2,k}(T_{j+1,k} - T_{j,k}) - D_{j-1/2,k}(T_{j,k} - T_{j-1,k})}{\Delta x^2} + \frac{D_{j,k+1/2}(T_{j,k+1} - T_{j,k}) - D_{j,k-1/2}(T_{j,k} - T_{j,k-1})}{\Delta y^2}$$

in stencil at right:

diamonds: T

triangles: D



general diffusion equation code

```
function [T,dtav] = diffusion(Lx,Ly,J,K,Dup,Ddown,Dright,Dleft,T0,tf,F,b)

dx = 2 * Lx / J;    dy = 2 * Ly / K;
[x,y] = ndgrid(-Lx:dx:Lx, -Ly:dy:Ly);
T = T0;
if nargin < 11, F = zeros(size(T0)); end
if nargin < 12, b = zeros(size(T0)); end

t = 0.0;    count = 0;
while t < tf
   maxD = [max(max(Dup)) max(max(Ddown)) max(max(Dleft)) max(max(Dright))];
   maxD = max(maxD);
   if maxD <= 0.0
     dt = tf - t;
   else
     dt0 = 0.25 * min(dx,dy)^2 / maxD;
     dt = min(dt0, tf - t);
   end
   mu_x = dt / (dx*dx);    mu_y = dt / (dy*dy);
   Tb = T + b;
   T(2:J,2:K) = T(2:J,2:K) + ...
       mu_y * Dup    .* ( Tb(2:J,3:K+1) - Tb(2:J,2:K)   ) - ...
       mu_y * Ddown  .* ( Tb(2:J,2:K)   - Tb(2:J,1:K-1) ) + ...
       mu_x * Dright .* ( Tb(3:J+1,2:K) - Tb(2:J,2:K)   ) - ...
       mu_x * Dleft  .* ( Tb(2:J,2:K)   - Tb(1:J-1,2:K) );
   T = T + F * dt;
   t = t + dt;    count = count + 1;
end
dtav = tf / count;
```

$\boxed{\texttt{diffusion.m}}$

- solves abstract diffusion equation $T_t = \nabla \cdot (D(x,y)\,\nabla(T+b))$
- user supplies diffusivity on staggered grid

verification of numerical ice flow codes

- STOP!   the code on the last slide is complicated!

verification of numerical ice flow codes

- how do we make sure an *implemented* numerical scheme is correct?

verification of numerical ice flow codes

- how do we make sure an *implemented* numerical scheme is correct?

technique 1 don't make any mistakes

verification of numerical ice flow codes

- how do we make sure an *implemented* numerical scheme is correct?

technique 1 don't make any mistakes

technique 2 compare your model with others, and hope that the outliers are the ones with errors = intercomparison

verification of numerical ice flow codes

- how do we make sure an *implemented* numerical scheme is correct?

technique 1 don't make any mistakes

technique 2 compare your model with others, and hope that the outliers are the ones with errors

technique 3 compare your model to an exact solution, and actually measure the numerical error = verification

verification of numerical ice flow codes

- yeah, BUT where do you get exact solutions for ice flow models!?
    - textbook with several SIA solutions: van der Veen (2013)
    - similarity solutions to SIA (Halfar 1983; Bueler et al 2005)
    - flowline SSA solutions (Böðvarsson, 1955; van der Veen, 1983; Bueler, 2014)
    - cross-flow SSA solution (Schoof, 2006)
    - flowline Stokes solutions for constant viscosity (Balise and Raymond 1985)
- if desperate, “manufacture” an exact solution
    - to thermo-coupled SIA (Bueler et al 2007)
    - to flowline Blatter-Pattyn (Glowinski and Rappaz 2003)
    - to the Glen-law Stokes equations (Sargent and Fastook 2010; Jouvet and Rappaz 2011; Leng et al 2014)

- yeah, BUT where do you get exact solutions for ice flow models!?
  - textbook with several SIA solutions: van der Veen (2013)
  - similarity solutions to SIA (Halfar 1983; Bueler et al 2005)
  - flowline SSA solutions (Böðvarsson, 1955; van der Veen, 1983; Bueler, 2014)
  - cross-flow SSA solution (Schoof, 2006)
  - flowline Stokes solutions for constant viscosity (Balise and Raymond 1985)
- if desperate, "manufacture" an exact solution
  - to thermo-coupled SIA (Bueler et al 2007)
  - to flowline Blatter-Pattyn (Glowinski and Rappaz 2003)
  - to the Glen-law Stokes equations (Sargent and Fastook 2010; Jouvet and Rappaz 2011; Leng et al 2014)
- Ed says:

  not having an exact solution is a sign that you don't understand the continuum model well enough

e.g. exact solution of heat equation

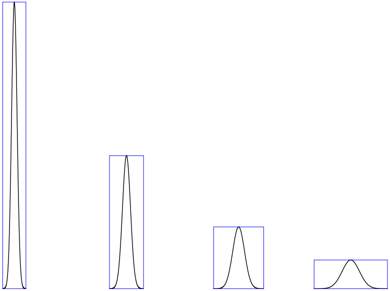
- recall 1D heat equation with constant diffusivity:

$$T_t = DT_{xx}$$

- many exact solutions to the heat equation are known
- here we find the "Green's function"
  - a.k.a. "fundamental solution" or "heat kernel"
  - starts at time $t = 0$ with a "delta function" of heat at the origin $x = 0$ and then it spreads out over time
  - we find it by a method which generalizes to the SIA

Green's function of heat equation

- the solution is "self-similar" over time
- as time increases it changes shape by
  - shrinking the output (vertical) axis and
  - lengthening the input (horizontal) axis
- . . . but otherwise it always has the same shape
  - by conservation of energy, integral over x independent of time



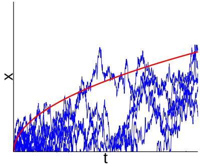
increasing time $\longrightarrow$

similarity solutions

- Green's function of 1D heat equation ($T_t = DT_{xx}$) is

$$T(t,x) = C\, t^{-1/2} e^{-x^2/(4Dt)}$$

- "similarity" variable is $s = t^{-1/2}x$
  - *historical note*: in 1905 Einstein saw that the average distance traveled by particles in thermal motion scales like $\sqrt{t}$, so $s = t^{-1/2}x$ is an invariant



- similarity solution as scaling:

$$\underbrace{s = t^{-1/2}x}_{\text{input scaling}}, \qquad \underbrace{T(t,x) = t^{-1/2}\phi(s)}_{\text{output scaling}}$$

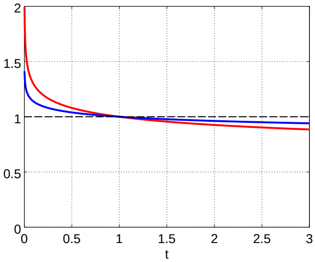
where

$$\phi(s) = Ce^{-s^2/(4D)}$$

similarity solution to SIA

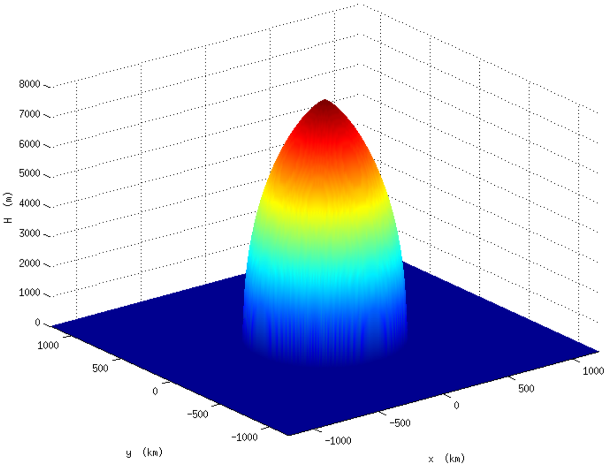
- jump forward to 1981
- P. Halfar found the similarity solution of the SIA in the case of flat bed and no surface mass balance
- Halfar's 2D solution for Glen flow law with $n = 3$ has scalings

$$H(t,r) = t^{-1/9}\phi(s), \qquad s = t^{-1/18} r$$



- ...so the diffusion of ice really slows down as time goes on

Halfar solution to the SIA: the movie



frames from $t = 4$ months to $t = 10^6$ years, equal spaced in *exponential* time

Halfar solution to the SIA: the formula

- for $n = 3$:

$$H(t,r) = H_0 \left(\frac{t_0}{t}\right)^{1/9} \left[1 - \left(\left(\frac{t_0}{t}\right)^{1/18} \frac{r}{R_0}\right)^{4/3}\right]^{3/7}$$

  where H_0, R_0 are center height and ice cap radius at $t = t_0$
- the "characteristic time" is

$$t_0 = \frac{1}{18\Gamma} \left(\frac{7}{4}\right)^3 \frac{R_0^4}{H_0^7}$$

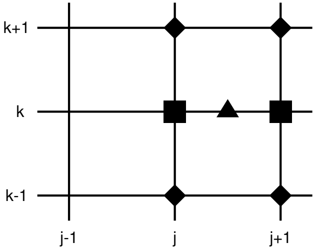
  (e.g. choose H_0 and R_0; this determines t_0)
- it is an easy formula to use for verification!
- code `verifysia.m` (not shown) uses it

computing diffusivity in SIA

- for numerical stability we compute $D = \Gamma H^{n+2} |\nabla h|^{n-1}$ on the staggered grid
- various schemes proposed (see Hindmarsh and Payne 1996)
- all schemes involve
  - averaging H
  - differencing h
  - in a "balanced" way (for better accuracy)

  to get diffusivity on staggered grid

- Mahaffy scheme $\rightarrow$



SIA implementation: flat bed case

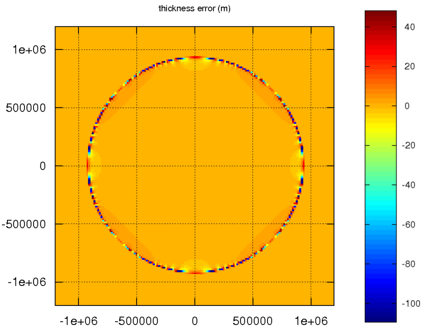
```
function [H,dtlist] = siaflat(Lx,Ly,J,K,H0,deltat,tf)

g = 9.81;    rho = 910.0;    secpera = 31556926;
A = 1.0e-16/secpera;    Gamma  = 2 * A * (rho * g)^3 / 5;
H = H0;

dx = 2 * Lx / J;    dy = 2 * Ly / K;
N = ceil(tf / deltat);    deltat = tf / N;
j  = 2:J;    k = 2:K;
nk = 3:K+1;    sk = 1:K-1;    ej = 3:J+1;    wj = 1:J-1;

t = 0;    dtlist = [];
for n=1:N
  Hup = 0.5 * ( H(j,nk) + H(j,k) );   Hdn = 0.5 * ( H(j,k) + H(j,sk) );
  Hrt = 0.5 * ( H(ej,k) + H(j,k) );   Hlt = 0.5 * ( H(j,k) + H(wj,k) );
  a2up = (H(ej,nk) + H(ej,k) - H(wj,nk) - H(wj,k)).^2 / (4*dx)^2 + ...
         (H(j,nk) - H(j,k)).^2 / dy^2;
  a2dn = (H(ej,k) + H(ej,sk) - H(wj,k) - H(wj,sk)).^2 / (4*dx)^2 + ...
         (H(j,k) - H(j,sk)).^2 / dy^2;
  a2rt = (H(ej,k) - H(j,k)).^2 / dx^2 + ...
         (H(ej,nk) + H(j,nk) - H(ej,sk) - H(j,sk)).^2 / (4*dy)^2;
  a2lt = (H(j,k) - H(wj,k)).^2 / dx^2 + ...
         (H(wj,nk) + H(j,nk) - H(wj,sk) - H(j,sk)).^2 / (4*dy)^2;
  Dup = Gamma * Hup.^5 .* a2up;  Ddn = Gamma * Hdn.^5 .* a2dn;
  Drt = Gamma * Hrt.^5 .* a2rt;  Dlt = Gamma * Hlt.^5 .* a2lt;
  [H,dtadapt] = diffusion(Lx,Ly,J,K,Dup,Ddn,Drt,Dlt,H,deltat);
  t = t + deltat;    dtlist = [dtlist dtadapt];
end
```

$\boxed{\texttt{siaflat.m}}$

- solves the $M = 0$, $b = 0$ case of SIA   (general case later)
- calls `diffusion.m` at each major time step

verifying SIA code vs Halfar

```
>> verifysia(20)
average abs error     = 22.310
maximum abs error     = 227.849
>> verifysia(40)
average abs error     = 9.490
maximum abs error     = 241.470
>> verifysia(80)
average abs error     = 2.800
maximum abs error     = 155.796
>> verifysia(160)
average abs error     = 1.059
maximum abs error     = 109.466
```



Trust but verify.

(Ronald Reagan)

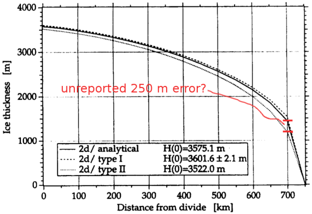


figure 2 in EISMINT I paper (1996)

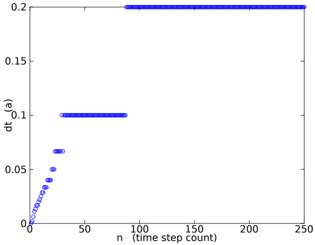
is the Halfar solution *good for any modeling*?

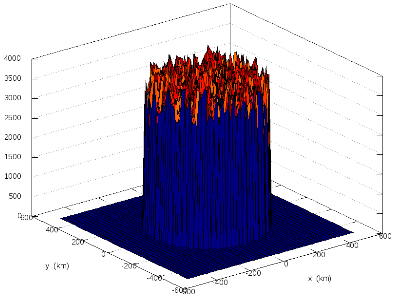
- John Nye and others (2000) compared different flow laws for the South Polar Cap on Mars
- they evaluated CO_2 ice and H_2O ice softness parameters by comparing the long-time behavior of the corresponding Halfar solutions
- conclusions:

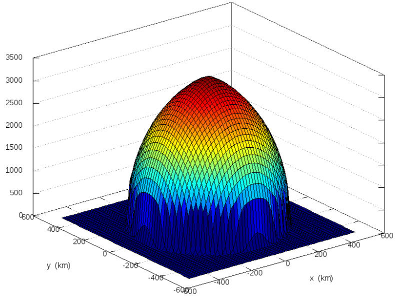
> *. . . none of the three possible [CO_2] flow laws will allow a 3000-m cap, the thickness suggested by stereogrammetry, to survive for 10^7 years, indicating that the south polar ice cap is probably not composed of pure CO_2 ice . . . the south polar cap probably consists of water ice, with an unknown admixture of dust.*

robustness comes from adaptivity

- `roughice.m` (not shown) sets-up nasty initial state (below)
- then calls `siaflat.m` for 50 year run
- adaptive time steps (right)
- final state (below right)





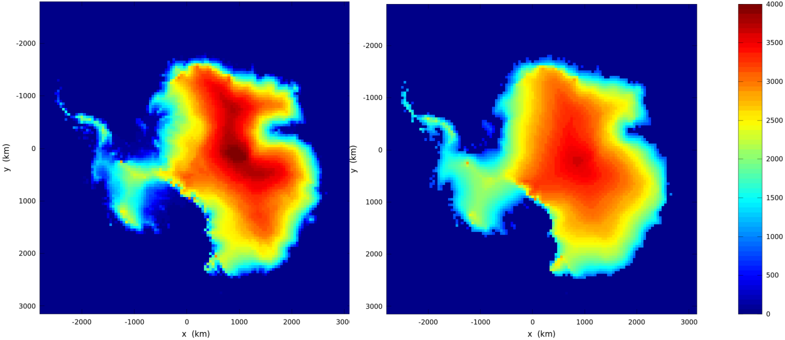


model the Antarctic ice sheet

- we are close to a first model of Antarctic ice sheet
  - it's just a toy because it's isothermal SIA
  - ... but more work needed, as follows
- careful-but-small modifications of `siaflat.m`:
  - observed accumulation as surface mass balance,
  - allow non-flat bed (so $H \neq h$),
  - remove (calve) all floating ice,

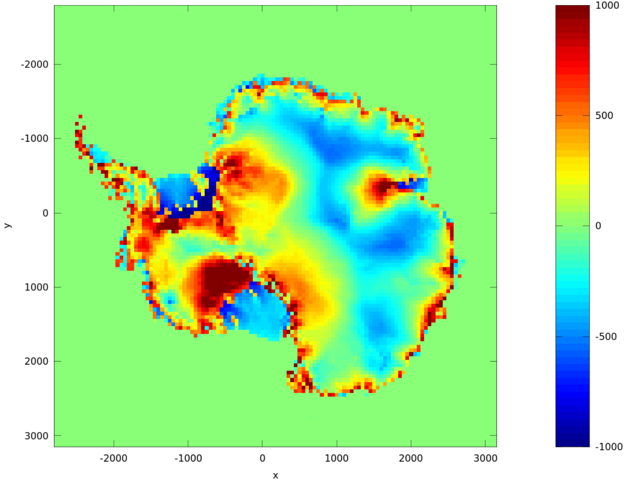
  gives `siageneral.m` (not shown)
- also write code `buildant.m` (not shown) which reads NetCDF file from Le Brocq et al (2010) = ALBMAPv1

- do 40 ka run on a $\Delta x = 50$ km grid
- runtime 4 minutes on laptop
- initial and final surface elevation below



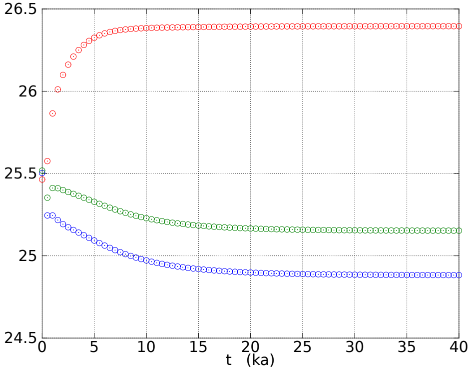
model of the Antarctic ice sheet

- thickness change from beginning to end of 50 km run
- . . . it's nearly a map of where flow is fast!
  - why?



model of the Antarctic ice sheet

- volume time series from 50 km, 25 km, 20 km runs
- units of 10^6 km^3
- conclusion: look at your results on *multiple grid resolutions*
- ...*before* interpreting your results (e.g. for their parameter-dependence)



Outline

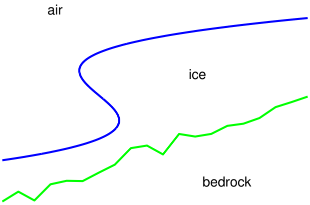
1. introduction: ice flow as viewed from outside glaciology
2. shallow ice sheets
3. **mass continuity**
4. shelves and streams
5. conclusions

the most-basic shallow assumption

- there are many shallow theories
  - e.g. SIA, SSA, hybrids, Blatter-Pattyn, ...
- *all* make one assumption not required in Stokes:

  the surface and base of the ice are given by functions $z = h(t,x,y)$ and $z = b(t,x,y)$

- surface overhang is not allowed
  - by contrast, you can solve Stokes on any old blob
- this most-basic assumption has consequences ... next



not shallow!

not shallow!

three equations for ice sheet geometry change

- define:
  - $a =$ the surface mass balance rate ($a > 0$ *is accumulation*)
  - $s =$ the basal melt rate ($s > 0$ *is basal melting*)
  - $\mathbf{q} =$ the map-plane flux of ice:

$$\mathbf{q} = \int_b^h (u, v)\, dz = \bar{\mathbf{U}}\, H$$

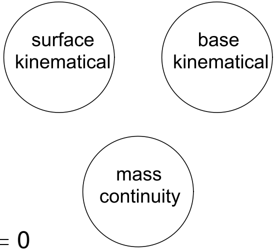
- then there are *three* equations for geometry change:

surface kinematical	$h_t = a - u\big\|_h h_x - v\big\|_h h_y + w\big\|_h$
base kinematical	$b_t = s - u\big\|_b b_x - v\big\|_b b_y + w\big\|_b$
mass continuity	$H_t = (a - s) - \nabla \cdot \mathbf{q}$

- $M = a - s$ is called the "climatic-basal mass balance function"

kinematic and mass continuity equations

- what does the “most basic shallow assumption” get you?:
  1. you can derive mass continuity equation from the kinematical equations and incompressibility
  2. in fact, any pair of these 3 equations implies the other:



- to show such equivalences, recall:
  - incompressibility

$$u_x + v_y + w_z = 0$$

  - and Leibniz rule for differentiating integrals

$$\frac{d}{dx}\left(\int_{g(x)}^{f(x)} h(x,y)\,dy\right) = f'(x)h(x,f(x)) - g'(x)h(x,g(x)) + \int_{g(x)}^{f(x)} h_x(x,y)\,dy$$

- see exercises in notes

kinematic and mass continuity equations 2

- why tell you this?:
  - literature is full of incomplete calculations of such equivalences
  - . . . often mixed in with small-parameter arguments about shallowness (confusing)
- most ice sheet models use the mass continuity equation
  - . . . but they could instead use the surface kinematical equation

standard recipe for ice sheet models

- the ingredients of a typical ice sheet model:
  1. numerical solution of a stress balance: compute velocity (u, v)
  2. compute vertical velocity w from incompressibility
  3. from the horizontal velocity (u, v) and the surface balance, do a time-step of mass continuity equation to get H_t, thus $\Delta H = H_t \Delta t$
  4. update surface elevation (assumes $b_t = 0$): $h^{n+1} = h^n + \Delta H$
  5. decide on next time-step
  6. repeat

standard recipe for ice sheet models

- the ingredients of a typical ice sheet model:
  1. numerical solution of a stress balance: compute velocity (u, v)
  2. compute vertical velocity w from incompressibility
  3. from the horizontal velocity (u, v) and the surface balance, do a time-step of mass continuity equation to get H_t, thus $\Delta H = H_t \Delta t$
  4. update surface elevation (assumes $b_t = 0$): $\quad h^{n+1} = h^n + \Delta H$
  5. decide on next time-step
  6. repeat

- the SIA is atypical because we can write $\mathbf{q} = -D\nabla h$, in addition to $\mathbf{q} = \bar{\mathbf{U}} H$, and apparently skip steps 1 and 2 above

standard recipe for ice sheet models

- the ingredients of a typical ice sheet model:
  1. numerical solution of a stress balance: compute velocity (u, v)
  2. compute vertical velocity w from incompressibility
  3. from the horizontal velocity (u, v) and the surface balance, do a time-step of mass continuity equation to get H_t, thus $\Delta H = H_t \Delta t$
  4. update surface elevation (assumes $b_t = 0$): $\quad h^{n+1} = h^n + \Delta H$
  5. decide on next time-step
  6. repeat

- the SIA is atypical because we can write $\mathbf{q} = -D\nabla h$, in addition to $\mathbf{q} = \bar{\mathbf{U}}H$, and apparently skip steps 1 and 2 above

- often models also solve conservation of energy . . . and surface processes . . . and perhaps calving laws . . . and perhaps subglacial hydrology . . . additional steps . . .

Outline

1. introduction: ice flow as viewed from outside glaciology
2. shallow ice sheets
3. mass continuity
4. **shelves and streams**
5. conclusions

flow model II: shallow shelf approximation stress balance (= SSA)

SSA model applies very well to ice shelves

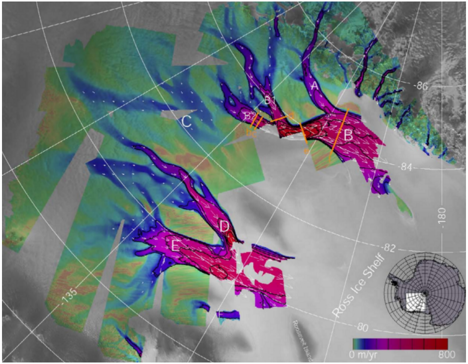
- away from grounding lines
- . . . and calving fronts

edge of Ekström ice shelf (Hans Grobe)

shallow shelf approximation stress balance 2

SSA also applies reasonably well to ice streams

- with modest bed topography
- . . . and weak bed strength[1]
- imperfect near shear margins and grounding lines

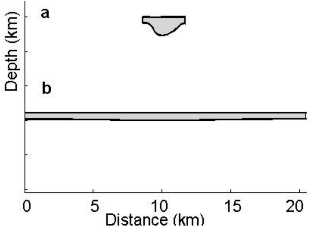


surface velocity for Siple Coast ice streams, Antarctica

[1] energy conservation (esp. basal melt rate) and subglacial hydrology (esp. effective pressure) are major aspects of models of ice stream flow . . . *not addressed here*

what is, *and is not*, an ice stream?

- ice streams
  - slide (100–1000 m a^{-1})
  - have concentrated vertical shear in thin layer near base
- "outlet glaciers"
  - fast surface speed ($\sim$ 10 km a^{-1})
    - . . . not clear how much is sliding
  - substantial bed topography
  - thick layer of soft temperate ice
  - substantial vertical shear "up" in the ice column
- few simplifying assumptions are appropriate for outlet glaciers



Jakobshavns Isbrae (**a**) and Whillans Ice Stream (**b**); plotted without vertical exaggeration (Truffer and Echelmeyer (2003), *Of isbrae and ice streams*)

SSA stress balance equation

- only plane flow ("flow line") case here
- this equation determines velocity u in an *ice stream*:

$$\boxed{\left(2A^{-1/n}H|u_x|^{1/n-1}u_x\right)_x - C|u|^{m-1}u = \rho g H h_x} \qquad (3)$$

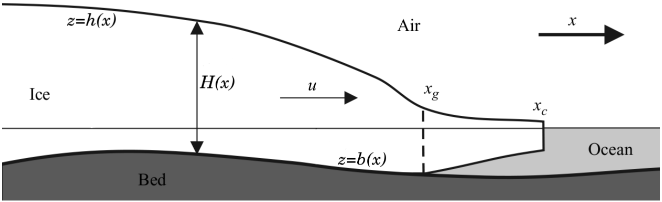
- the red term inside parentheses is the vertically-integrated "longitudinal" or "membrane" stress
- the blue term is basal resistance
  - this term is zero in an *ice shelf*
- the green term is driving stress

- *how to think about this equation*?
- *how do you solve it numerically*?

SSA context: from stream to shelf

- here is what a 1D "marine ice sheet model" looks like:

$$
\begin{aligned}
u &= u_0 && \text{at } x = 0 \\
\left. \begin{aligned} \left(2A^{-1/n}H|u_x|^{1/n-1}u_x\right)_x - C|u|^{m-1}u &= \rho g H h_x \\ h &= H + b \end{aligned} \right\} && \text{on } 0 < x < x_g \\
\left. \begin{aligned} \left(2A^{-1/n}H|u_x|^{1/n-1}u_x\right)_x + 0 &= \rho g H h_x \\ h &= (1 - \rho/\rho_w)H \end{aligned} \right\} && \text{on } x_g < x < x_c \\
2A^{-1/n}H|u_x|^{1/n-1}u_x &= \frac{1}{2}\rho(1-\rho/\rho_w)gH^2 && \text{at } x = x_c
\end{aligned}
$$



flotation criterion and grounding line

- inequality "$\rho H < -\rho_w b$" for floating ice is the flotation criterion
- at the grounding line $x = x_g$ get equality: $\rho H = -\rho_w b$
- the driving stress switches form at the grounding line:
  - on the grounded side:

$$\rho g H h_x = \rho g H (H_x + b_x)$$

  - on the floating side, since Archimedes says $h = (1 - \rho/\rho_w)H$:

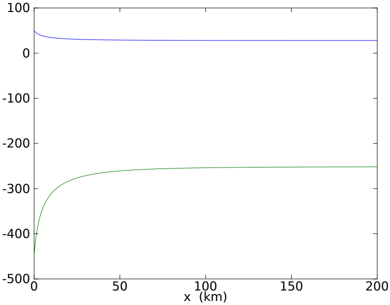
$$\rho g H h_x = \rho (1 - \rho/\rho_w) g H H_x$$

exact velocity and thickness for steady ice shelf

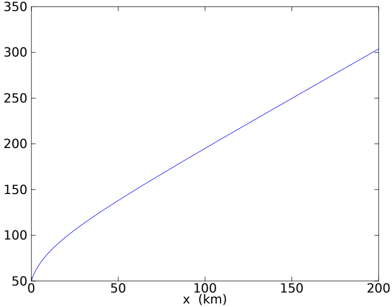
- limited goal here: describe a steady state, 1D ice shelf
- it is a nice by-hand exact solution (next slide; see exercises) to both the stress balance and mass continuity equations
- in 1D the thickness and velocity functions in the ice shelf are determined by just two numbers at grounding line $x_g = 0$:
  - thickness H_g
  - velocity u_g
- we will use this to
  - understand the SSA better
  - verify a numerical SSA code

exact velocity and thickness for steady ice shelf

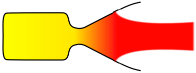
- see `testshelf.m`
- here $H_g = 500$ m, $u_g = 50$ m/a



$h(x)$ and $b(x)$



velocity $u(x)$



numerically solving the SSA stress balance

- for an example, we fix ice thickness $H(x)$ and find the velocity numerically
- recall the stress balance is a nonlinear equation for velocity:

$$\left(2A^{-1/n}H|u_x|^{1/n-1}u_x\right)_x - C|u|^{m-1}u = \rho g H h_x$$

- iteration is needed
- I'll describe the numerical method for a shelf *or* stream, but only give a code for an ice shelf

numerically solving the SSA stress balance 2

- coefficient $\bar\nu = A^{-1/n}|u_x|^{1/n-1}$ is the "effective viscosity":

$$(2\,\bar\nu\,Hu_x)_x - C|u|^{m-1}u = \rho g H h_x$$

- *simplest iteration idea*: use old effective viscosity to get new velocity solution, and repeat until things stop changing
  - this is "Picard" iteration
  - Newton iteration is a superior alternative
- recipe starts with initial iterate $u^{(0)}$, then:
  - define $\bar\nu^{(k-1)} = A^{-1/n}|u_x^{(k-1)}|^{1/n-1}$ from last iterate $u^{(k-1)}$
  - current iterate (unknown) $u^{(k)}$
  - solve repeatedly:

$$\left(2\bar\nu^{(k-1)}Hu_x^{(k)}\right)_x - C|u^{(k-1)}|^{m-1}u^{(k)} = \rho g H h_x$$

solving the "inner" linear problem

- abstract the problem:

$$(W(x)\, u_x)_x - \alpha(x)\, u = \beta(x)$$

on $0 < x < L$, with boundary conditions

$$u(0) = V, \qquad u_x(L) = \gamma$$

- an *elliptic* PDE boundary value problem
- $W(x)$, $\alpha(x)$, $\beta(x)$ are known functions in the SSA context:
  - both $W(x)$ and $\alpha(x)$ come from previous iteration
  - $\beta(x)$ is driving stress

where do you get an initial guess $u^{(0)}$?

- *for floating ice*, use velocity from assuming a uniform strain rate:
$$u^{(0)}(x) = \gamma(x - x_g) + u_g$$
where γ is the value of u_x found from calving front stress imbalance
- *for grounded ice*, use velocity from assuming ice is held by basal resistance only:
$$u^{(0)}(x) = \left(-C^{-1}\rho g H h_x\right)^{1/m}$$

numerics of the "inner" linear problem

- index $j = 1, 2, \dots, J + 1$
- $x_1 = x_g$ and $x_{J+1} = x_c$ are endpoints
- $W(x)$ is needed on the staggered grid; the approximation is:

$$\frac{W_{j+1/2}(u_{j+1} - u_j) - W_{j-1/2}(u_j - u_{j-1})}{\Delta x^2} - \alpha_j u_j \overset{*}{=} \beta_j$$

- left-hand boundary condition: $u_1 = V$ given
- right-hand boundary condition ("$u_x(L) = \gamma$"):
  - introduce notional point x_{J+2} and use equation

$$\frac{u_{J+2} - u_J}{2\Delta x} = \gamma$$

  - using equation $*$ in $j = J + 1$ case, eliminate u_{J+2} variable "by-hand" before coding numerics

numerics of the "inner" linear problem 2

- so SSA stress balance has form $A\mathbf{x} = \mathbf{b}$, namely:

$$\begin{bmatrix} 1 & & & & & \\ W_{3/2} & A_{22} & W_{5/2} & & & \\ & W_{5/2} & A_{33} & & & \\ & & \ddots & \ddots & & \\ & & W_{J-1/2} & A_{JJ} & W_{J+1/2} & \\ & & & A_{J+1,J} & A_{J+1,J+1} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_J \\ u_{J+1} \end{bmatrix} = \begin{bmatrix} 0 \\ \beta_2 \Delta x^2 \\ \beta_3 \Delta x^2 \\ \vdots \\ \beta_J \Delta x^2 \\ b_{J+1} \end{bmatrix}$$

with diagonal entries

$$A_{22} = -(W_{3/2} + W_{5/2} + \alpha_1 \Delta x^2)$$

$$A_{33} = -(W_{5/2} + W_{7/2} + \alpha_2 \Delta x^2)$$

and so on, up to A_{JJ}, and with special cases in last equation:

$$A_{J+1,J} = 2W_{J+1/2}$$

$$A_{J+1,J+1} = -(2W_{J+1/2} + \alpha_{J+1} \Delta x^2)$$

$$b_{J+1} = -2\gamma \Delta x W_{J+3/2} + \beta_{J+1} \Delta x^2$$

- this is a *tridiagonal* linear system
- give it to a matrix-solving black box!

numerics of the "inner" linear problem 3

```
function u = flowline(L,J,gamma,W,alpha,beta,V0)

dx = L / J;
rhs = dx^2 * beta(:);
rhs(1) = V0;
rhs(J+1) = rhs(J+1) - 2 * gamma * dx * W(J+1);

A = sparse(J+1,J+1);
A(1,1) = 1.0;
for j=2:J
  A(j,j-1:j+1) = [ W(j-1), -(W(j-1) + W(j) + alpha(j) * dx^2), W(j) ];
end
A(J+1,J) = W(J) + W(J+1);
A(J+1,J+1) = - (W(J) + W(J+1) + alpha(J+1) * dx^2);

scale = full(max(abs(A),[],2));
for j=1:J+1,  A(j,:) = A(j,:) ./ scale(j);  end
rhs = rhs ./ scale;

u = A \ rhs;
```

flowline.m

- solves

$$(W(x)u_x)_x - \alpha(x)u = \beta(x)$$

testing the "inner" linear code

- before proceeding to solve nonlinear SSA problem, we can test the "abstracted" code `flowline.m`
- test by "manufacturing" solutions
  - see `testflowline.m`; not shown
- results:
  - converges at optimal rate $O(\Delta x^2)$

numerical SSA

```
function [u,u0] = ssaflowline(p,J,H,b,ug,initchoice)

if nargin ~= 6, error('exactly 6 input arguments required'), end

dx = p.L / J;   x = (0:dx:p.L)';
xstag = (dx/2:dx:p.L+dx/2)';

alpha = p.C * ones(size(x));
h = H + b;
hx = regslope(dx,h);
beta = p.rho * p.g * H .* hx;
gamma = ( 0.25 * p.A^(1/p.n) * (1 - p.rho/p.rhow) *...
          p.rho * p.g * H(end) )^p.n;

u0 = ssainit(p,x,beta,gamma,initchoice);  u = u0;

Hstag = stagav(H);
tol = 1.0e-14;
eps_reg = (1.0 / p.secpera) / p.L;
maxdiff = Inf;  W = zeros(J+1,1);  iter = 0;
while maxdiff > tol
  uxstag = stagslope(dx,u);
  sqr_ux_reg = uxstag.^2 + eps_reg^2;
  W(1:J) = 2 * p.A^(-1/p.n) * Hstag .* sqr_ux_reg.^(((1/p.n)-1)/2.0);
  W(J+1) = W(J);

  unew = flowline(p.L,J,gamma,W,alpha,beta,ug);
  maxdiff = max(abs(unew-u));
  u = unew;
  iter = iter + 1;
end
```

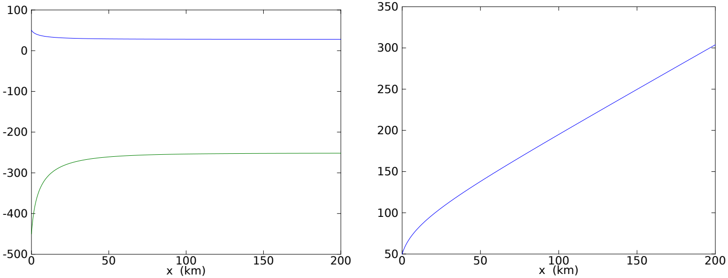
$\boxed{\texttt{ssaflowline.m}}$

- solves

$$\left(2A^{-1/n}H|u_x|^{1/n-1}u_x\right)_x - C|u|^{m-1}u = \rho g H h_x$$

numerical SSA model results

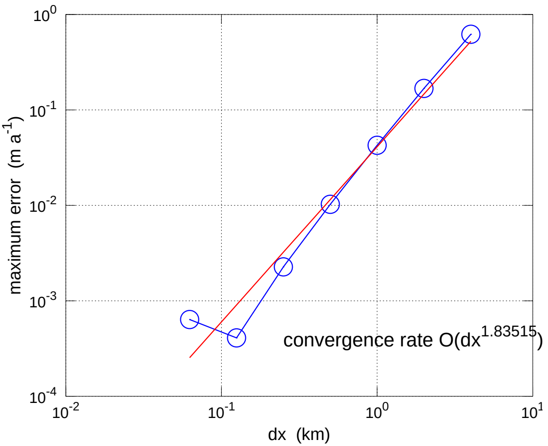
- `testshelf.m` (not shown) applies `ssaflowline.m` with same boundary conditions as exact solution; get



- *this looks suspiciously like figures for the exact solution . . .*
- yes

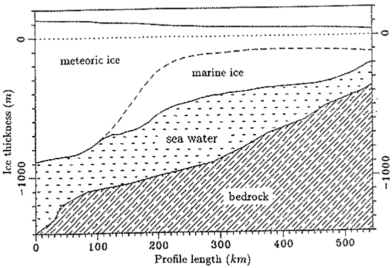
numerical thickness and velocity for steady ice shelf

- "convergence analysis" means looking at exact/numerical difference as grid is refined
- below: convergence analysis of velocity from `testshelf.m`



realistic ice shelf modeling

- flow lines are never very realistic
- real ice shelves have "side drag"
  - one can parameterize it . . .
- real ice shelves have more processes:
  - high basal melt near grounding lines
  - marine ice freeze-on at bottom (below)
  - fractures and calving
- real ice shelves have nontrivial dynamics:
  - "reverse slope" bed instability and WAIS . . .

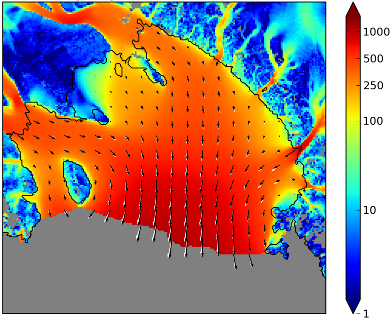


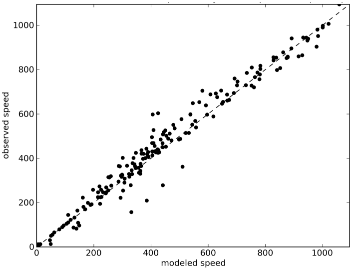
from Grosfeld & Thyssen 1994

ice shelf modeling in 2D

- nonetheless "diagnostic" (static geometry) ice shelf modeling in 2D has been quite successful
- observed surface velocities validate SSA model
  - e.g. Ross ice shelf example below using PISM
  - . . . but many models can do this

ice velocity (m/year); white=observed, black=model





Outline

1. introduction: ice flow as viewed from outside glaciology
2. shallow ice sheets
3. mass continuity
4. shelves and streams
5. conclusions

numerical solution of stress balances: a summary

- stress balance equations:
  - the non-sliding SIA stress balance is just a formula for velocity (from a vertical integral)
  - the SSA stress balance for ice streams and shelves is a PDE for velocity:

$$\left(2A^{-1/n}H|u_x|^{1/n-1}u_x\right)_x - C|u|^{m-1}u = \rho g H h_x$$

- stress balance equations like SSA, Blatter-Pattyn, Stokes:
  - are equations which determine velocity from geometry and boundary conditions
  - are nonlinear, so iteration is necessary
  - require solving a sparse matrix "inner" problem at each iteration
  - which requires a matrix-solver software package

the mass continuity equation: a summary

- the mass continuity equation[2] is

$$H_t = M - \nabla \cdot (\bar{\mathbf{U}} H) \qquad (*)$$

- $\mathbf{q} = \bar{\mathbf{U}} H$ is the *map-plane ice flux*
- the character/nature of $(*)$ depends on the stress balance:
  - it is diffusive for non-sliding, large scale flows
    - using the SIA stress balance makes it exactly a diffusion: $\mathbf{q} = -D\nabla h$
    - with other stress balances it is nearly a diffusion
  - it is not very diffusive for ice shelves
    - use SSA or other membrane stress balance
  - ice streams are an intermediate case
    - use SSA or other membrane stress balance

[2] there is not much helpful theory on this transport problem because it is not hyperbolic and you have to handle the free boundary in an *ad hoc* way

The End

- general principles
  - return often to the continuum model
  - modularize your code
  - test the parts:
    - is the component robust?
    - does it show convergence?
- that's the end
- questions?