Tutorial IV
Multi-material, self-consistent subduction
with a free surface\textsuperscript{1}

Anne Glerum

\textsuperscript{1}Schmeling et al., PEPI, 2008
Succeeded in:

• Setting up a model with one compositional heterogeneity
• Using ASPECT’s function parser
• Setting up mesh-independent initial conditions
• Tackling benchmark problems
By the end of this tutorial, you should be able to

• Write and install new material plugins

• Modify the input parameter file for a subduction model with multiple materials

• Understand issues regarding averaging

• Understand the concept of “sticky-air” and its effect on the solver
Simple subduction

Start simple:
Subduction model of Schmeling et al. 2008 (PEPI 171)
• 2D
• No temperature effects
• Constant viscosities
• Benchmark → results of other codes to compare
Schmeling et al. 2008 subduction.
Although a relatively simple setup, it discusses very important points:

- Effect of different **averaging** methods on viscosities near rheological boundaries
- **Decoupling** subducting plate from the surface
- Approximation of **free surface** through sticky-air
Sticky-air

- Thin layer of relatively low viscosity ($10^{19}$ Pas) and density (0 kg/m$^3$) to allow for surface deformation
- No need to deform grid, but
- High viscosity contrasts and
- High resolution needed
Tasks

Changes compared to Tutorial III:
• Prescribe parameters of multiple (>2) materials
• Implement 4 different types of averaging of materials

→

1. Modify schmeling_empty.prm
2. Write a new Material Model based on assigned averaging method
3. Run simulation and visualize results
4. Report slap tip depth after 1 and 2 My
We will begin by editing the input file

1. Change to the appropriate directory
   ```
   > cd ~/ASPECT_TUTORIAL/models
   ```

2. Open the parameter file for editing
   ```
   > gedit schmeling_empty.prm
   ```
Now read through the following sections in the input file and edit the red sections:

1. Global parameters
2. Geometry model
3. Compositional fields
4. Material model
5. Compositional initial conditions
Editing the input file

Global parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimension</td>
<td>2</td>
</tr>
<tr>
<td>Start time</td>
<td>0</td>
</tr>
<tr>
<td>End time</td>
<td>0</td>
</tr>
<tr>
<td>Use years in output instead of seconds</td>
<td>true</td>
</tr>
<tr>
<td>Number of cheap solver steps</td>
<td>0</td>
</tr>
<tr>
<td>Output directory</td>
<td>schmeling</td>
</tr>
</tbody>
</table>

Geometry model

subsection Geometry model
  set Model name = box
  subsection Box
    set X extent = 2000000
    set Y extent = 750000
    set X repetitions = 3
  end
end
Editing the input file

Compositional fields

subsection Compositional fields
  set Number of fields = end
subsection Compositional initial conditions
  set Model name = function
subsection Function
  set Variable names = x, z
  set Function constants = \[ \begin{align*}
  Ax &= 1000000.0, \\
  Az &= 700000.0, \\
  Bz &= 500000.0, \\
  Cx &= 1100000.0, \\
  Dz &= 600000.0
\end{align*} \]
  set Function expression = \\
  \begin{align*}
  &\text{if}( (z<Bz) | (x<Ax & z<Az) | (x>Cx & z<Dz), 1, 0); \ #mantle} \\
  &\text{if}( (x>=Ax & z>=Dz & z<Az) | (x>=Ax & x<=Cx & z>=Bz & z<Dz), 1, 0); \ #lithosphere} \\
  &\text{if}(z>=Az, 1, 0) \ # air
\end{align*} \\
end
end

How many compositional fields? For simplicity, we describe all materials as fields

You already know how to use the function parser!
Editing the input file

3 Compositional fields with different densities and viscosities
Material model

subsection Material model
  set Model name = XXX
subsection XXX model
  set Thermal conductivity = 0.0
  set Thermal expansion coefficient = 0.0
  set Reference specific heat = 1.0
  set List of densities of fields =
  set List of viscosities of fields =
end
end

Normally, here you provide a functional name for your new Material Model plugin.

No temperature effects

Here we provide the densities and viscosities of the compositions to the Material Model plugin. Use the same order as in the definition of the compositional fields.
Material Model plugins

1. Change to the appropriate directory
   > cd ~/ASPECT_TUTORIAL/aspect/source/material_model

2. What files are there?
   interface.cc
   simple.cc
   simpler.cc
   steinberger.cc
   table.cc ...

Material Model plugins
Plugin organization

Plugins:
• for Geometry, Material, Gravity etc. in ~/source
• derive from interface.cc
• can be selected from the input file

A Material Model plugin should at least provide
1. Viscosity
2. Density
3. Specific heat
4. Thermal conduct.
5. Thermal expansion
6. Compressibility
So far, we used Material Model simple.cc

> gedit simple.cc

template <int dim>
double
Simple<dim>::
viscosity (const double temperature,
    const double pressure,
    const std::vector<double> &composition,
    const SymmetricTensor<2,dim> &strain_rate,
    const Point<dim> &position) const
The Material Model plugin – viscosity function

Simulator
- temperature
- pressure
- composition
- strain_rate
- position

Input file
- densities_fields
- viscosities_fields

Material Model

viscosity

viscosity function
So far, we used Material Model simple.cc

```c++
const double delta_temp = temperature-reference_T;
const double temperature_dependence = (reference_T > 0
    ? std::max(std::min(std::exp(-thermal_viscosity_exponent*delta_temp/reference_T),1e2),1e-2)
    : 1.0);
double composition_dependence = 1.0;
if (((composition_viscosity_prefactor != 1.0) && (composition.size() > 0))
    { return (pow(10, ((1-composition[0]) * log10(eta*temperature_dependence)
        + composition[0] * log10(eta*composition_viscosity_prefactor
            *temperature_dependence))));
    }
return composition_dependence * temperature_dependence * eta;
```
So far, we used Material Model simple.cc

```cpp
if (composition.size() > 0)
    { return (pow(10, ((1-composition[0]) * log10(eta)
                      + composition[0] * log10(eta*composition_viscosity_prefactor)));
    }
```

What kind of averaging?
Writing a viscosity function

ASPECT: build on others!
→

• XXX.cc is a slightly adapted copy of simple.cc
• Implement a viscosity function (line 35) that averages the contribution of the fields as follows:
  • Group 1: Harmonic averaging
  • Group 2: Geometric averaging
  • Group 3: Arithmetic averaging
  • Group 4: Infinite norm
Writing a viscosity function

Group 1 - Harmonic
\[ \eta_{\text{harm}} = \frac{c_1 + c_2 + c_3}{\frac{c_1}{\eta_1} + \frac{c_2}{\eta_2} + \frac{c_3}{\eta_3}} \]

Group 2 - Geometric
\[ \log \eta_{\text{geom}} = \frac{c_1 \log \eta_1 + c_2 \log \eta_2 + c_3 \log \eta_3}{c_1 + c_2 + c_3} \]

Group 3 - Arithmetic
\[ \eta_{\text{arith}} = \frac{c_1 \eta_1 + c_2 \eta_2 + c_3 \eta_3}{c_1 + c_2 + c_3} \]

Group 4 - Infinite norm
\[ \eta_{\text{inf}} = \eta_{\text{max}}(c_i) \]

Where \( c_i \) represent the values of the 3 compositional fields, and \( \eta_i \) are the viscosities of each corresponding field.

~/ASPECT_TUTORIAL/aspect/source/material_model/schmeling.cc is a working material model with each averaging method implemented in case you need it.
Advection of a field

The advection of compositional fields can result in under- and overshooting of the $c_i$ values near steep gradients →

1. Prevent oscillations
   \[
   \frac{\partial c_i}{\partial t} + u \cdot \nabla c_i - \nabla \cdot \nu_h \nabla c_i = 0
   \]

2. Deal with oscillations
   Cut off $c_i$ between 0 and 1

Fig. 1. Numerical advection of a step function over 25 Courant-Friedrichs-Lewy time steps. T0 is the initial step function and T25 is the advected step function. Two types of numerical errors are present: (1) numerical diffusion reflected in the tilting of the step; and (2) numerical dispersion resulting in the leading edge ripples. The numerical scheme employed was second-order accurate for smooth flow problems.
C++ syntax

• Always end declarations and assignments with “;”
• The first entry of a vector is accessed with “0”: e.g. composition[0]
• Calculating a minimum of two numbers with: e.g. std::min(composition[0], 1.0)
> cd ~/ASPECT_TUTORIAL/aspect/include/ \ aspect/material_model/
Here the corresponding header file XXX.h is located

> cd ~/ASPECT_TUTORIAL/aspect/debug
Normally, you would call
> cmake .
> make
to compile and install your new plugin. Build system
`cmake` will automatically detect it. Now you only need to call
> make –j2
Using ASPECT

Now run ASPECT in the terminal

1. Change to the appropriate directory
   
   > cd ~/ASPECT_TUTORIAL/models/

2. Run ASPECT with the tutorial parameter file
   
   > mpirun -np 2 aspect-debug schmeling_empty.prm

3. If correct,
   
   > cd ~/ASPECT_TUTORIAL/aspect/release
   > make -j2

4. Change model time to 2.5 My and rerun (this will take about 15 minutes, have a coffee)

5. Use ParaView to visualize slab evolution

   > paraview schmeling/solution.pvd
Subduction evolution

Report slab tip depth after 1 and 2 My and model time

<table>
<thead>
<tr>
<th></th>
<th>Harmonic</th>
<th>Geometric</th>
<th>Arithmetic</th>
<th>Infinite</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 My Slab tip depth</td>
<td>(???)</td>
<td>(???)</td>
<td>(???)</td>
<td>(???)</td>
</tr>
<tr>
<td>2 My Slab tip depth</td>
<td>(???)</td>
<td>(???)</td>
<td>(???)</td>
<td>(???)</td>
</tr>
<tr>
<td>Model time after 15 min wall time</td>
<td>(???)</td>
<td>(???)</td>
<td>(???)</td>
<td>(???)</td>
</tr>
</tbody>
</table>
Finding slab tip depth in ParaView:

- Plot isocontour $C_2 = 0.5$

1. Use grid lines to estimate, or
2. Use Spreadsheet view of isocontour with 
   *Show only selected elements* and 
   3D view *Select Points On*, or
3. Use Spreadsheet view and Python calculator, or
4. Next time, write an ASPECT postprocessor 😊
## Subduction evolution answer key

<table>
<thead>
<tr>
<th></th>
<th>Harmonic</th>
<th>Geometric</th>
<th>Arithmetic</th>
<th>Infinite</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 My Slab tip depth</td>
<td>205,892 m</td>
<td>202,983</td>
<td>202,387</td>
<td>202,291</td>
</tr>
<tr>
<td>2 My Slab tip depth</td>
<td>215,181 m</td>
<td>205,587</td>
<td>203,725</td>
<td>203,629</td>
</tr>
<tr>
<td>Model time after 15 min wall time</td>
<td>2.11903e6 yr</td>
<td>2.14136e6</td>
<td>2.20938e6</td>
<td>329,512</td>
</tr>
</tbody>
</table>

### Diagram

- **Slab tip depth [m]**
  - 1.95E+05
  - 2.00E+05
  - 2.05E+05
  - 2.10E+05
  - 2.15E+05
  - 2.20E+05

- **Model time after 15 min wall time**
  - 2.11903e6 yr
  - 2.14136e6
  - 2.20938e6
  - 329,512

- **Legend**
  - Harmonic
  - Geometric
  - Arithmetic
  - Infinite

- **GeoMod2014**

---

*September 4, 2014*
Results after 2 My

Slab detaching from surface
Air favored → slab thinned

Harmonic
Viscosity (Pas)

1e+19
1e+20
1e+21
1e+22
1e+23

Geometric

Boundary smoothed, but not shifted

Arithmetic
Slab favored → air more viscous
Slower evolution

Infinite
Sharp but step-like boundary, slow computations

September 4, 2014
So what averaging do we use?

- Averaging affects rheological boundaries

Schmeling et al. (2008):
- Harmonic \(\rightarrow\) equivalent to effective viscosity of 2 viscous elements acting in series, like channel flow with flow-parallel compositional boundary, i.e. simple shear. Results in weak effective viscosity.
- Arithmetic \(\rightarrow\) 2 viscous elements in parallel, interface-parallel pure shear. Results in stiff effective viscosity.
- Geometric norm has no physical model, intermediate viscosity.

\(\rightarrow\) Harmonic mean more appropriate for high viscosity contrasts (1e4) and flows dominated by cusp-like overriding wedges

Schmeling et al., PEPI, 2008
Subduction evolution

**Harmonic**: higher resolution, slower subduction

**Geometric**: higher resolution, faster subduction

Shaded areas from Schmeling et al., PEPI, 2008, lines obtained with ASPECT
Results harmonic averaging

Red colors indicate composition 2
Extending on subduction models

• More materials with different characteristics, i.e. overriding plate and crust
• Realistic deformation mechanisms, i.e. elasto-visco-plasticity
• Complex boundary conditions, i.e. plate velocities, free surfaces, open boundaries
• 4D modeling
Example – Quinquis et al. (in prep)

ASPECT 2014 – visco-plastic, thermo-mechanically coupled subduction, 8 materials
Example – Quinquis et al. (in prep)

ASPECT 2014 – visco-plastic, thermo-mechanically coupled subduction, 8 materials