Viscosity variations for a convective box with aspect

How to change the viscosity definition

One way to change the viscosity definition is to define a new class of the MaterialModel, in a ".cc" file. In this exemple, the name of the class is "SimplerWithCrust". Each new parameter that is added has to be declared in order to be read in the parameter file.

```
SimplerWithCrust<dim>::parse_parameters (ParameterHandler &prm)
    {
      prm.enter_subsection("Material model");
      {
        prm.enter_subsection("Simpler with crust model");
        {
          reference_rho              = prm.get_double ("Reference density");
          reference_T                = prm.get_double ("Reference temperature");
          eta_L                      = prm.get_double ("Lower viscosity");
          eta_U                      = prm.get_double ("Upper viscosity");
          jump_height                = prm.get_double ("Jump height");
          k_value                    = prm.get_double ("Thermal conductivity");
          reference_specific_heat    = prm.get_double ("Reference specific heat");
          thermal_alpha              = prm.get_double ("Thermal expansion coefficient");
        }
        prm.leave_subsection();
      }
      prm.leave_subsection();
```

It's necessary to compile in order to have the shared library associated (cmake and make into the folder of the new ".cc" file).

In the input file, this shared librairy has to be called and the new class that has been implemented has to be added in the Material Model subsection, together with the new parameters that are defined in the ".cc" file :

```
subsection Material model
  set Model name = simpler with crust

  subsection Simpler with crust model
    set Reference density             = 1
    set Reference specific heat       = 1
    set Reference temperature         = 0
    set Thermal conductivity          = 1
    set Thermal expansion coefficient = 1
#    set Viscosity                     = 1

    set Lower viscosity               = 0.4
    set Upper viscosity               = 1
    set Jump height                   = 0.8
  end
end
```

For the moment, three different definitions of the viscosity have been tested (Figure 1) : a constant viscosity (cookbooks/convection-box), a jump of viscosity at a given position (cookbooks/convection-box-viscosity-jump) and a linear viscosity (cookbooks/convection-box-linear-viscosity).

The viscosity jump is defined as :

```
 SimplerWithCrust<dim>::
   evaluate(const typename Interface<dim>::MaterialModelInputs &in, typename Interface<dim>::
            ↪ MaterialModelOutputs &out ) const
   {
     for (unsigned int i=0; i<in.position.size(); ++i)
       {
         const double z = in.position[i][1];
         if (z>jump_height)
           out.viscosities[i] = eta_U;
         else
           out.viscosities[i] = eta_L;

         out.densities[i] = reference_rho * (1.0 - thermal_alpha * (in.temperature[i] -
             ↪ reference_T));
         out.thermal_expansion_coefficients[i] = thermal_alpha;
         out.specific_heat[i] = reference_specific_heat;
         out.thermal_conductivities[i] = k_value;
         out.compressibilities[i] = 0.0;
         out.entropy_derivative_pressure[i] = 0.0;
         out.entropy_derivative_temperature[i] = 0.0;
         for (unsigned int c=0; c<in.composition[i].size(); ++c)
           out.reaction_terms[i][c] = 0.0;
       }
   }
```

The linear viscosity is defined as :

```
LinearViscosity<dim>::
   evaluate(const typename Interface<dim>::MaterialModelInputs &in, typename Interface<dim>::
            ↪ MaterialModelOutputs &out ) const
   {
     for (unsigned int i=0.1; i<in.position.size(); ++i)
       {
         const double z = in.position[i][1];
         out.viscosities[i]= reference_vis * z;

         out.densities[i] = reference_rho * (1.0 - thermal_alpha * (in.temperature[i] -
             ↪ reference_T));
         out.thermal_expansion_coefficients[i] = thermal_alpha;
         out.specific_heat[i] = reference_specific_heat;
         out.thermal_conductivities[i] = k_value;
         out.compressibilities[i] = 0.0;
         out.entropy_derivative_pressure[i] = 0.0;
         out.entropy_derivative_temperature[i] = 0.0;
         for (unsigned int c=0; c<in.composition[i].size(); ++c)
           out.reaction_terms[i][c] = 0.0;
       }
   }
```

For the viscosity jump, a "high" jump and a "low" jump have been tested (Figure 2).

Some results - Next steps

Figure 1 shows preliminary results with a Rayleigh number of 10^4 for the differents viscosity definitions that have been tested. Figure 2 shows the difference between a low and a high viscosity jump.

The next steps will be first to look for simulations with higher Rayleigh numbers and thus more complicate plumes. It could also be interessant to compare with a "smooth" definition, in between the linear viscosity and the jump of viscosity.

Solid-state phase transition (Treatrise on Geophysics -Numerical method for mantle convection, p212). In the long term, it could be good to look at solid state phase transition in the inner core, in order to see the effect on the dynamic. Phase transitions k affect the momentum equation, by adding buoyancy force at the phase boundary. This is called 'phase boundary undulations' and can be described by a dimensionnless phase change function Γ_k that varies between 0 to 1, which respectively represent the two phases separated by this phase change boundary. The momentum equation can be written as

$$\sigma_{ij,j} + \left(RaT - \sum_k Ra_k \Gamma_k \right) \delta_{iz} = 0 \tag{1}$$

where phase change Rayleigh number Ra_k is

$$Ra_k = \frac{\Delta\rho_k g D^3}{\kappa \eta_0} \tag{2}$$

with $\Delta\rho_k$ the density change. The phase change function being defined via the 'excess pressure', which depends on the Clapeyron slope γ_k and the phase change pressure at zero degree temperature for the kth phase transition P_0, it will thus be necessary to define a Clapeyron slope adapted for the inner core (in *aspect* the phase change parameters are for the mantle). The nondimensionnal 'excess pressure' can be written as

$$\pi_k = 1 - d_k - z - \gamma_k (T - T_k) \tag{3}$$

where γ_k, d_k and T_k are the nondimensionnal Clapeyron slope, reference phase transition depth, and reference phase transition temperature for the kth phase transition, respectively. The dimensionless phase change function is thus given as

$$\Gamma_k = \frac{1}{2} \left(1 + tanh \frac{\pi_k}{d} \right) \tag{4}$$

where d is dimensionless phase transition width that measures the depth segment over which the phase change occurs.

The nonlinear energy equation has also to be change, in order to include the latent heating effect along with viscous heating and the adiabatic heating :

$$\left[1 + \sum_k \frac{Ra_k d\Gamma_k}{Rad\pi_k} D_i (T + T_s) \right] \left(\frac{\partial T}{\partial t} + \overrightarrow{v} . \nabla T \right) + \left(1 + \sum_k \gamma_k \frac{Ra_k d\Gamma_k}{Rad\pi_k} \right) (T - T_s) D_i v_z = \Delta T + \frac{D_i}{Ra} \tau_{ij} \frac{\partial v_i}{\partial x_j} + \gamma \tag{5}$$

where T_s, v_z and τ_{ij} are the surface temperature, vertical velocity, and deviatoric stress, respectively; k is a phase change index and D_i is the dissipation number defined as

$$D_i = \frac{\alpha g D}{C_p}. \tag{6}$$

Finally, apply all this definitions to a sphere in order to properly model an inner core dynamics with solid-state phase change transition.

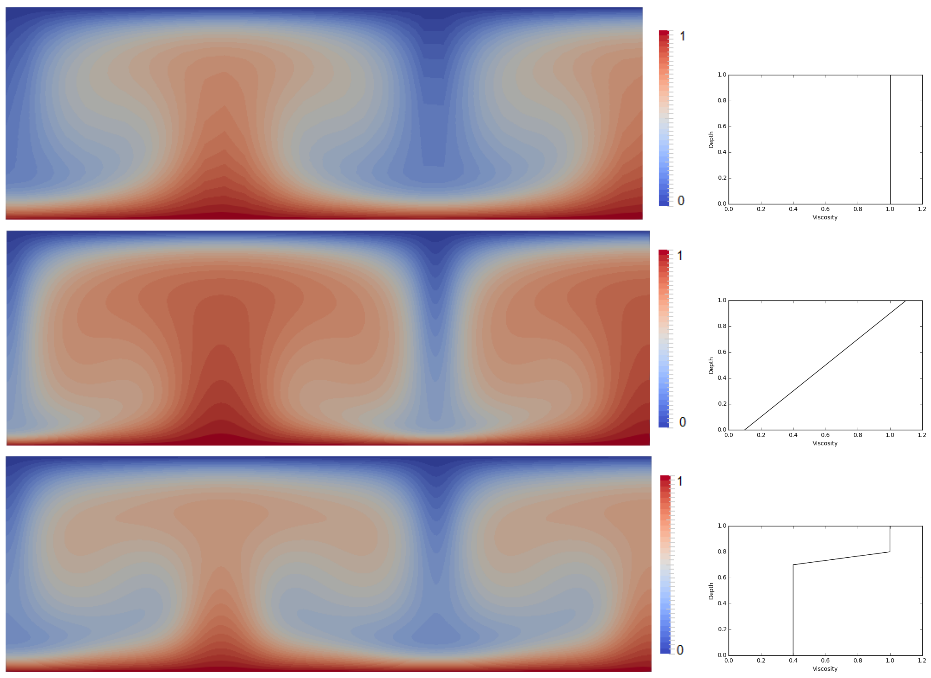

Figure 1: Convection box and the viscosity profile associated with from the top to the bottom, constant viscosity, linear viscosity and viscosity jump, for $Ra = 10^4$.

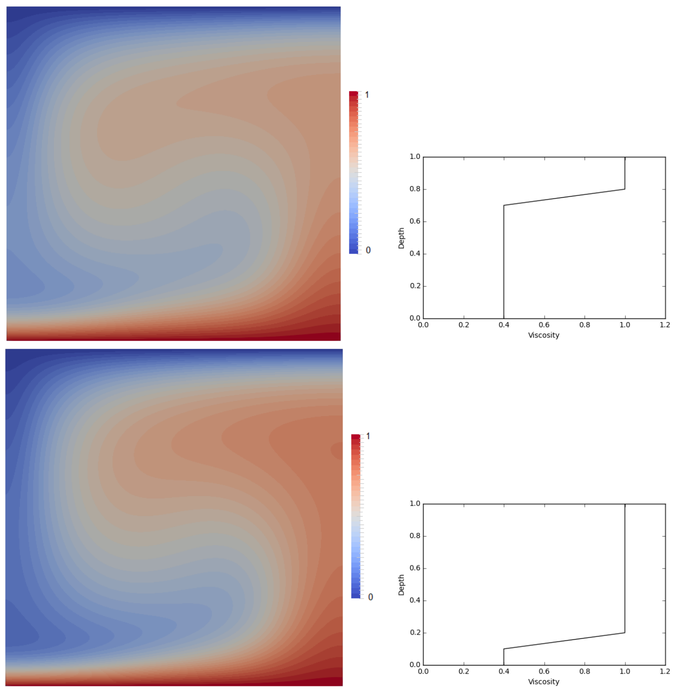

Figure 2: Convection box and the viscosity profile associated for $Ra = 10^4$.