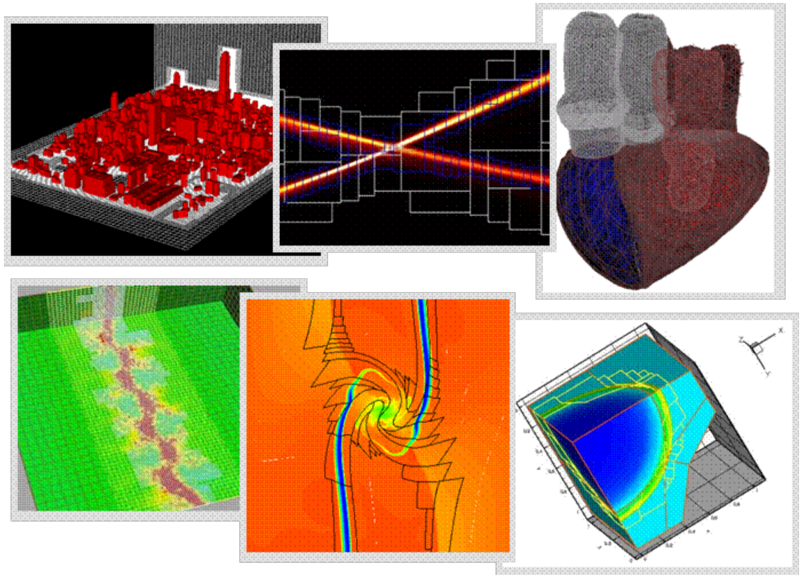
Simulating Grounding Line Motion in a 1D AMR (Adaptive Mesh Refinement) Ice Sheet Model

Rupert Gladstone[1], Victoria Lee[1],
Tony Payne[1], Andreas Vieli[2]

- (1) University of Bristol, UK
- (2) University of Durham, UK



Talk structure

1. Motivation
2. What we did
3. Conclusions

Marine ice sheet instability

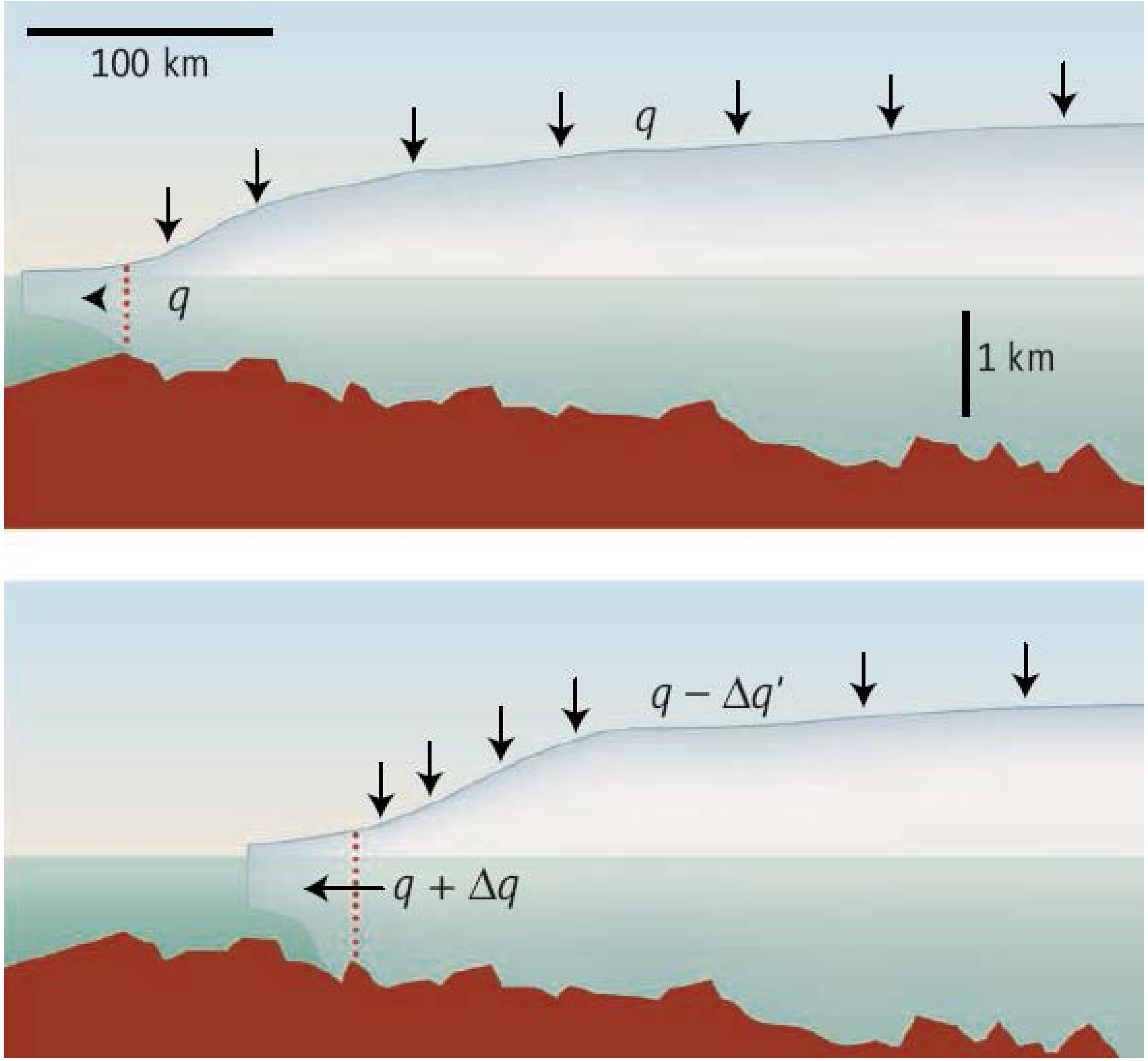
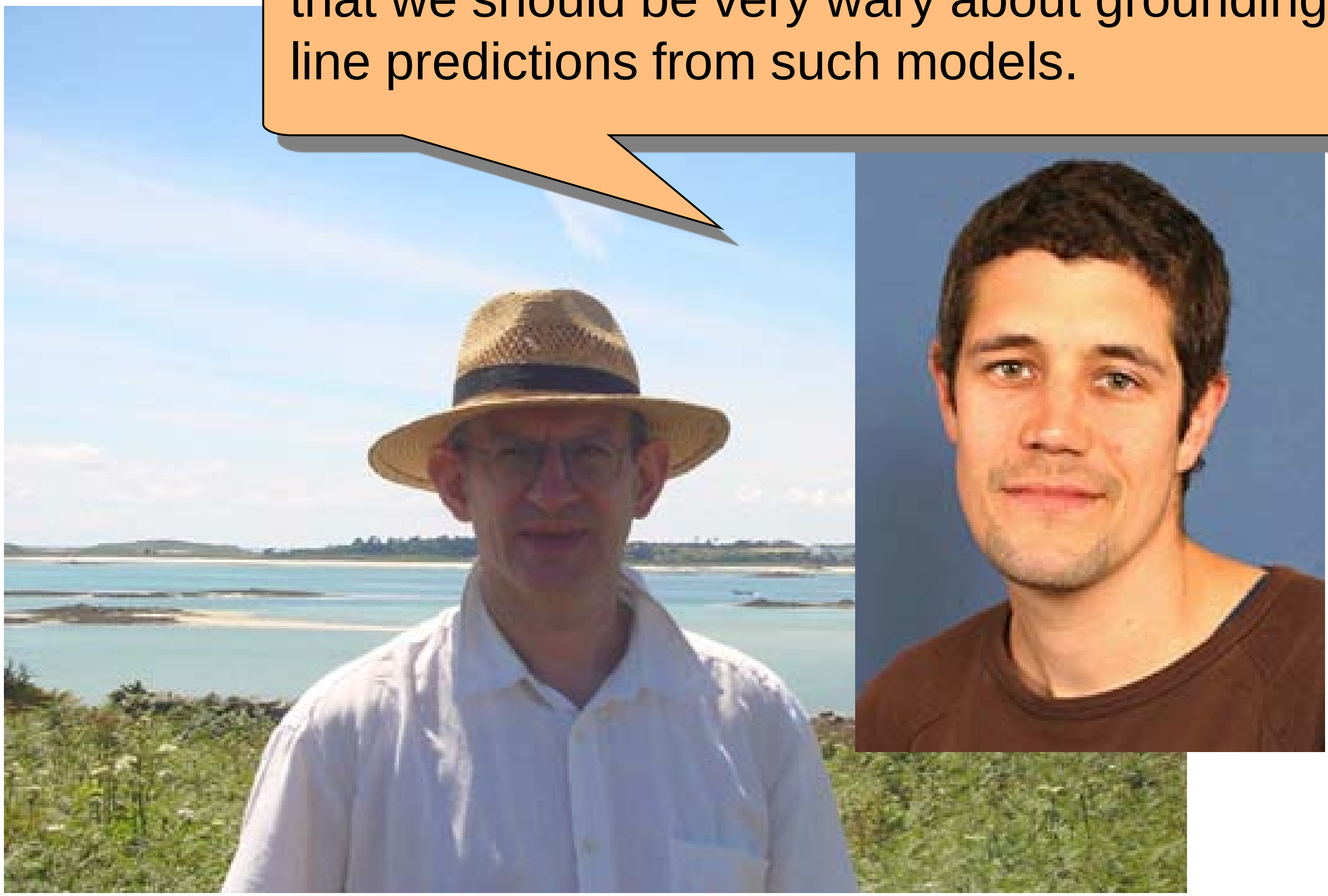


Illustration by David Vaughan, British Antarctic Survey

A crucial finding is the strong **dependency** of models using a **fixed grid** on numerical details such as the **horizontal grid size**. This implies that we should be very wary about grounding line predictions from such models.



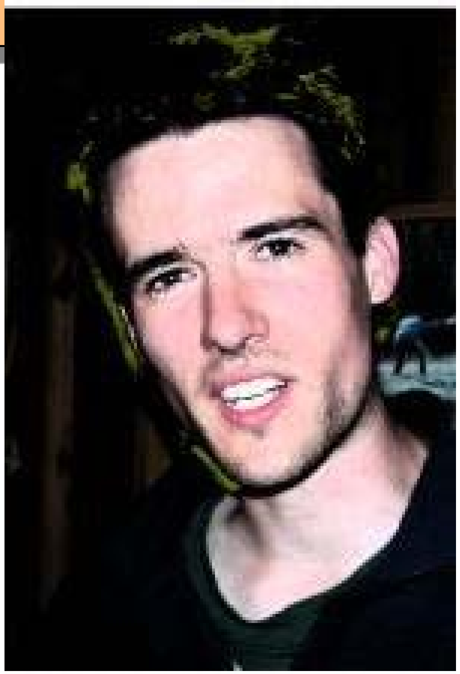
Andreas Vieli and Tony Payne “Assessing the ability of numerical ice sheet models to simulate grounding line migration” JGR 2005 the position of the grounding line is determined with **subgrid precision** based on the flotation criterion.

$$f = \frac{(z_{sl} - b)\rho_w}{\rho_i h}.$$

$$x_g = \frac{1 - f_j + \nabla f \ x_j}{\nabla f},$$

Frank Pattyn, Ann Huyghe, Sang De Brabander, and Bert De Smedt “Role of transition zones in marine ice sheet dynamics”, JGR 2006

Grid refinement in the transition zone is identified as a **critical** component in obtaining reliable numerical results.



Christian Schoof, “Ice sheet grounding line dynamics: steady states, stability and hysteresis”, JGR, 2007

Talk structure

1. Motivation
2. What we did
3. Conclusions

Our Aims

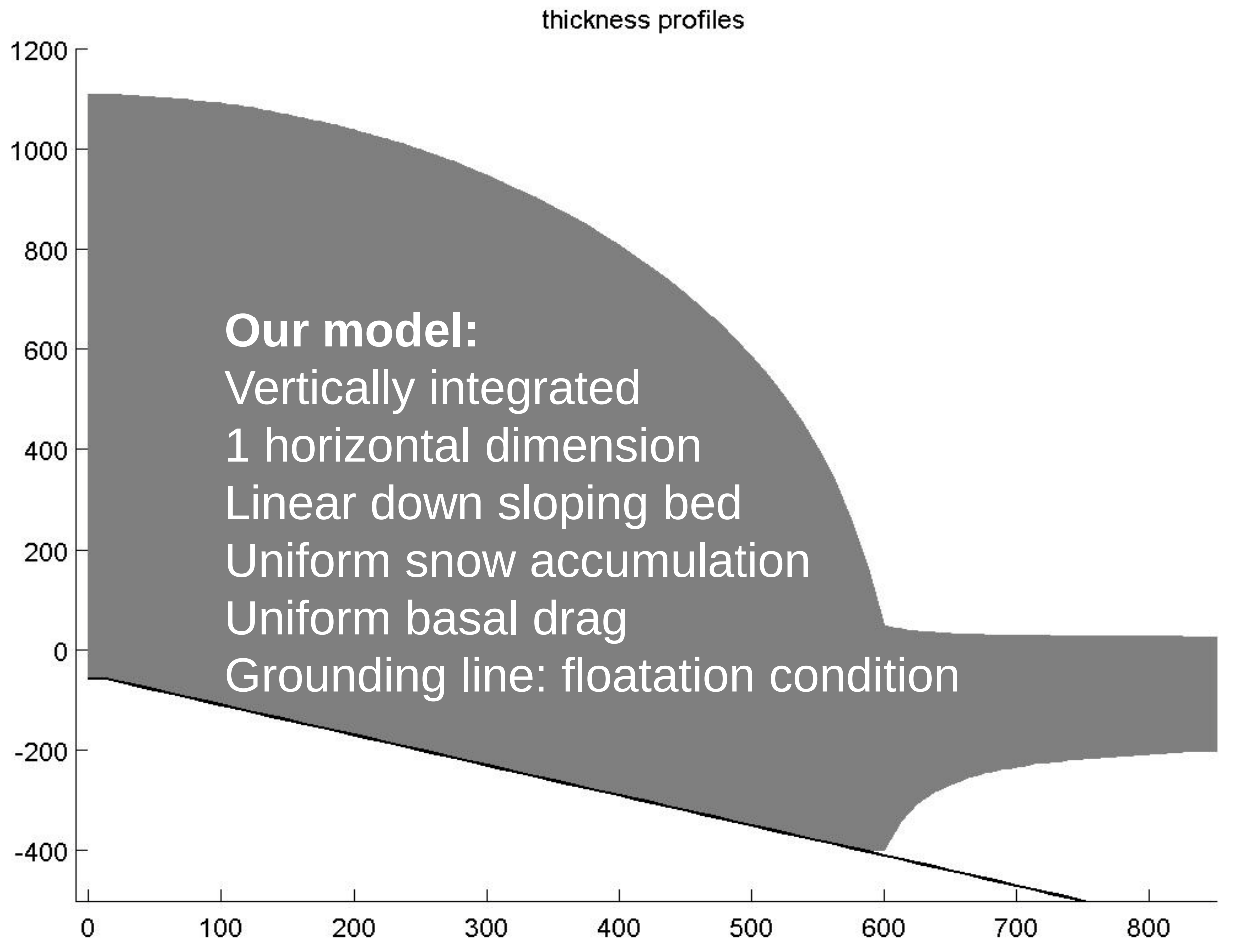
Enhance Vieli & Payne fixed grid model:

- Implement mesh refinement.
- Implement grounding line parameterisation.

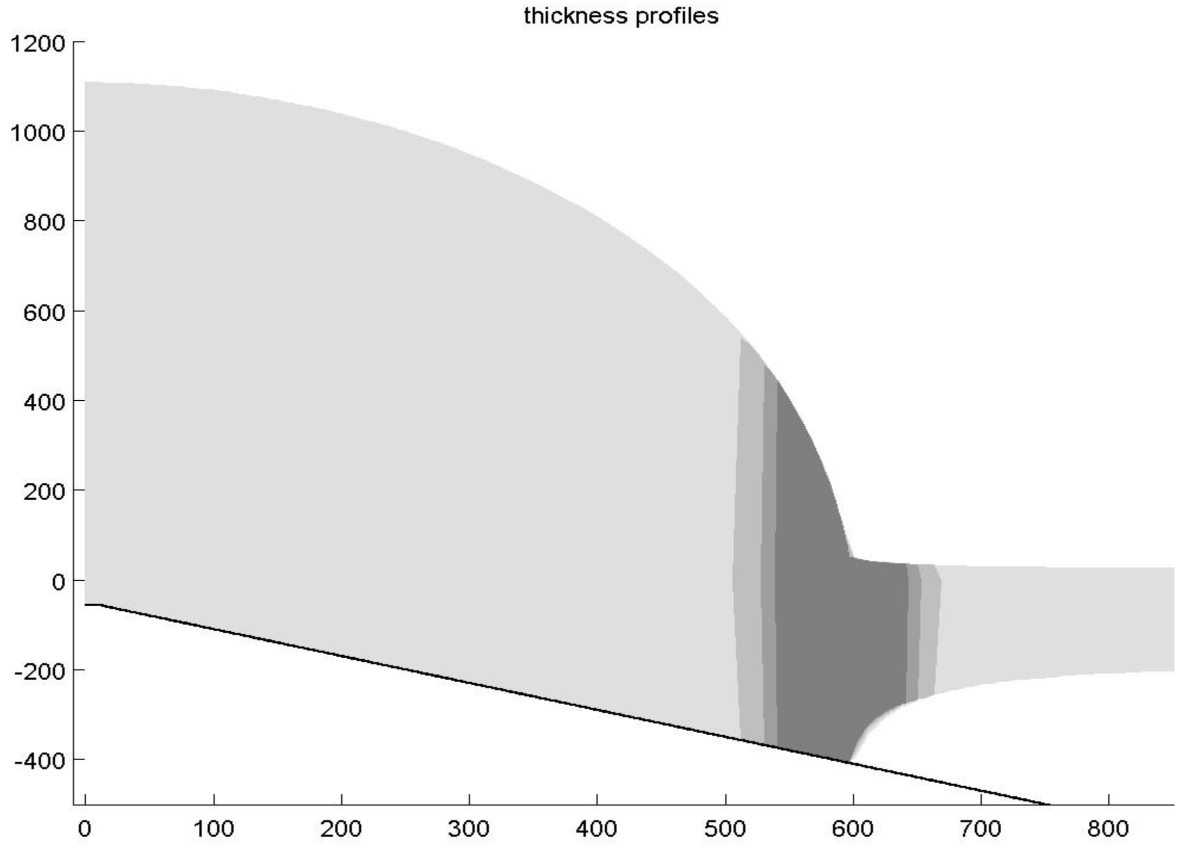
Assess performance:

- Accuracy - match Schoof predictions.
- Convergence with increasing resolution.

Our 1D model


thickness profiles
1200
1000
800
600
400
200
0
-200
-400
0
100
200
300
400
500
600
700
800
Our model:
Vertically integrated
1 horizontal dimension
Linear down sloping bed
Uniform snow accumulation
Uniform basal drag
Grounding line: floatation condition

Mesh refinement



Experiments

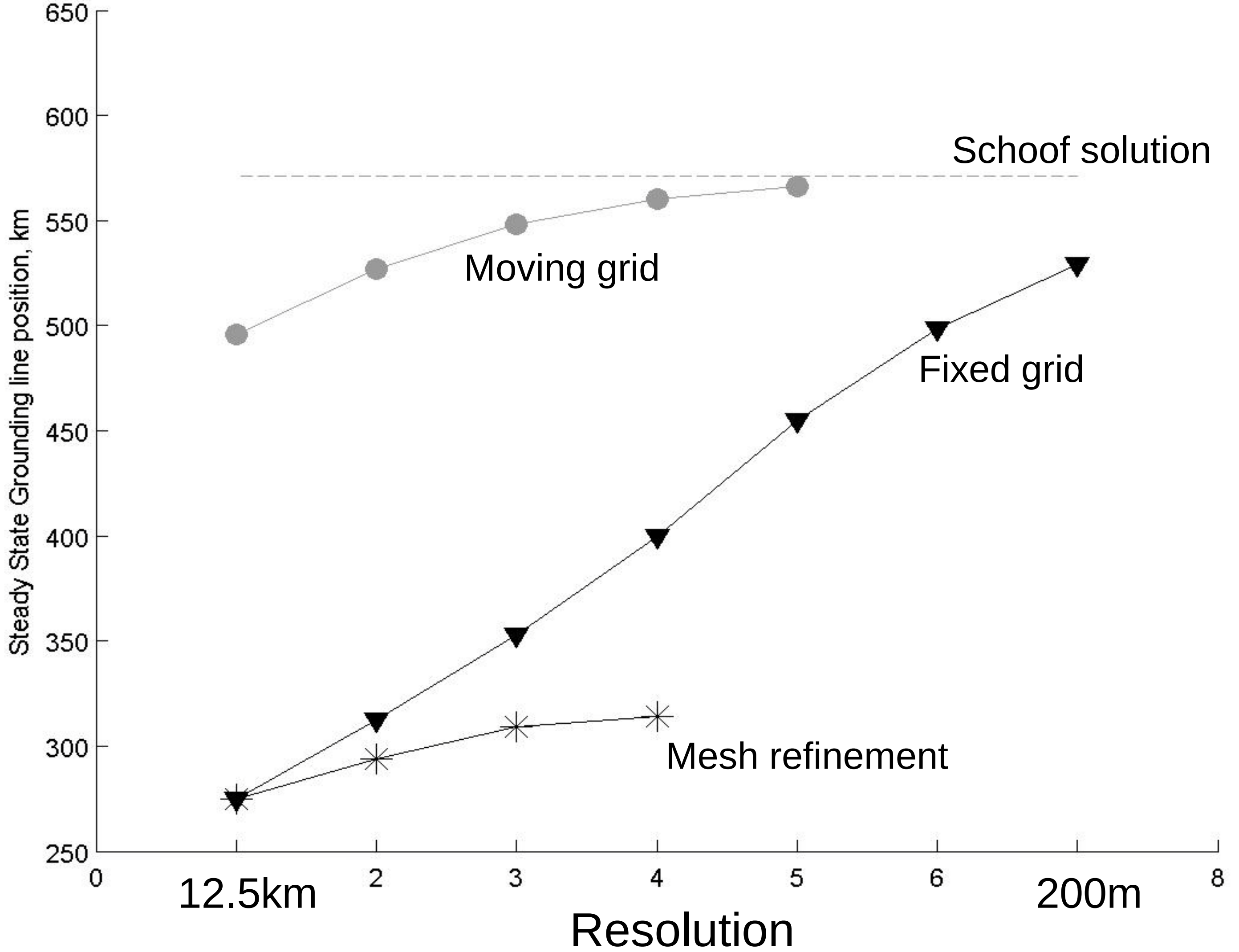
Grounding line advance experiments:

- Start from 200m thick slab.
- Constant accumulation.
- Spin up to steady state (25 – 50 ka).

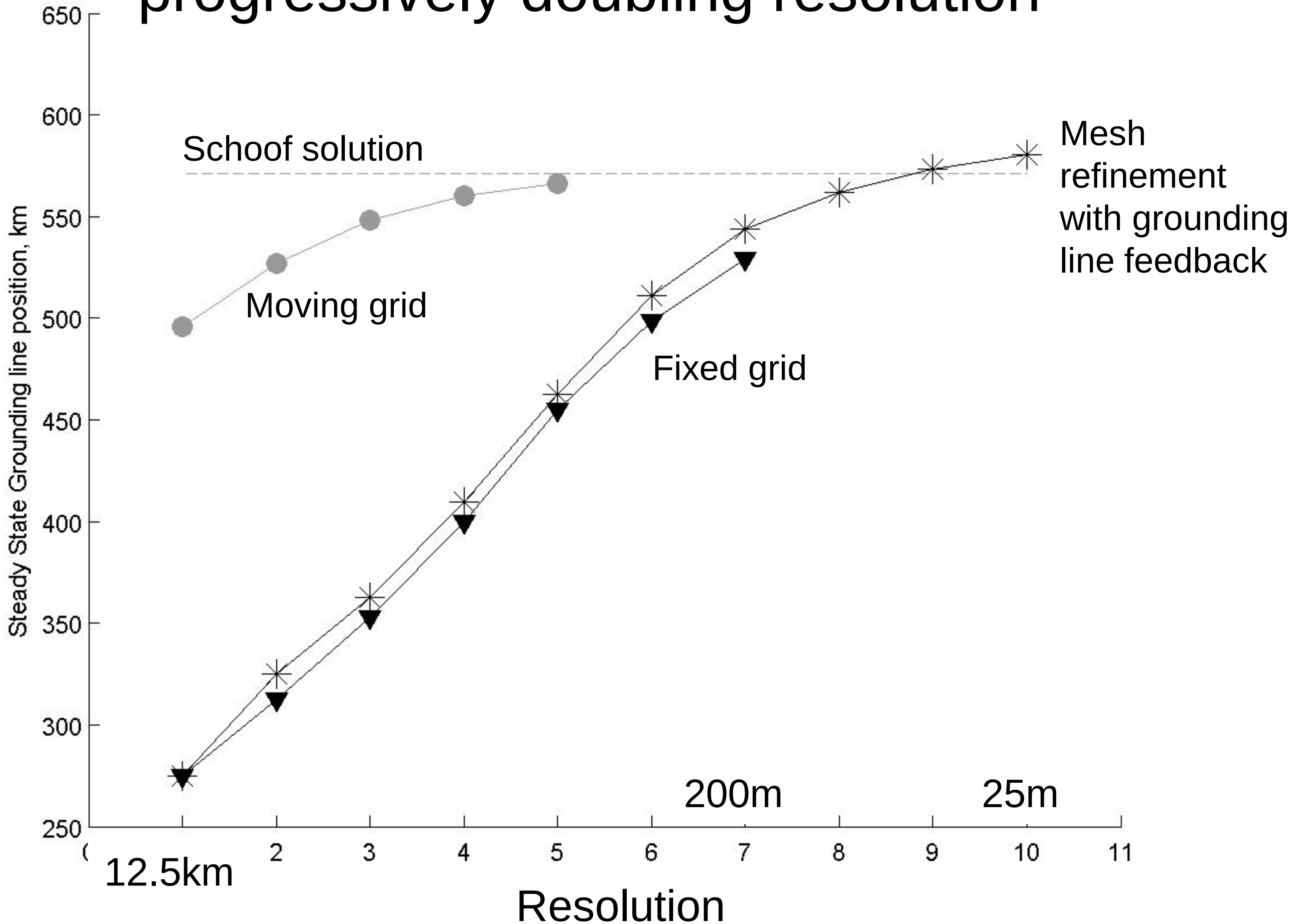
Grounding line retreat experiments:

- Start from 200m thick slab.
- High initial accumulation, spin up to steady state.
- Reset accumulation to and continue to new steady state.

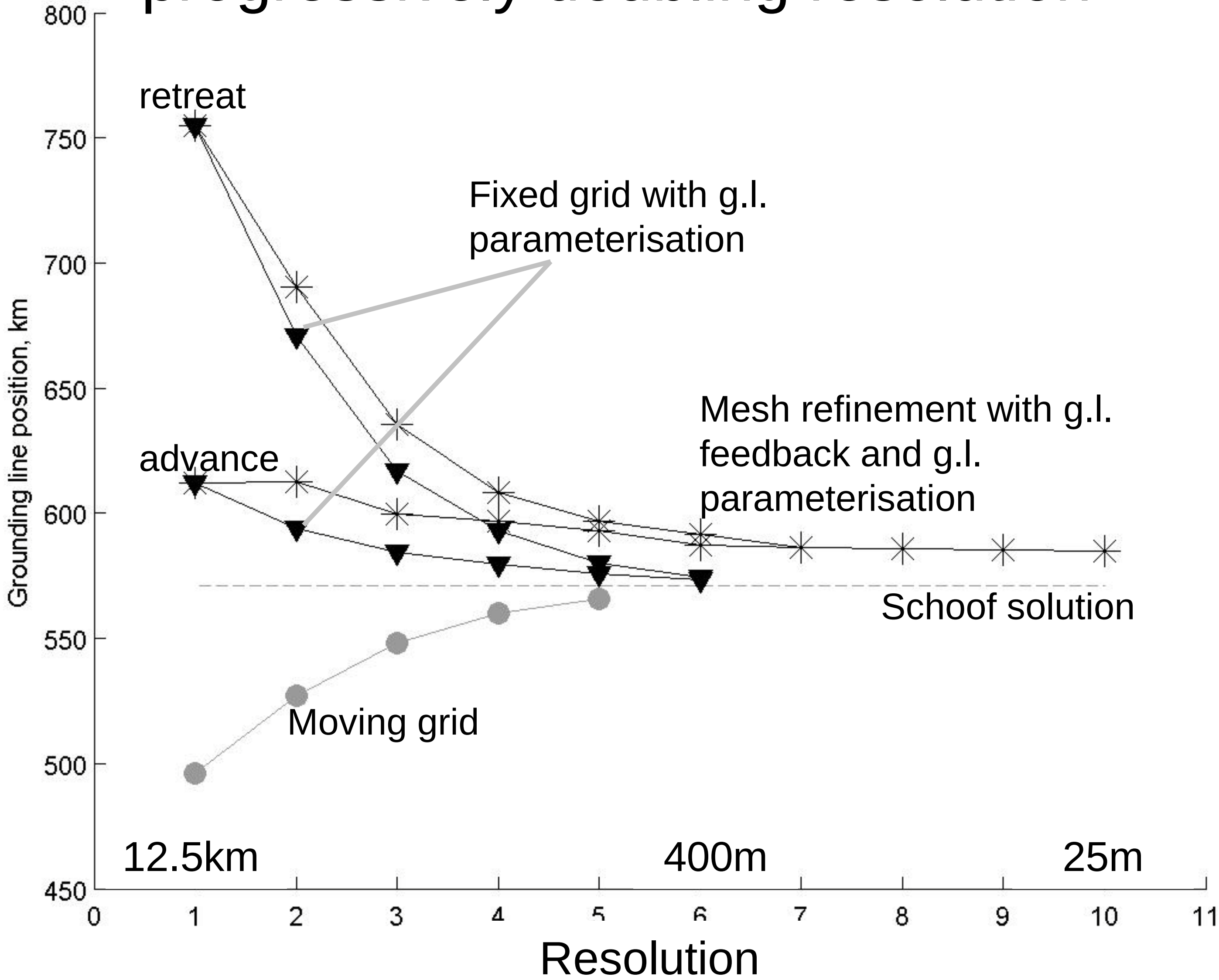
Steady state grounding line position with progressively doubling resolution



Steady state grounding line position with progressively doubling resolution



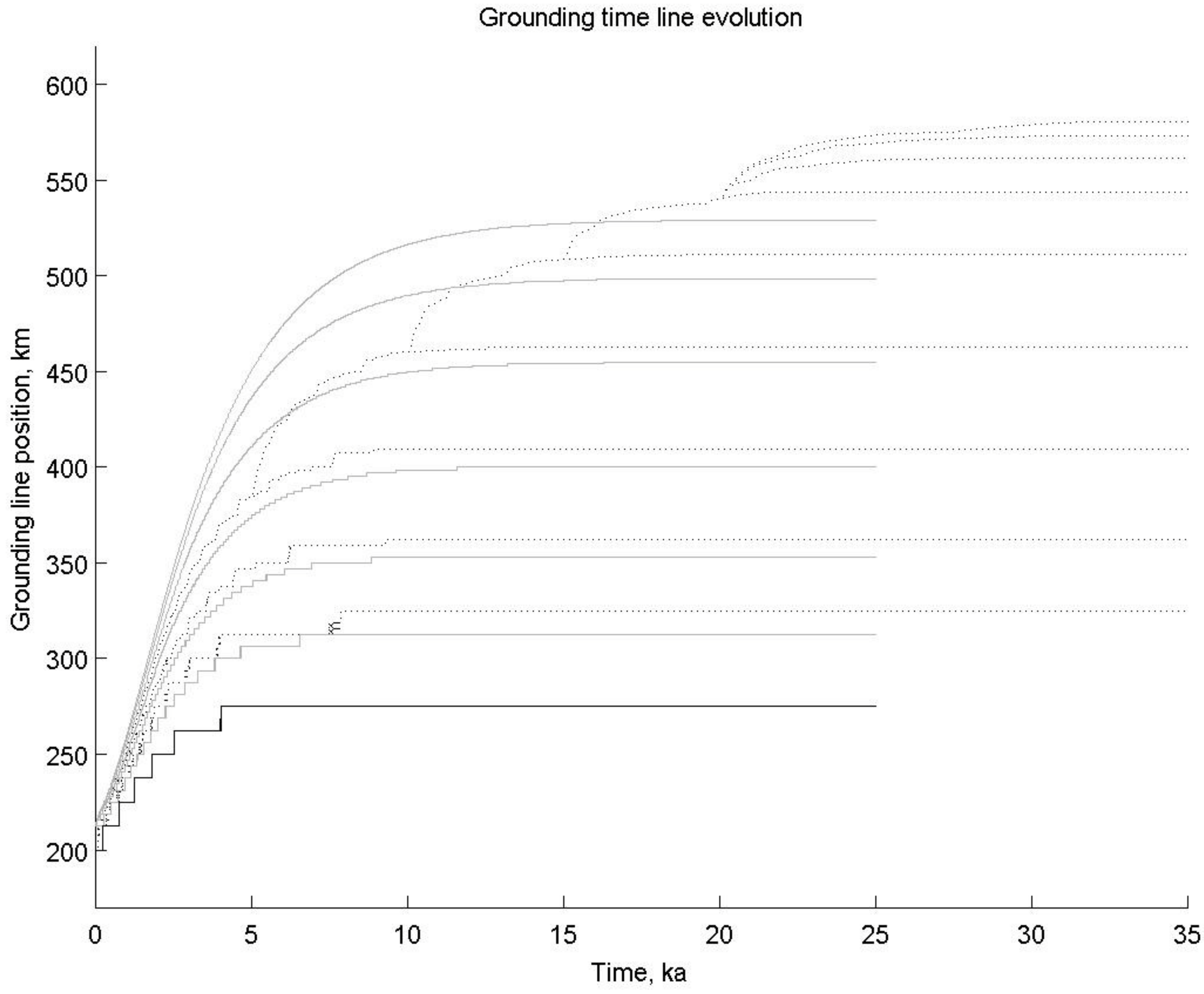
Steady state grounding line position with progressively doubling resolution



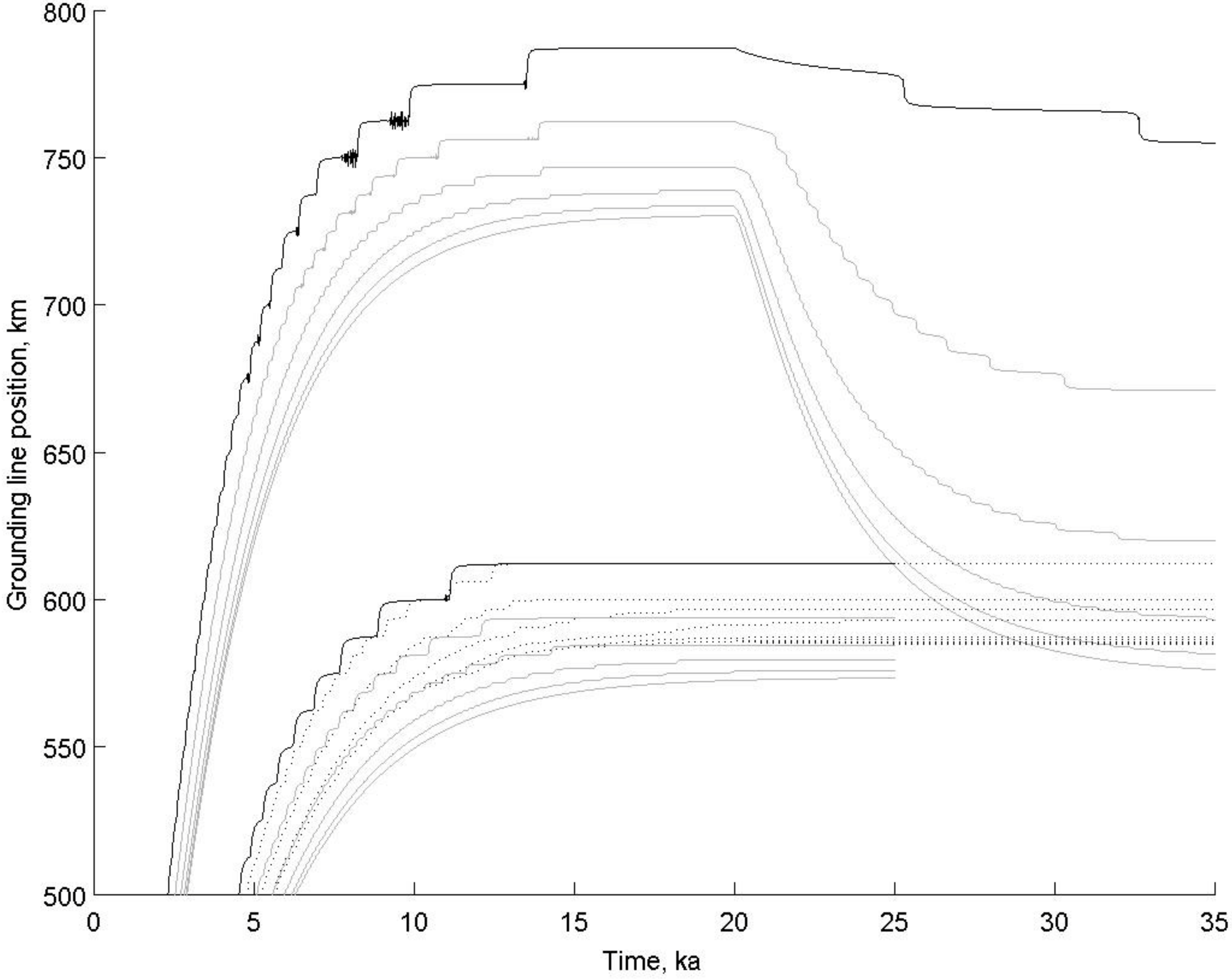
Conclusions

- ‘Conventional’ fixed grid models are very poor at predicting grounding line motion (unstable, resolution dependant and wrong at all feasible resolutions).
- Mesh refinement shows improvement (with grounding line position feedback through the refinement hierarchy).
- Grounding line position parameterisation shows improvement.
- Both adaptivity and the parameterisation together MIGHT be sufficient to realistically simulate grounding line migration in a full 2/3D model (should at least be able to avoid resolution dependence and have good steady state accuracy).

Grounding line evolution: advance experiments with fixed grid and mesh refinement models



Grounding line evolution: advance and retreat experiments using g.l. parameterisation



Governing equations ref***vieliPayne

2.1.3. Ice Stream

[16] For an ice stream, we modify equation (4) by adding a resistance from the bed which is assumed to be linearly related to the velocity u at the bed (a viscous till is assumed). The equation determining the vertically averaged ice stream velocity u is then

$$2\frac{\partial}{\partial x}h\nu\frac{\partial u}{\partial x} - \beta^2 u = \rho_i g h\frac{\partial s}{\partial x}, \qquad (6)$$

where β^2 is a friction coefficient to be specified. Equation (6) indicates that the driving stress on the right-hand side is balanced by basal traction and longitudinal stress gradients alone. Equations (4) and (6) both ignore lateral resistance to flow, which is thought to be particularly important for ice streams [*Whillans and Van der Veen*, 1997]. However, we do not regard this as important for the purposes of our study

[12] The evolution of the ice thickness of a marine ice sheet for plane flow is described by

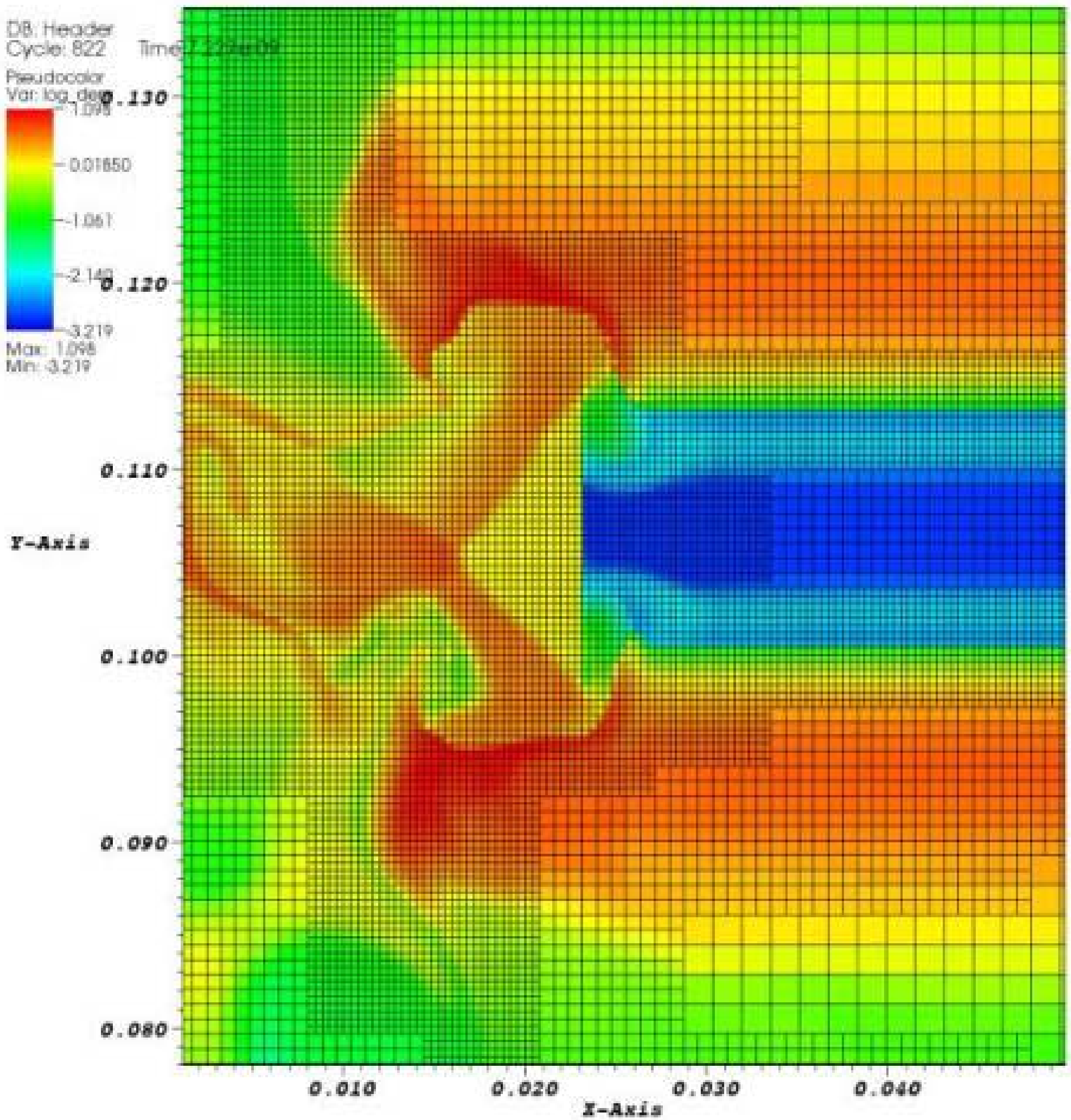
$$\frac{\partial h}{\partial t} = a - \frac{\partial q}{\partial x}, \qquad (1)$$

where t is the time, a the accumulation rate and q is the horizontal flux of ice through a vertical column of ice with thickness h and is given by $q = hu$, where u is the vertically averaged horizontal ice velocity.

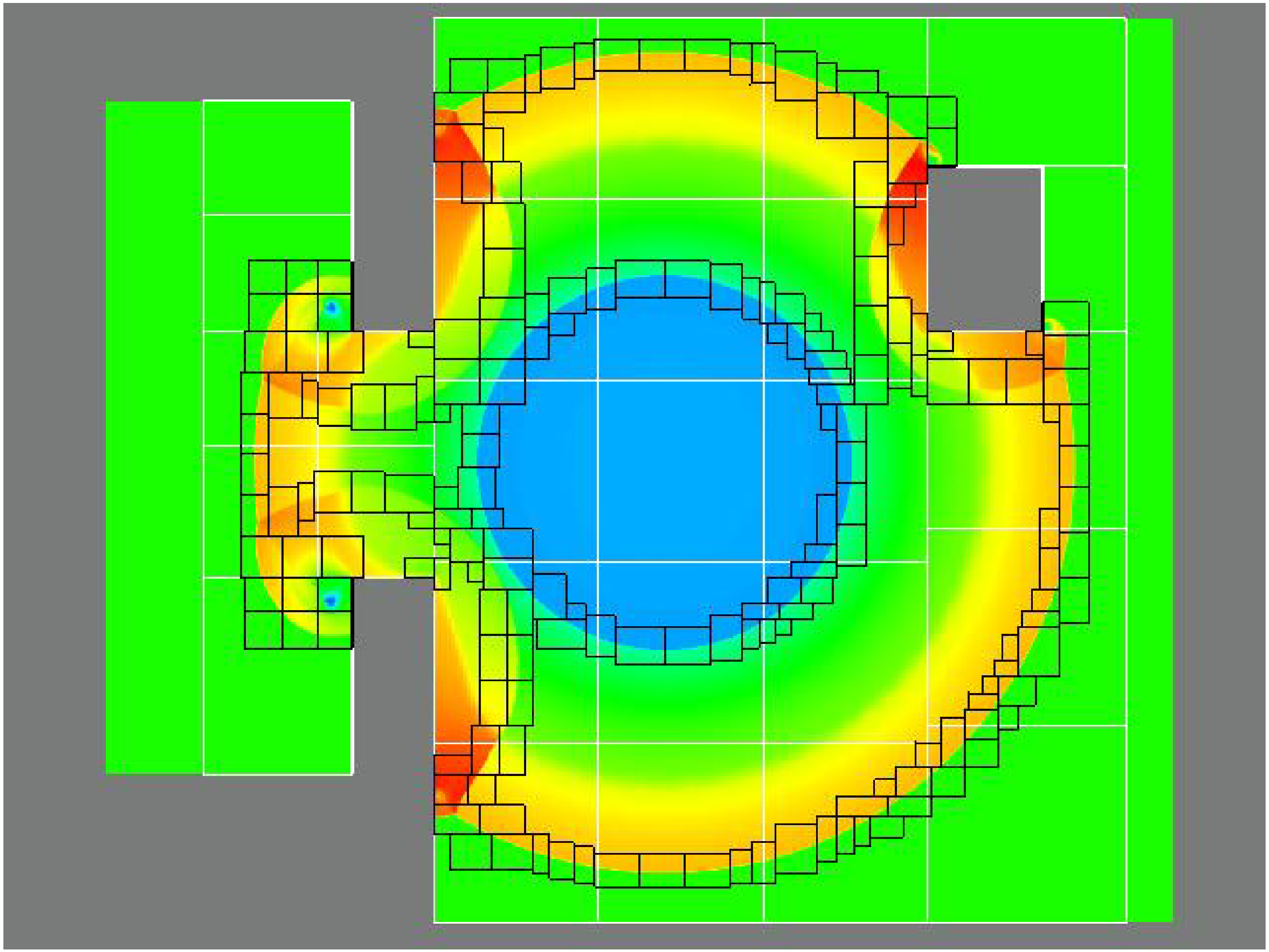
What is structured adaptive mesh refinement (SAMR)?

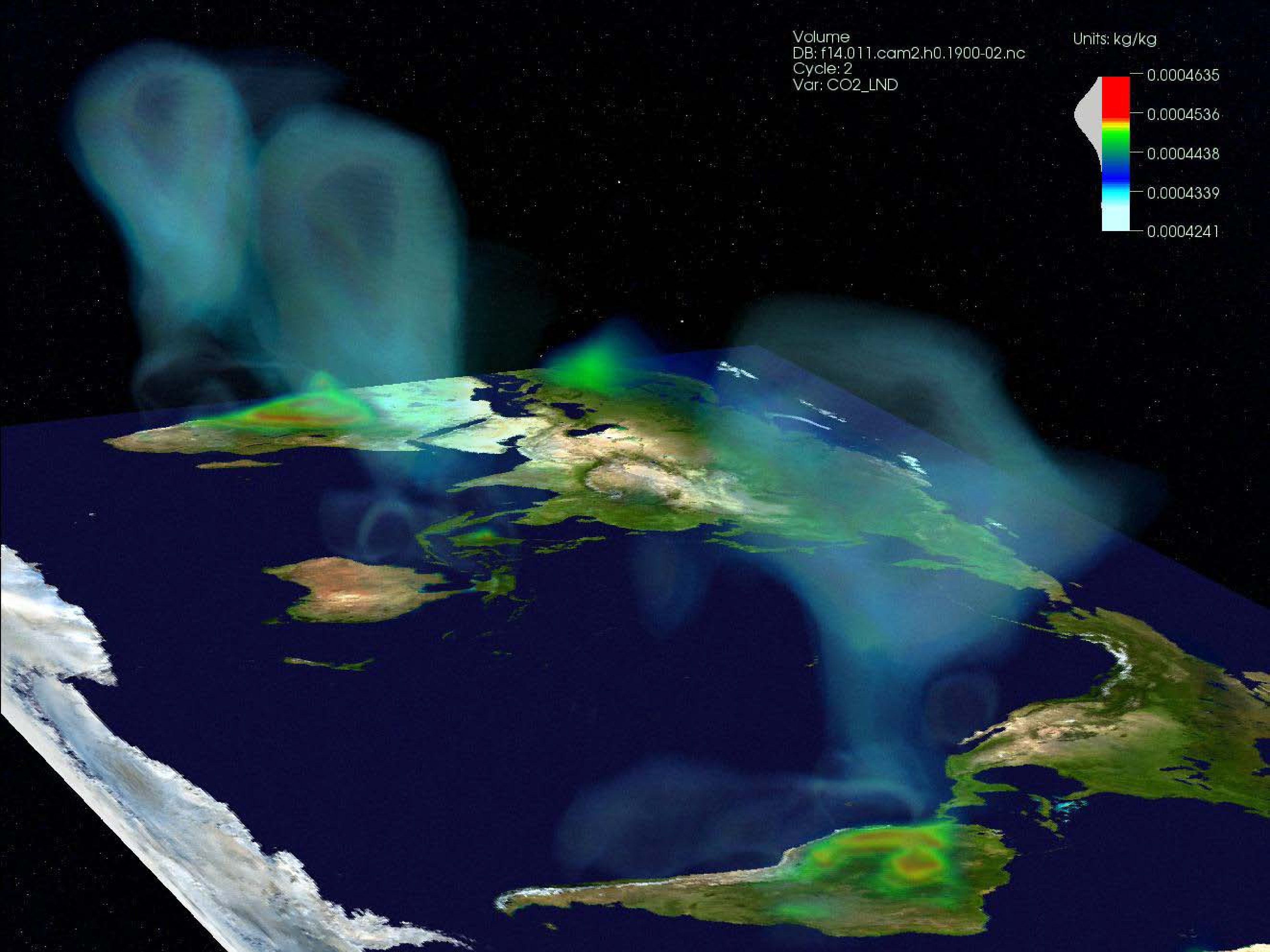
- Structured: on a rectangular grid (or ‘mesh’)
- Mesh refinement: regions of higher resolution can be introduced
- Adaptive: mesh refinement is automated (e.g. based on estimated truncation error) and occurs at runtime

3rd party packages for SAMR have been developed, e.g. ‘SAMRAI’ and ‘CHOMBO’



DB: Header
Cycle: 822
Pseudocolor
1.098
0.01850
-1.061
-2.140
-3.219
Max: 1.098
Min: -3.219
0.130
0.120
0.110
0.100
0.090
0.080
Y-Axis
0.010
0.020
0.030
0.040
X-Axis

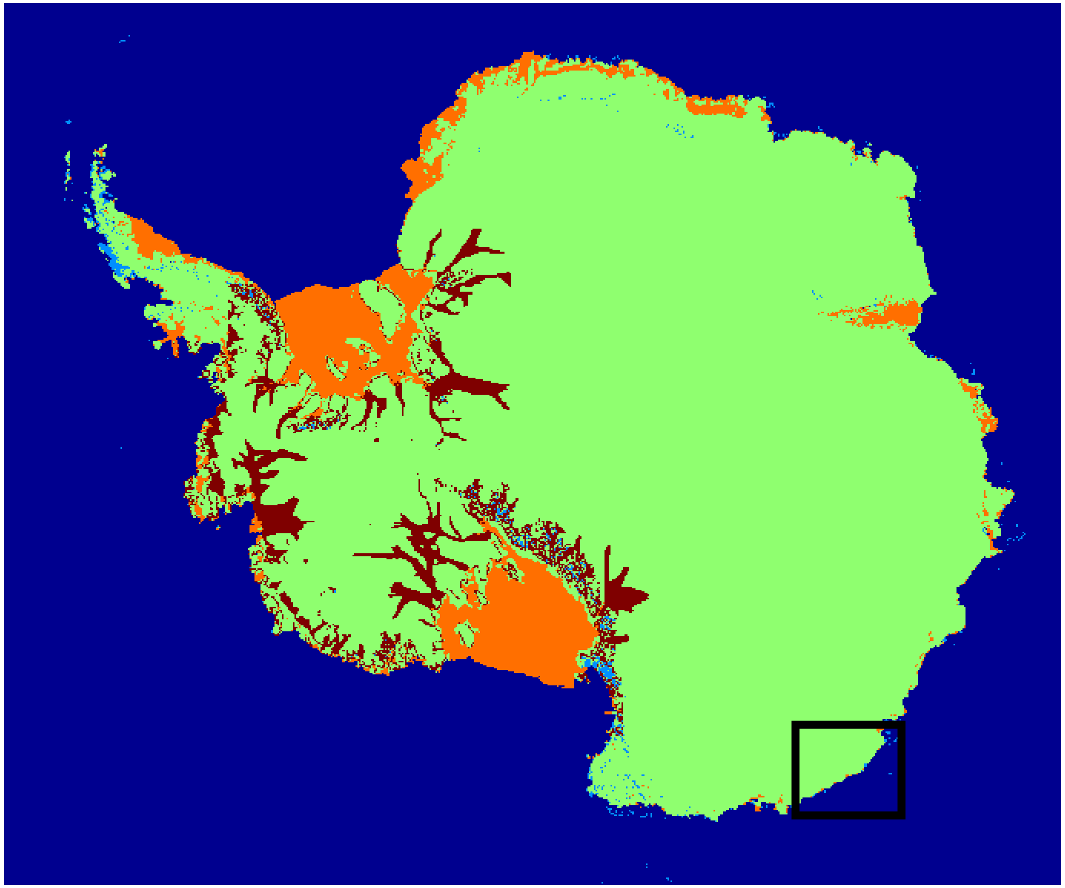



Volume
DB: f14.011.cam2.h0.1900-02.nc
Cycle: 2
Var: CO2_LND
Units: kg/kg
0.0004635
0.0004536
0.0004438
0.0004339
0.0004241

Why use SAMR?

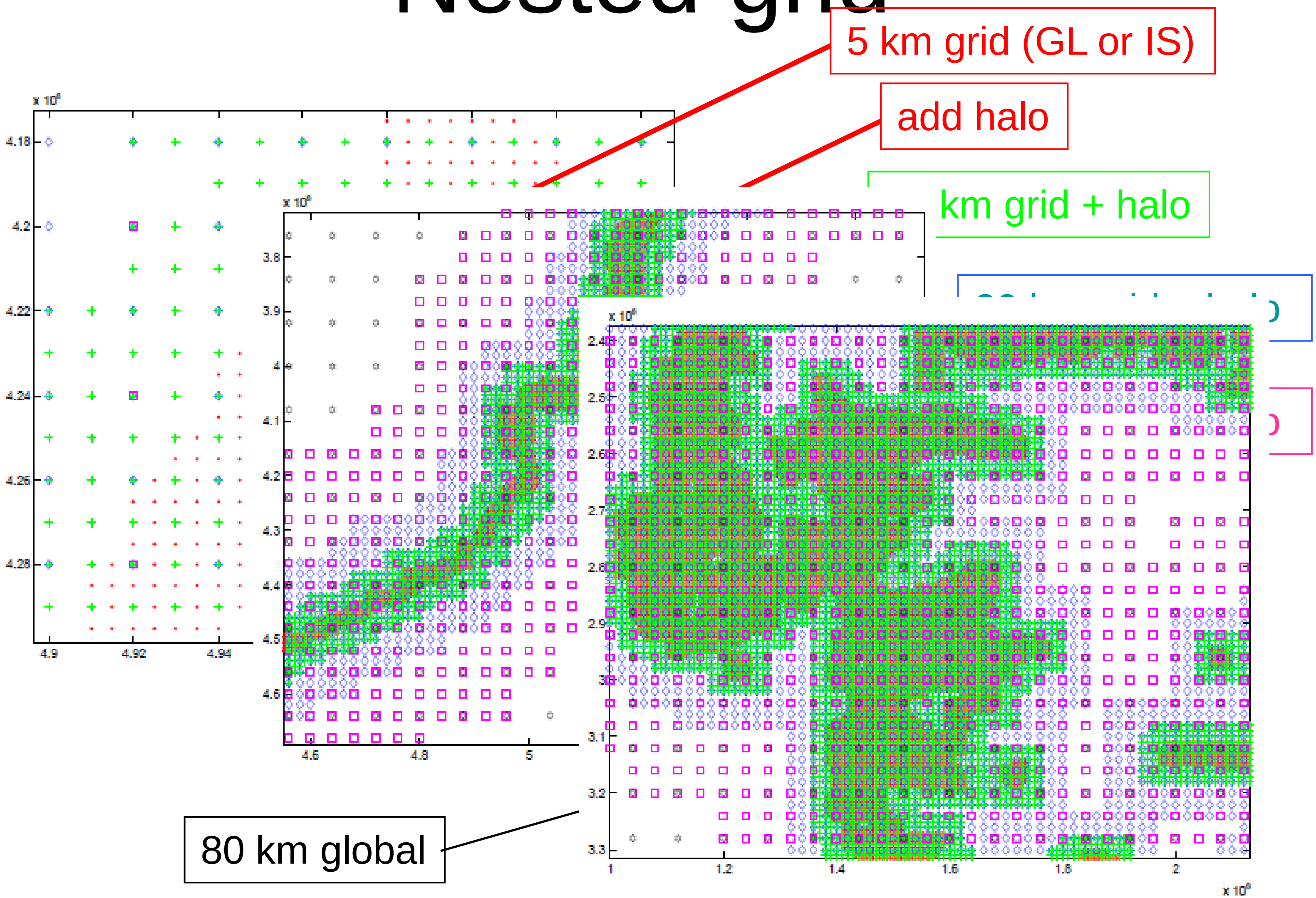
Need high resolution to resolve:

- Ice streams
- Grounding line dynamics



- 5 km BEDMAP dataset
- WAIS ice streams
- grounding line

Nested grid



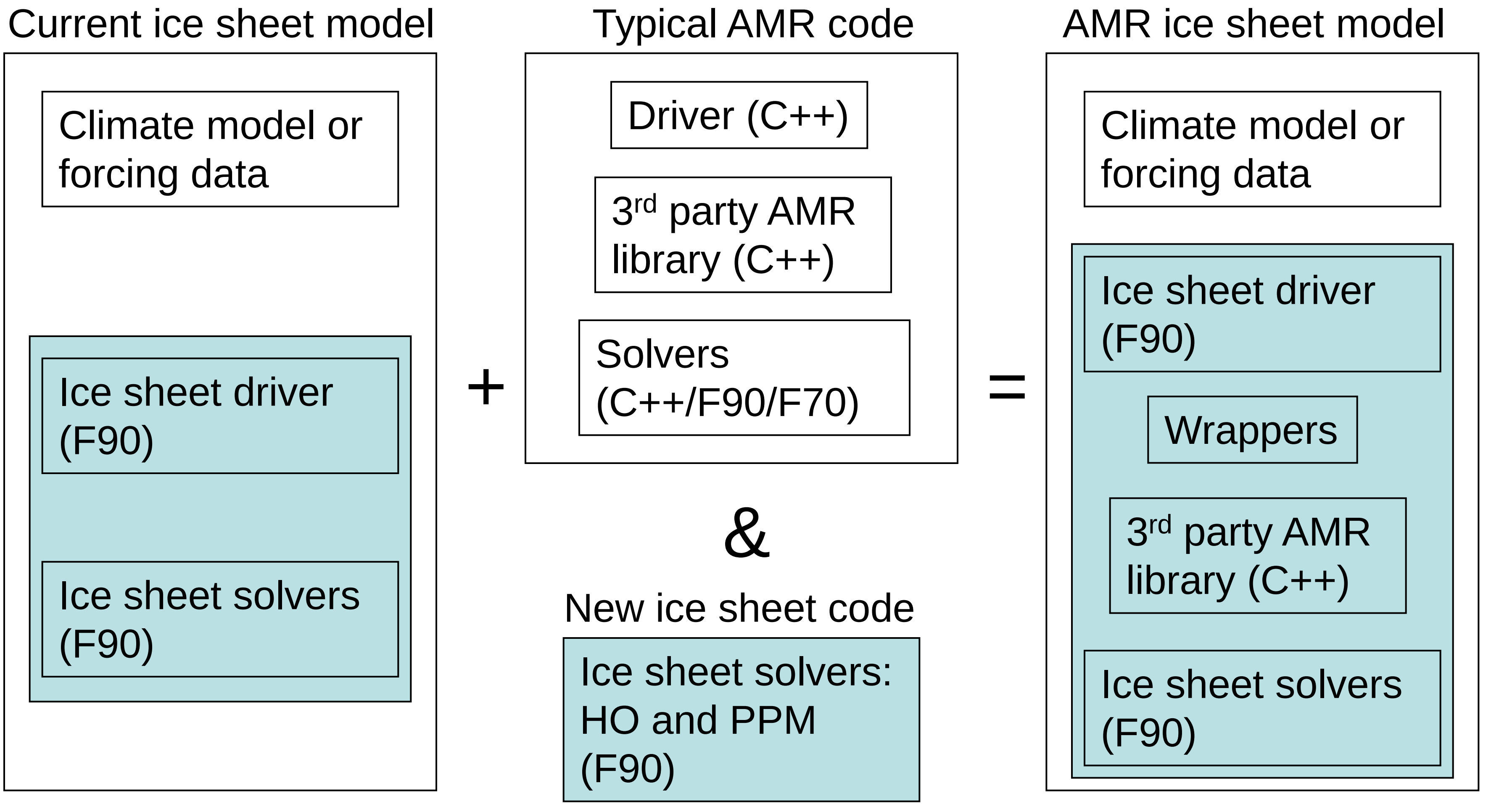
Our plan

The aim is to create a SAMR ice sheet model and apply it to the Antarctic ice sheet, in particular the WAIS.

Ingredients:

- GLIMMER – current ice sheet model!
- Higher order velocity solver!
- PPM thickness solver!
- SAMR!

Code structure

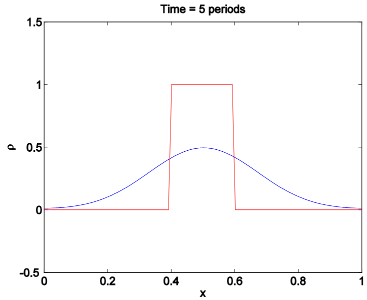


Piecewise-parabolic method (PPM)

- developed to solve shock problems in gas dynamics and applied to problems in astrophysics and hydrodynamics
- explicit upwinding scheme for solving advection equations
- uses parabolic interpolation functions which
  - lie within limits of neighbouring data to stop over and under shooting
  - steepen where gradient is high
  - use a monotonicity constraint to stop oscillations
- 3^{rd} order accurate in space (4^{th} order for uniform grid)
- 2^{nd} order accurate in time

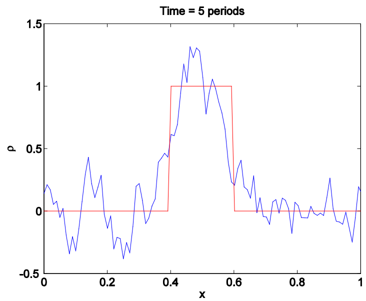
Need for PPM scheme

Comparing numerical solution (blue) with exact solution (red) for simple advection of a square wave with uniform velocity after 5 periods



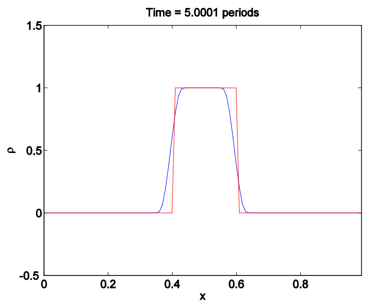
1[st] order upwinding scheme with explicit Euler time stepping

Advantage: smooth solution
Disadvantage: strong numerical diffusion



2[nd] order central difference scheme with leapfrog time stepping

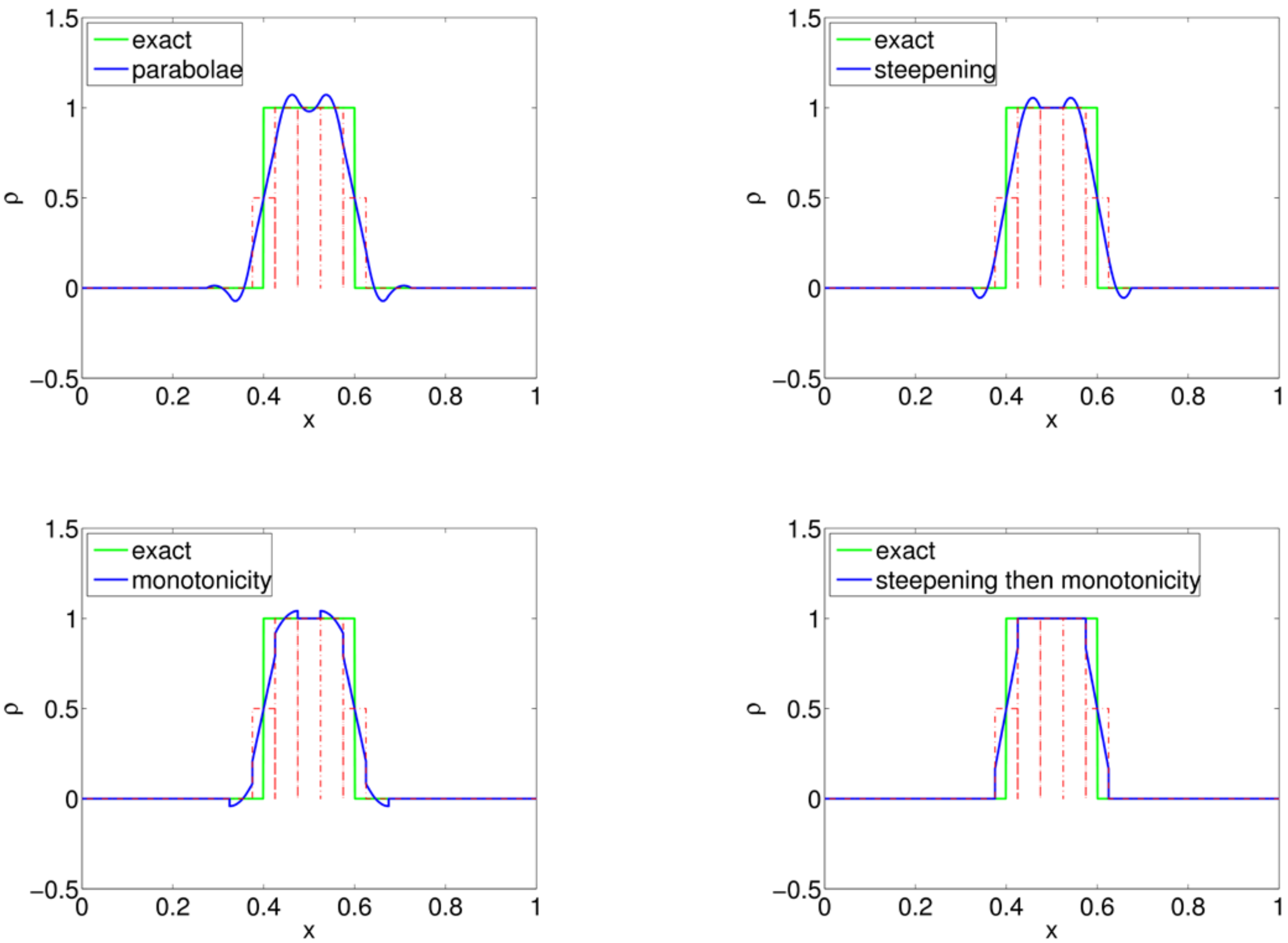
Advantage: amplitude of wave maintained
Disadvantage: oscillations in solution.



PPM scheme

Advantages: little numerical diffusion
smooth solution

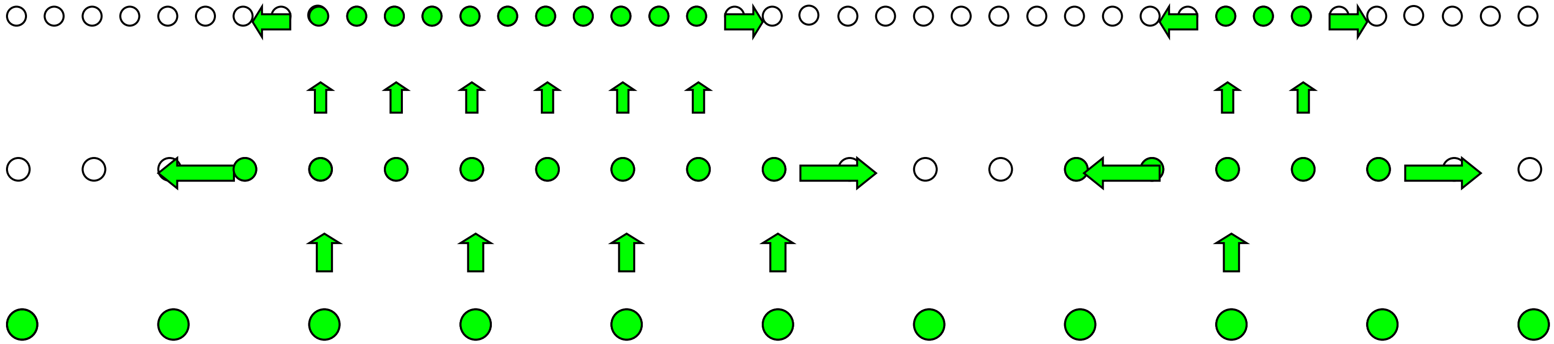
Piecewise-parabolic interpolation



Work so far

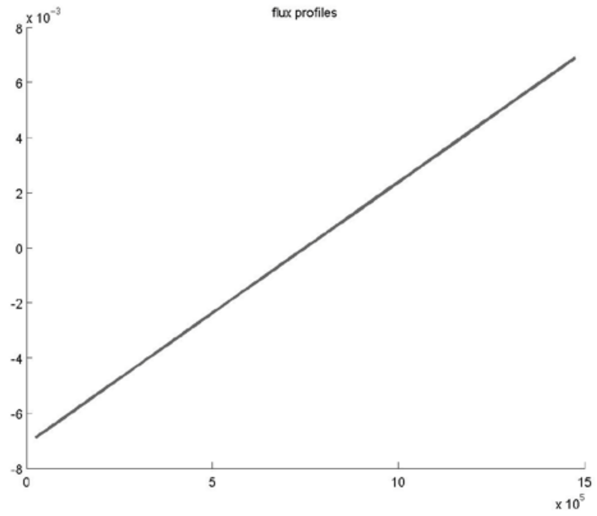
- Independent 1D AMR ice sheet model developed at Bristol, initially using diffusion based thickness solver.
- Simple AMR ice sheet simulations carried out.
- PPM incorporated.
- Currently carrying out 1D AMR ice sheet/shelf simulations focussing on grounding line migration.

Grid refinement example

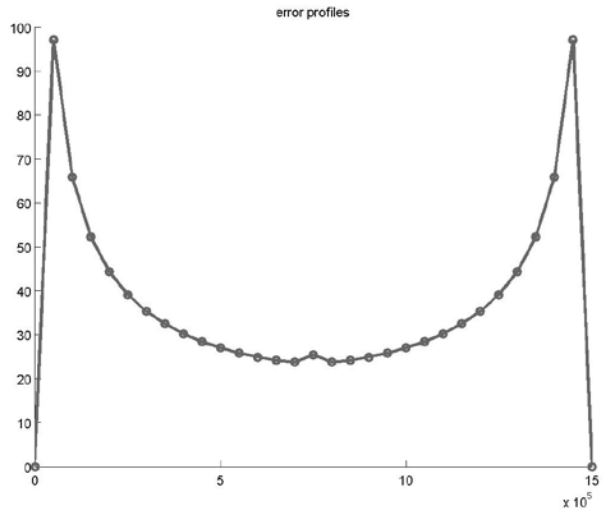


Thickness

Flux



Error

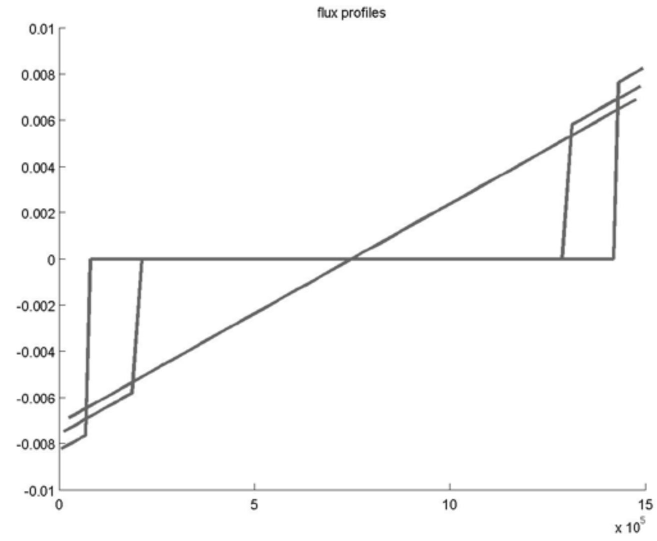


Simple experimental setup

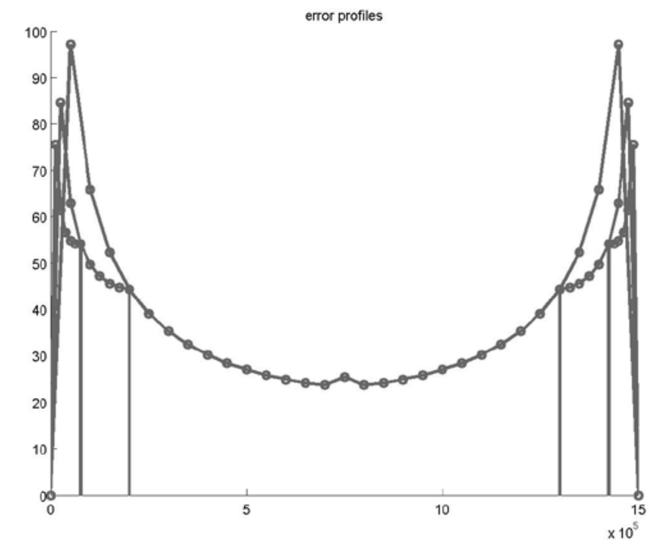
- Constant uniform SMB
- 1500km domain
- 50km resolution
- Zero thickness b.c.
- Spin up to equilibrium

Thickness

Flux

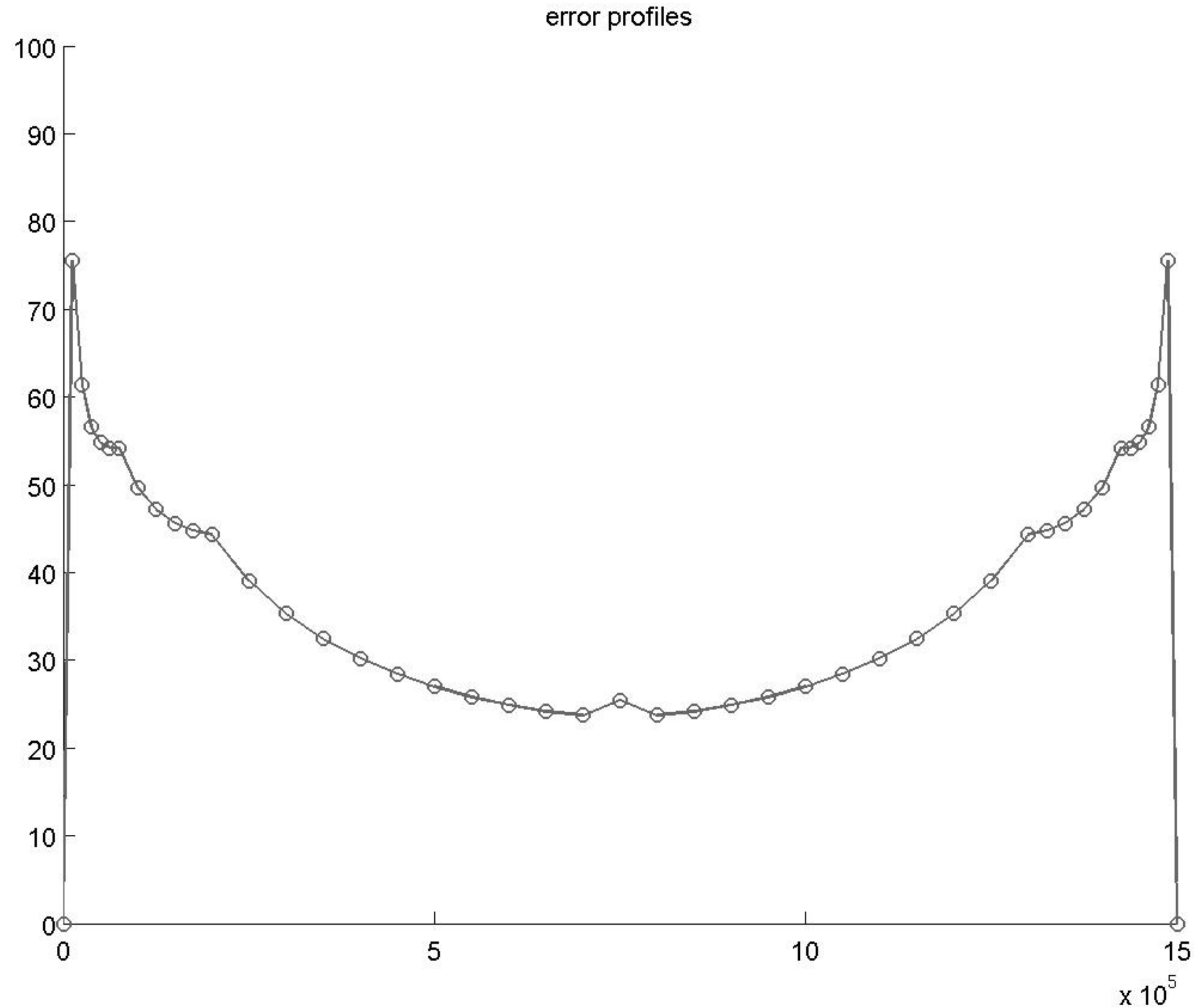


Error



As above but:

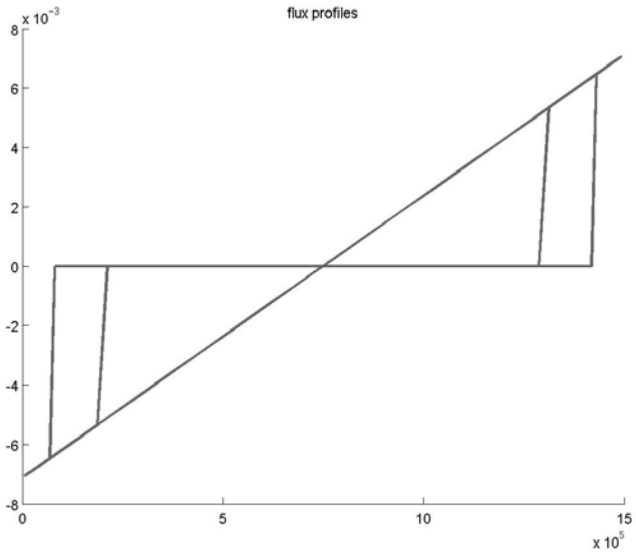
- with 3 level AMR
- refinement factor 2
- Richardson based truncation error estimate used for mesh refinement
- thickness internal boundary conditions



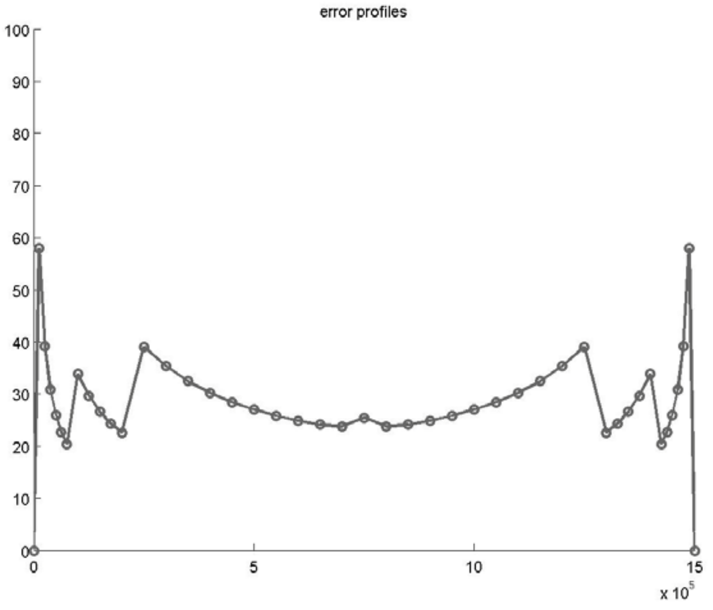
error profiles
100
90
80
70
60
50
40
30
20
10
0
0
5
10
15
x 10^5

Thickness

Flux



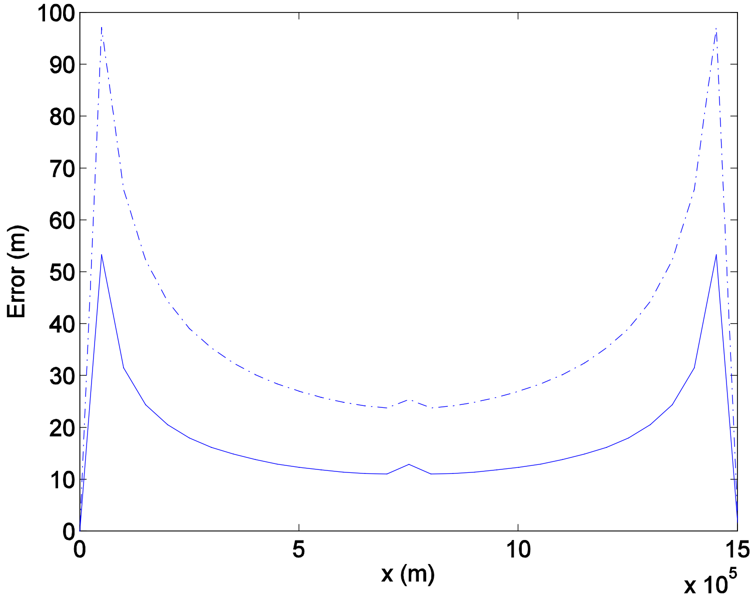
Error



As previous but:

- flux internal boundary condition

Note errors reduced and improved fluxes



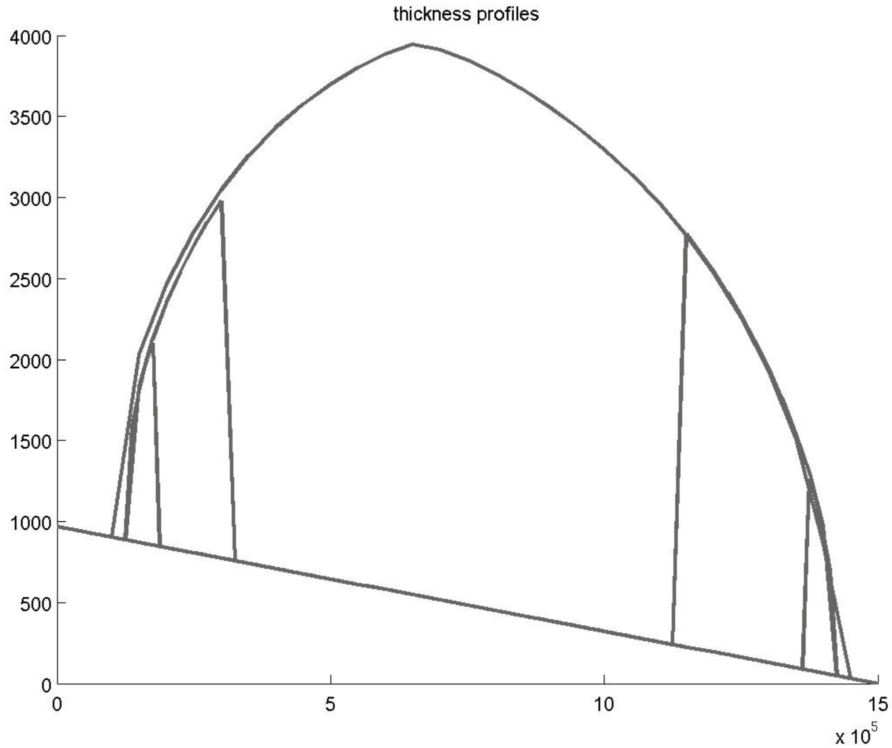
Comparing numerical solutions of 1d ice sheet with vertically integrated velocity with Glen's flow law exponent of 3.

Error = numerical solution – exact solution

Resolution is 50 km

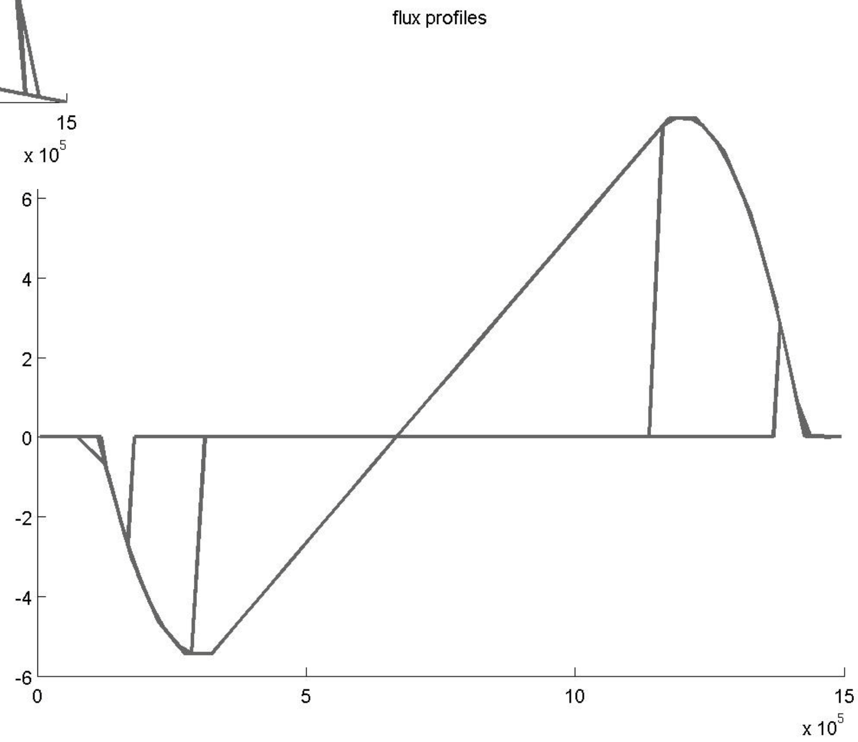
Diffusive form of the continuity equation with 2^{nd} order spatially accurate discretisation and implicit time stepping (dash dot line)

Advective formulation solved using PPM (solid line). Velocity is discretized using 2^{nd} order central difference scheme.



Surface height

Flux



As previous but:

- non-flat bedrock
- non-uniform SMB (negative near edge of domain)

Governing equations for the ice stream/shelf model

2.1.3. Ice Stream

[16] For an ice stream, we modify equation (4) by adding a resistance from the bed which is assumed to be linearly related to the velocity u at the bed (a viscous till is assumed). The equation determining the vertically averaged ice stream velocity u is then

$$2\frac{\partial}{\partial x}h\nu\frac{\partial u}{\partial x} - \beta^2 u = \rho_i g h\frac{\partial s}{\partial x}, \qquad (6)$$

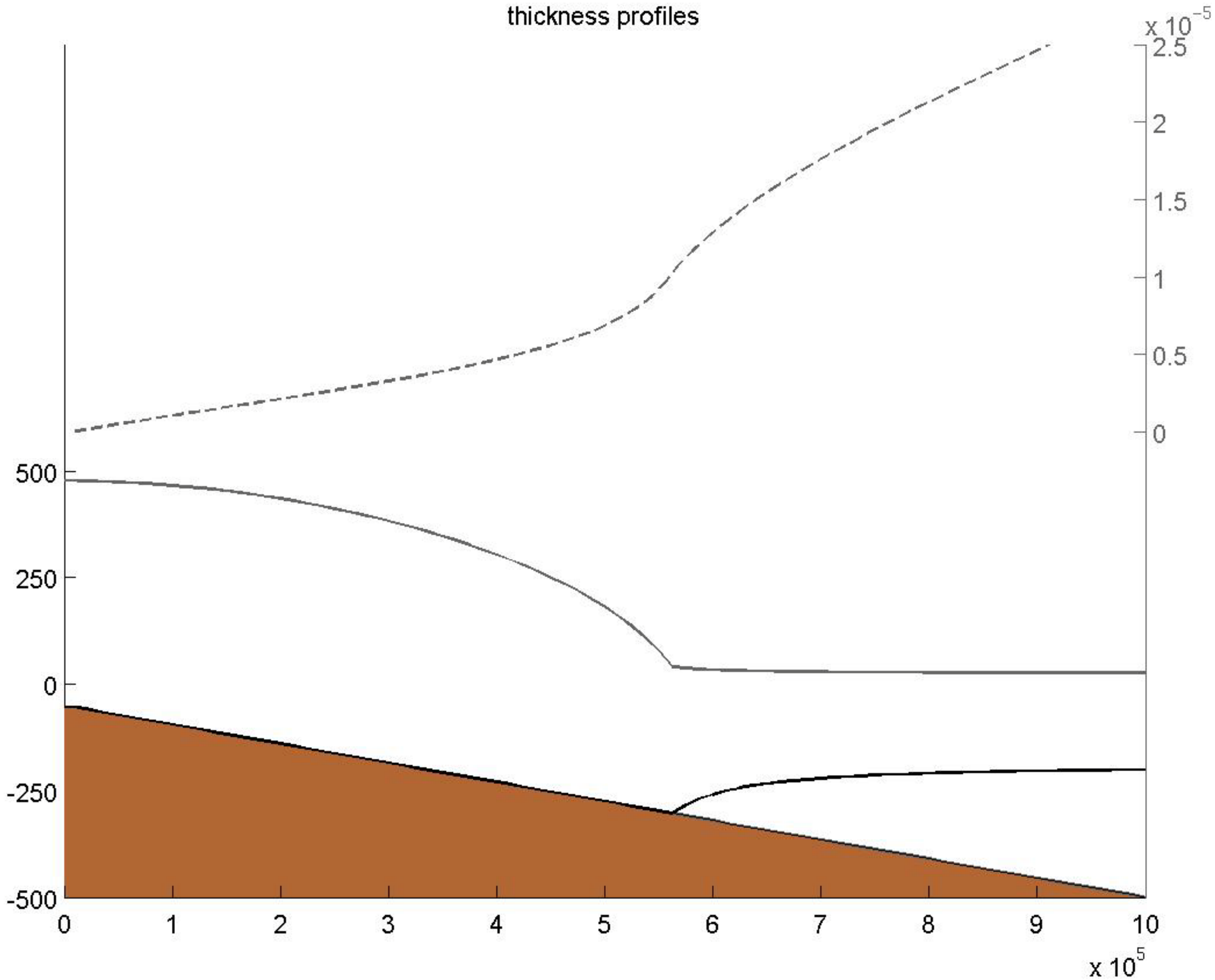
where β^2 is a friction coefficient to be specified. Equation (6) indicates that the driving stress on the right-hand side is balanced by basal traction and longitudinal stress gradients alone. Equations (4) and (6) both ignore lateral resistance to flow, which is thought to be particularly important for ice streams [*Whillans and Van der Veen*, 1997]. However, we do not regard this as important for the purposes of our study

[12] The evolution of the ice thickness of a marine ice sheet for plane flow is described by

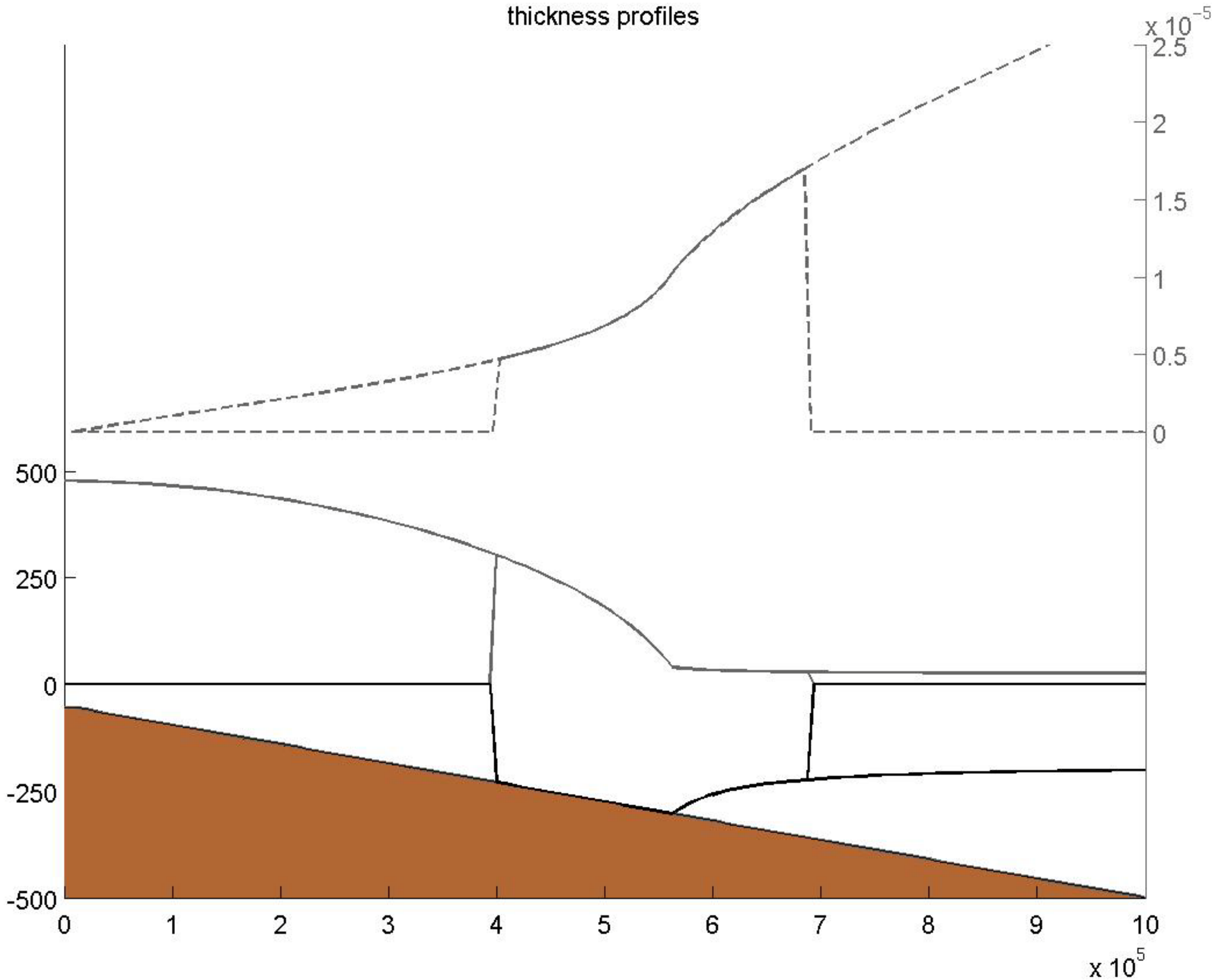
$$\frac{\partial h}{\partial t} = a - \frac{\partial q}{\partial x}, \qquad (1)$$

where t is the time, a the accumulation rate and q is the horizontal flux of ice through a vertical column of ice with thickness h and is given by $q = hu$, where u is the vertically averaged horizontal ice velocity.

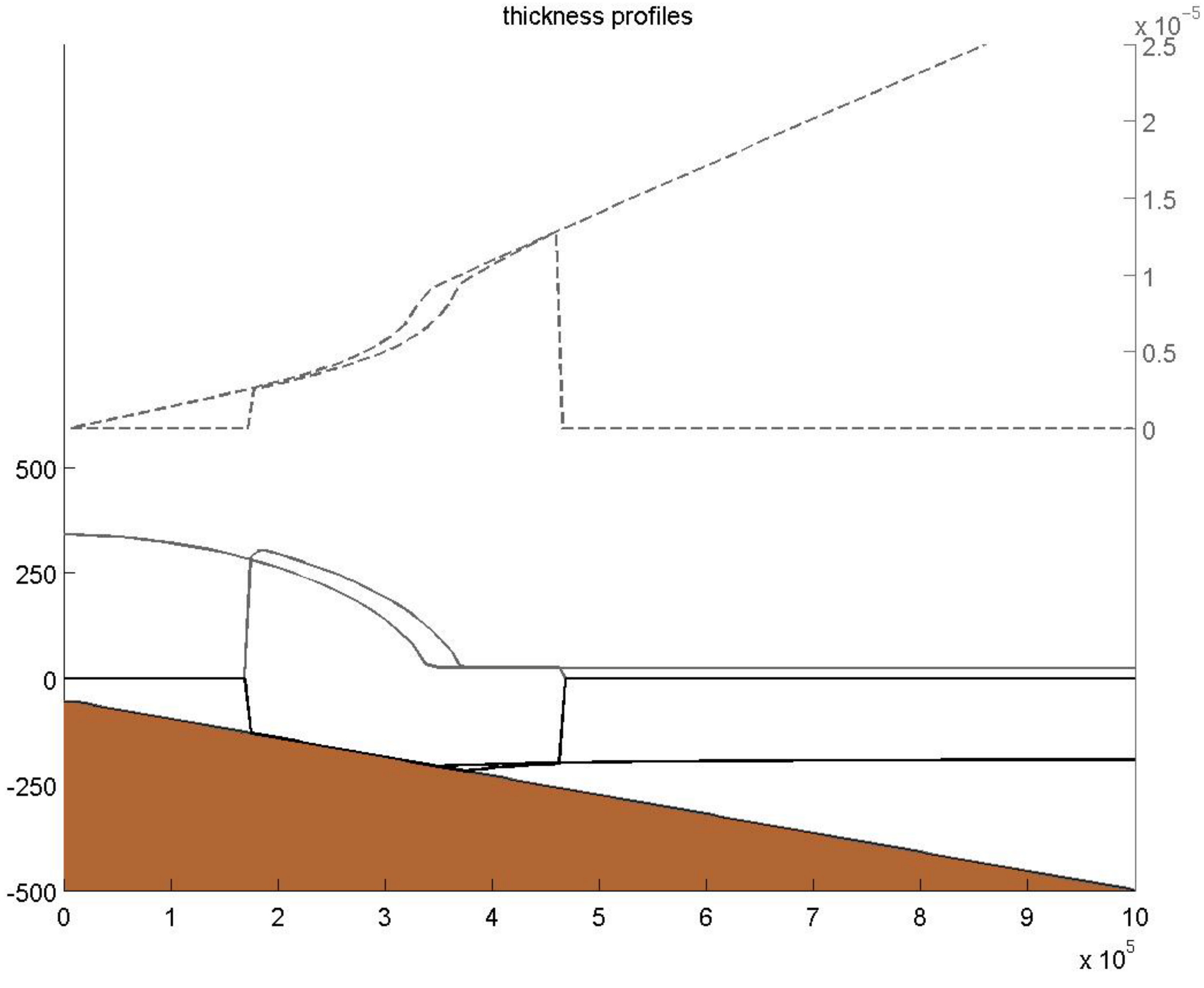
Example steady state for ice stream/shelf model



Example steady state for ice stream/shelf model with AMR (2 levels)

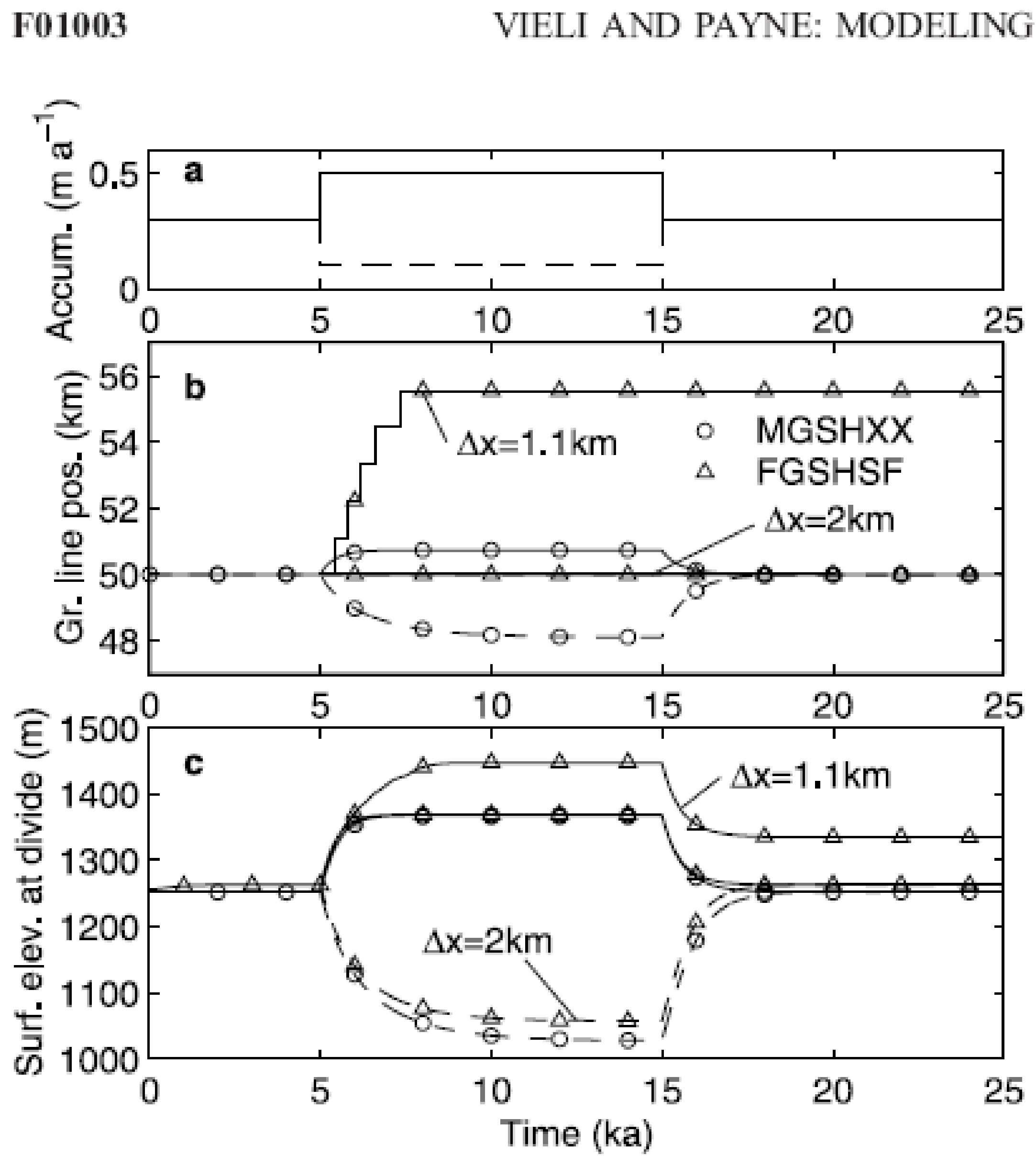


Another example steady state for ice stream/shelf model with AMR (2 levels)

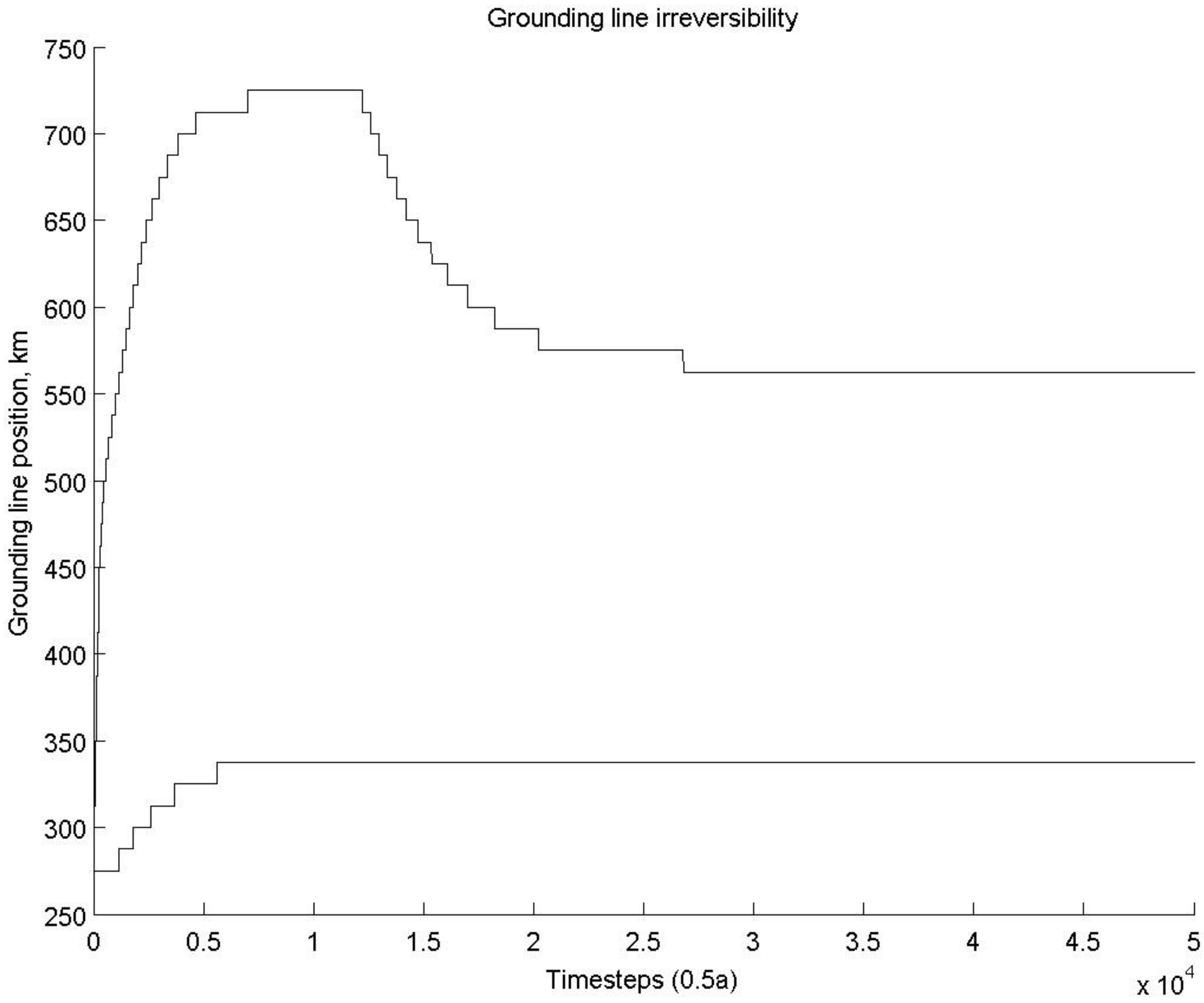


Irreversibility of grounding line migration in fixed grid models

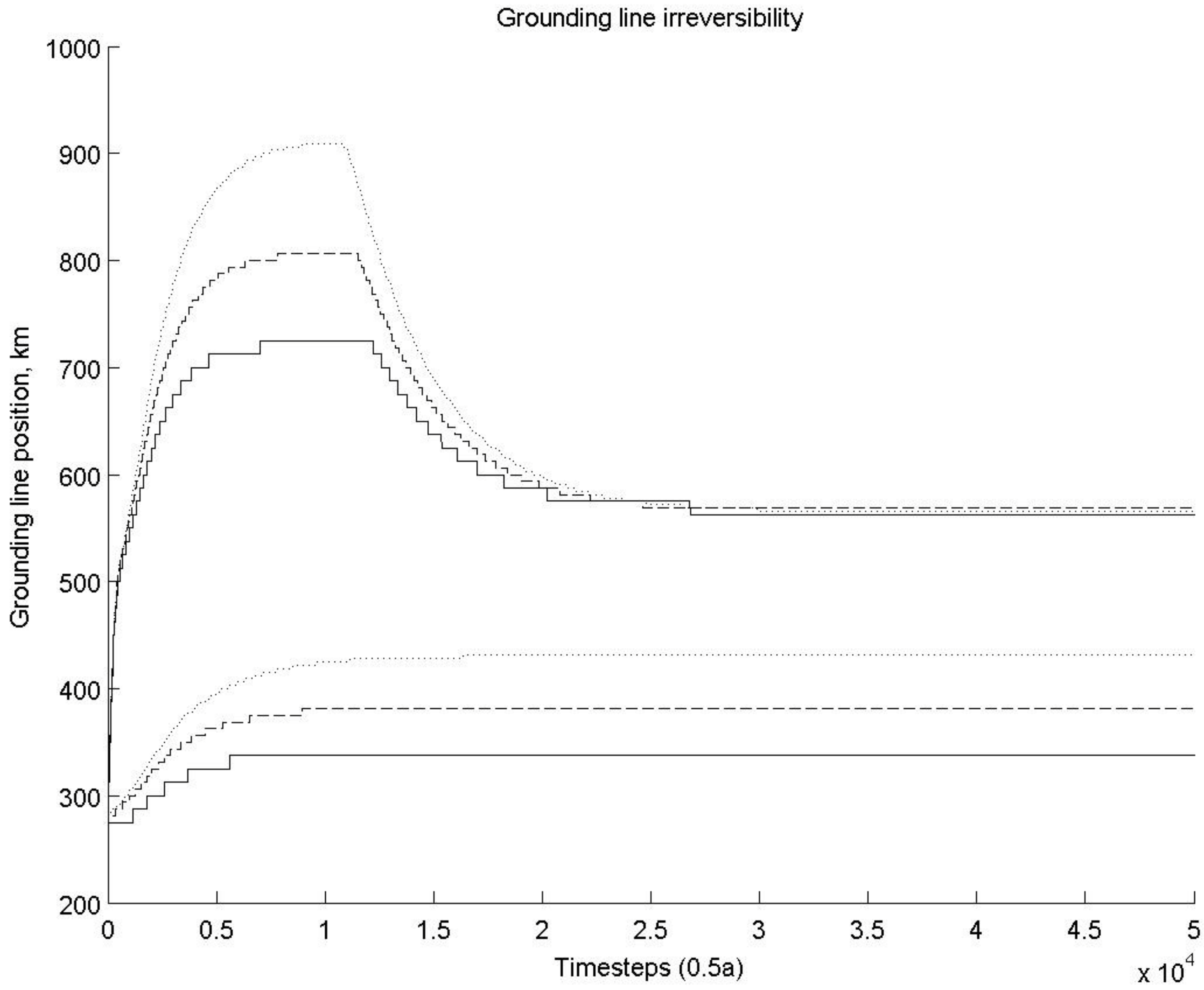
Irreversibility => multiple valid steady states



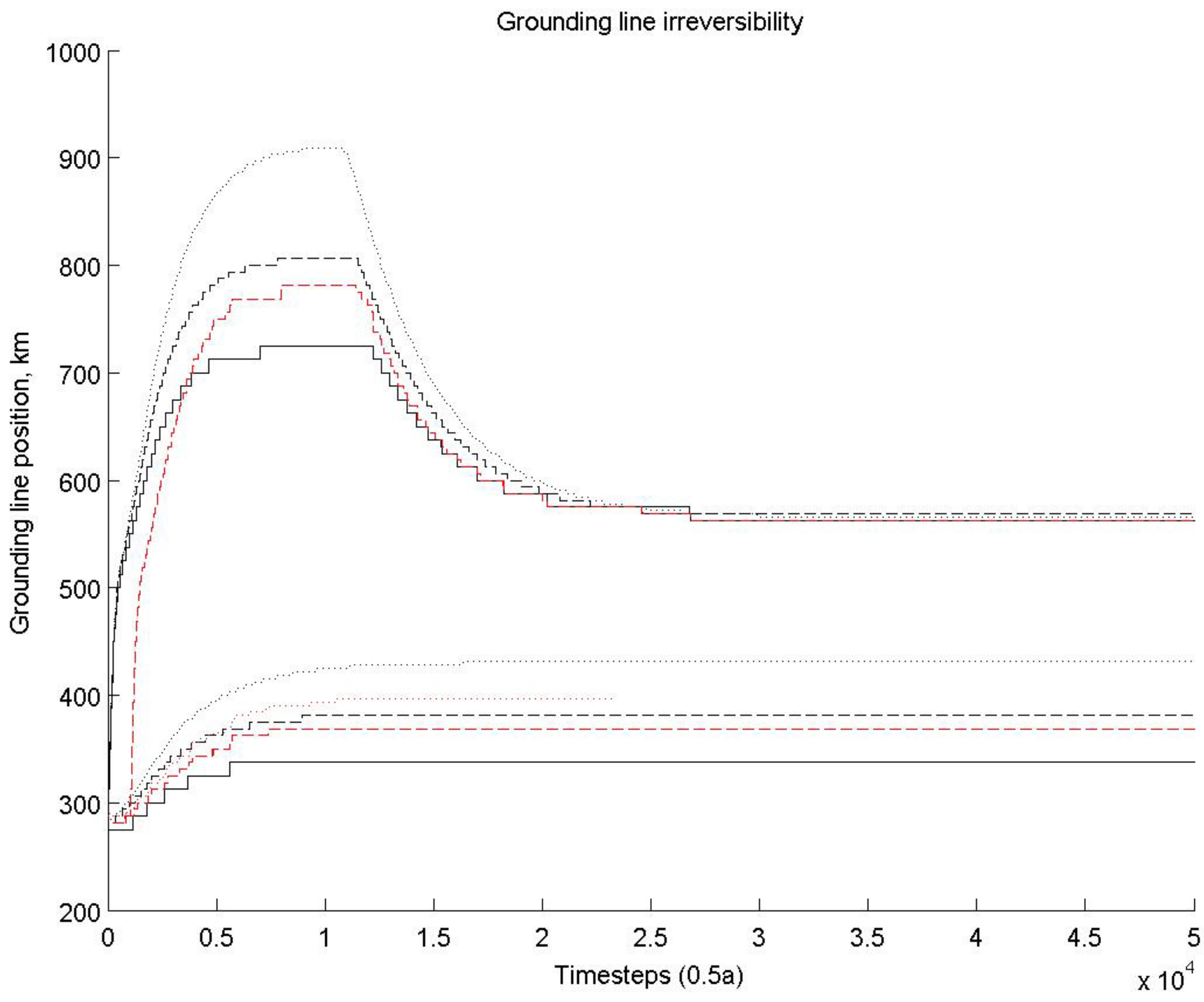
‘Irreversibility’ experiments



Increasing resolution: 12.5km (solid), 6.25km (dashed), 3.125km (dotted)

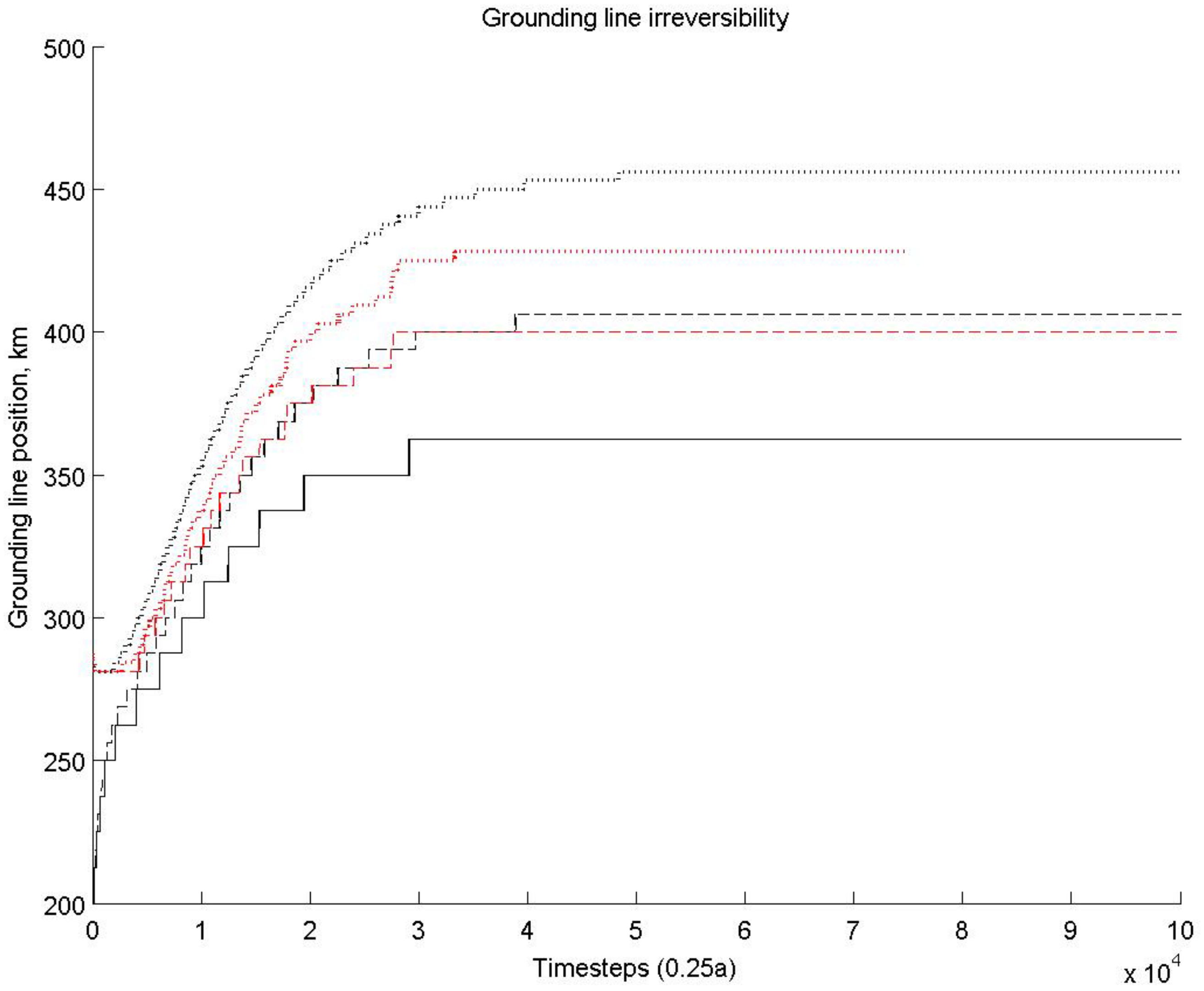


Adding AMR (red)

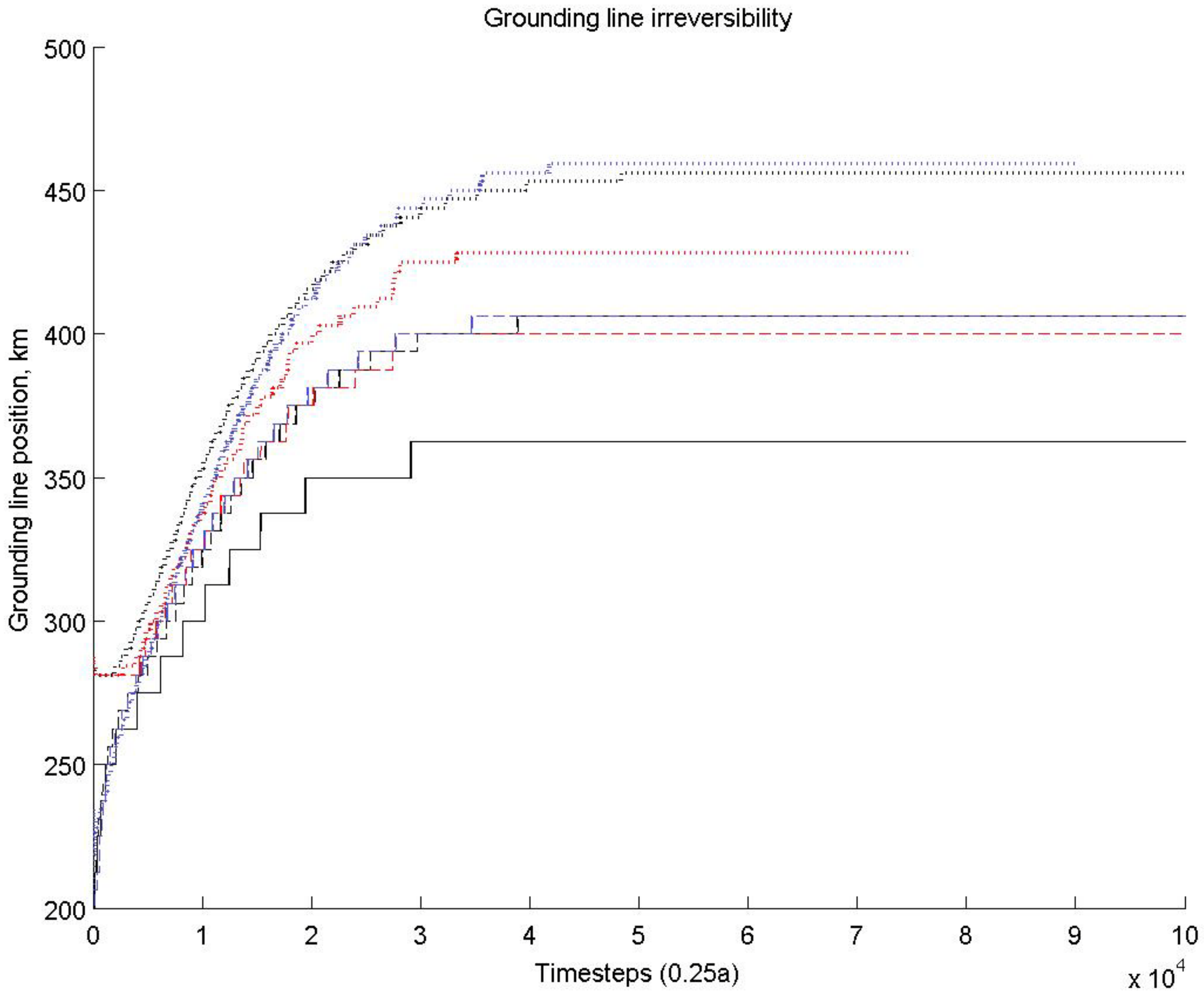


Using the ‘best’ estimate for grounding line position

More of the same kind of thing (slightly different setup)



Adding grounding line position adjustment from high to low refinement levels



(provisional) Conclusions

Regarding a suitable metric for model ability to represent grounding line migration:

- It must be recognised that irreversibility experiments are potentially sensitive to starting state.
- ‘Size of region of locally stable steady states’ (more catchy name please?) is a more informative metric.

The main result:

- AMR does improve ability to model grounding line migration.

```
% contains configuration info for running the amism1d model

&config

! amr stuff
cfg%initlevels          = 1,
cfg%maxlevels           = 1,
 cfg%ref_fact           = 2,
 cfg%refine_freq        = 20,
 cfg%refine_criterion   = 25,   !*
 cfg%refine_cutoff      = 0.25,

! which solver and types of boundary conditions
 cfg%solver             = 55,   !*
 cfg%ppm_relgridsize    = 92,   !*

 cfg%bc_thick_left_internal  = 2,   !*
 cfg%bc_thick_right_internal = 1,   !*
 cfg%bc_vel_left_internal    = 8,   !*
 cfg%bc_vel_right_internal   = 8,   !*
 cfg%bc_thick_left           = 1,   !*
 cfg%bc_thick_right          = 1,   !*
 cfg%bc_vel_left             = 11,  !*
 cfg%bc_vel_right            = 9,   !*

! restart information
 cfg%st_timestep        = 1,
 cfg%thick_file         = 'none',
```

```
! how to handle grounding line
 cfg%gl_mode            = 1,    !*
 cfg%gl_xdist_init      = 750.e3,

! basic grid setup
 cfg%totpts             = 81,
 cfg%dx                 = 12500.,
 cfg%trun               = 15.0e3,
 cfg%dt                 = 0.5,

! physical 'constants' that might very from run to run...
 cfg%rate_factor        = 0.5e-17,
 cfg%slip_coeff         = 1.0e9,


! changing smb forcing
 cfg%initial_acab       = 0.2,
 cfg%acab_step_changes  = .TRUE.,
 cfg%acab_value_ch1     = 0.2,
 cfg%acab_value_ch2     = 0.2,
 cfg%ts_acab_step_change_1  = 5000,
 cfg%ts_acab_step_change_2  = 10000/

!* see ami_global.f90 for what these numbers mean
```

Types of equilibrium

```
Well, I have an old animation I made for demonstrating stochastic
resonance, which is probably what you're thinking of.  That was a bit
specific to that application though, with randomly forced motion in an
oscillating double well potential.  You're probably better off just
drawing some pictures.  Here -- I'll help:

  \     /              /O\
   \   /   ___O___    /   \
    \O/              /     \

   STABLE  NEUTRAL   UNSTABLE

Done!  (Assuming you read your email in a monospaced font.  Which
everyone who isn't a complete heathen obviously does.)
```

Diagram courtesy of Ian Ross, department for sophisticated visualisation techniques, Bristol University

- &config

- ! cfg%output_dir = './plotting',
- cfg%output_dir = '.',

- ! amr stuff
- cfg%initlevels = 1,
- cfg%maxlevels = 1,
- cfg%ref_fact = 2,
- cfg%refine_freq = 20,
- cfg%refine_criterion = 25, !*
- cfg%refine_cutoff = 0.25,

- ! which solver and types of boundary conditions
- cfg%solver = 55, !*
- cfg%ppm_relgridsize = 92, !*

- cfg%bc_thick_left_internal = 2, !*
- cfg%bc_thick_right_internal = 1, !*
- cfg%bc_vel_left_internal = 8, !*
- cfg%bc_vel_right_internal = 8, !*
- cfg%bc_thick_left = 1, !*
- cfg%bc_thick_right = 1, !*
- cfg%bc_vel_left = 11, !*
- cfg%bc_vel_right = 9, !*

- ! steady state instability fudge
- cfg%fudge = .FALSE.,

- ! restart information
- cfg%st_timestep = 1,
- cfg%thick_file = 'none',
- ! cfg%thick_file = 'thickness_restart',
- cfg%init_thickness = 200.

- ! how to handle grounding line
- cfg%gl_mode = 1, !*
- cfg%gl_xdist_init = 750.e3,

- ! basic grid setup
- cfg%totpts = 321,
- cfg%dx = 3125.,
- cfg%trun = 25.0e3,
- cfg%dt = 1.,

- ! physical 'constants' that might vary from run to run...

- ! standard values:
- cfg%rate_factor = 9.2e-18, Pa^-3 a^-1
- cfg%rate_fac_units = 110, !*
- cfg%slip_coeff = .5e10,
- cfg%rho_ice = 910.,
- cfg%rho_water = 1028.,

- ! mismip values:
- ! cfg%rate_factor = 2.1544e-24,
- ! cfg%rate_fac_units = 111, !*
- ! cfg%slip_coeff = 7.2082e9, ! factor 10 lower than mismip - more like ice stream
- ! cfg%rho_ice = 900.,
- ! cfg%rho_water = 1000.,

- ! changing smb forcing
- cfg%initial_acab = .3,
- cfg%acab_step_changes = .FALSE.,
- cfg%acab_value_ch1 = 0.3,
- cfg%acab_value_ch2 = 0.3,
- cfg%ts_acab_step_change_1 = 20000,
- cfg%ts_acab_step_change_2 = 30000/

- !* see ami_global.f90 for what these numbers mean

- ! type of boundary condition to use (for master domain and for nested grids)
- ! see ppm code for description of boundary conditions (for example see
- ! end of ppm_routines.f90)
- integer,parameter :: PPM_BC_0 = 0 ! not tested
- integer,parameter :: PPM_BC_1 = 1 ! outflow b.c. (ghost cells don't impact)
- integer,parameter :: PPM_BC_2 = 2 ! inflow b.c.
- integer,parameter :: PPM_BC_3 = 3 ! not tested
- integer,parameter :: PPM_BC_4 = 4 ! not tested
- integer,parameter :: PPM_BC_5 = 5 ! not tested (flux)
- integer,parameter :: THICK_BC = 6 ! fixed thickness b.c. for diffusion based solver
- integer,parameter :: FLUX_BC = 7 ! flux b.c. for diffusion based solver
- integer,parameter :: HO_BC_DIRICHLET = 8 ! fixed uvel b.c. for the HO solver
- integer,parameter :: HO_BC_VONNEUMANN = 9 ! strain rate, based on force balance, for HO solver
- integer,parameter :: NO_BC = 10 ! boundary condition not defined (e.g. no uvel b.c. defined when using diffusion based solver)
- integer,parameter :: HO_BC_DIRICHLET_ZERO=11!
- integer,parameter :: HO_BC_DIRICHLET_CONS=12! fixed uvel b.c. for the HO solver. uvel is calculated based on flux conservation.

- ! grounding line modes (implemented through betasq)
- integer,parameter :: GL_NORMAL = 1 ! each level evolves the gl independantly
- integer,parameter :: GL_FIXED = 2 ! gl position is prescribed at start of run and kept fixed
- integer,parameter :: GL_AMR = 3 ! allow the highest refinement level to determine a gl position to be used by all levels
- integer,parameter :: GL_LINTERP = 4 ! each level evolves the gl independantly, using linewar interpolation to locate exact position of groundling line
- integer,parameter :: GL_AMR_LI = 5 ! allow the highest refinement level to determine a gl position to be used by all levels, using linear interpolation to locate exact pos\
- ition of grounding line

- ! error measure for grid refinement
- integer,parameter :: DIFFBASED = 21
- integer,parameter :: THICKNORM = 22
- integer,parameter :: DIFFBASED_ANAL = 23
- integer,parameter :: THICKNORM_ANAL = 24
- integer,parameter :: GL_PROX = 25 ! grounding line proximity
-
- ! what units are used to define rate_factor in the config file
- integer,parameter :: PERPACUBED_PERS = 111 ! Pa^-3 s^-1
- integer,parameter :: PERPACUBED_PERA = 110 ! Pa^-3 a^-1

- ! error measure for grid refinement
- integer,parameter :: DIFFBASED = 21
- integer,parameter :: THICKNORM = 22
- integer,parameter :: DIFFBASED_ANAL = 23
- integer,parameter :: THICKNORM_ANAL = 24
- integer,parameter :: GL_PROX = 25 ! grounding line proximity

- ! which velocity and thickness solver(s) to use
- integer,parameter :: DIFFLINEAR = 50 ! the old diffusion solver

- integer,parameter :: DIFFNONLIN = 51 ! " (nonlinear)

- integer,parameter :: PPME_LOVEL = 52 ! piecewise parabolic calulcation of surface evolution, eulerian mode
- ! zero order velocity

- integer,parameter :: PPME_ANALVEL = 53 ! ppm surface evolution, eulerian mode
- ! analytic solution for velocity

- integer,parameter :: PPM_NONLIN = 54 ! defunct

- integer,parameter :: PPME_HOVEL = 55 ! ppm surface evolution, eulerian mode
- ! higher order stress solver to calculate velocity

- integer,parameter :: PPML_LOVEL = 56 ! ppm surface evolution, lagrangian mode
- ! zero order velocity

- integer,parameter :: PPML_ANALVEL = 57 ! ppm surface evolution, lagrangian mode
- ! analytic solution for velocity

- integer,parameter :: PPML_HOVEL = 58 ! ppm surface evolution, lagrangian mode
- ! higher order stress solver to calculate velocity

- integer,parameter :: UPWN_HOVEL = 59 ! simple explicit upwind scheme for thickness
- ! higher order velocity