BurnMan

CIDER
June 30th 2016

Timo Heister, Bob Myhill, Ian Rose, Cayman Unterborn, Sanne Cottaar
https://geodynamics.org/cig/software/burnman/

History of BurnMan

- Initial question at CIDER 2012 workshop: ‘What is the Mg/Si ratio of the lower mantle?’ or ‘Do the upper and lower mantle have the same major element composition?’
- Started diving into the forward model - from a composition to seismic velocities.
- Realized that:
  - It is often hard to reproduce the results of a paper.
  - Many people do the same problem with various different codes and excel sheets.
- Started to produce an open source code to improve reproducibility.

CIDER

Cooperative Institute for Dynamic Earth Research

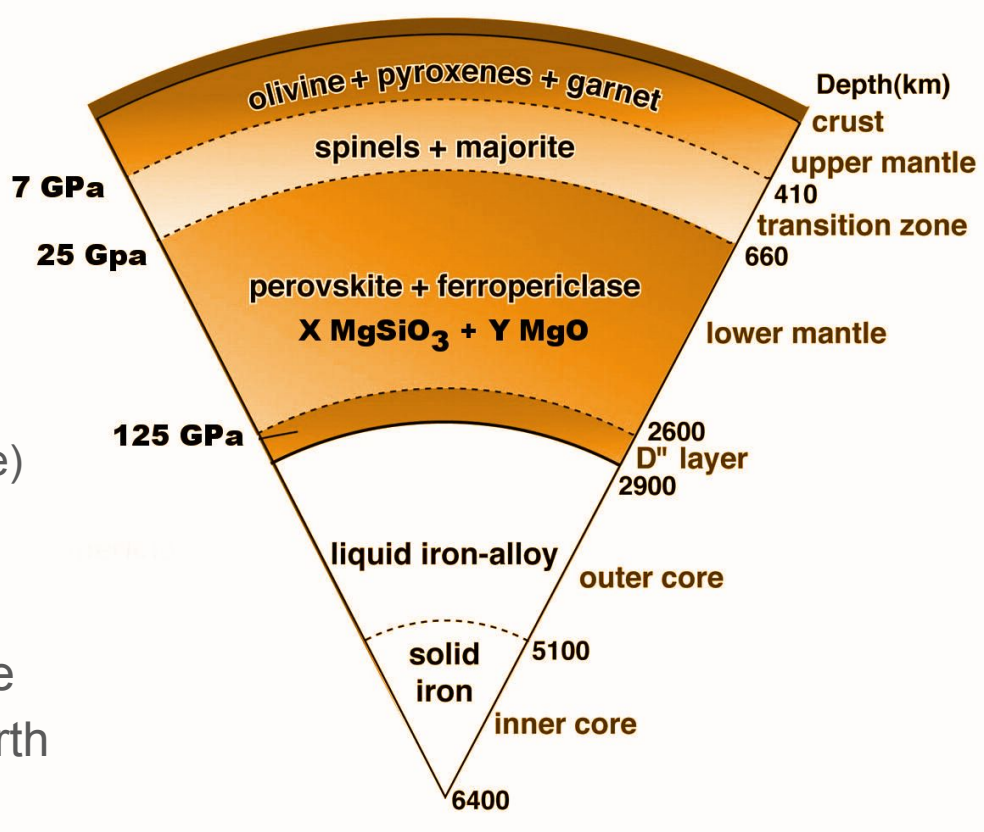
Since CIDER 2012

- AGU 2012 - Hackathon before. Talk at pre-AGU CIDER Workshop. Hackathon at Texas A&M, April 2013
- AGU 2013: Hackathon before, poster at the AGU meeting
- Paper published in 2014. Ian Rose, Timo Heister, Cayman Unterborn, Sanne Cottaar
- June 2014 Version 0.7 distributed through CIG
- Number of us worked at CIDER 2014. *Bob Myhill joins.*
- Hackathon before AGU 2014: Booked AirBnB for a week
- Hackathon before AGU 2015: Booked AirBnB for a week
- April 2016 Version 0.9 distributed through CIG
- Sub-group met in Cambridge, June 2016. *Juliane Dannberg joins.*

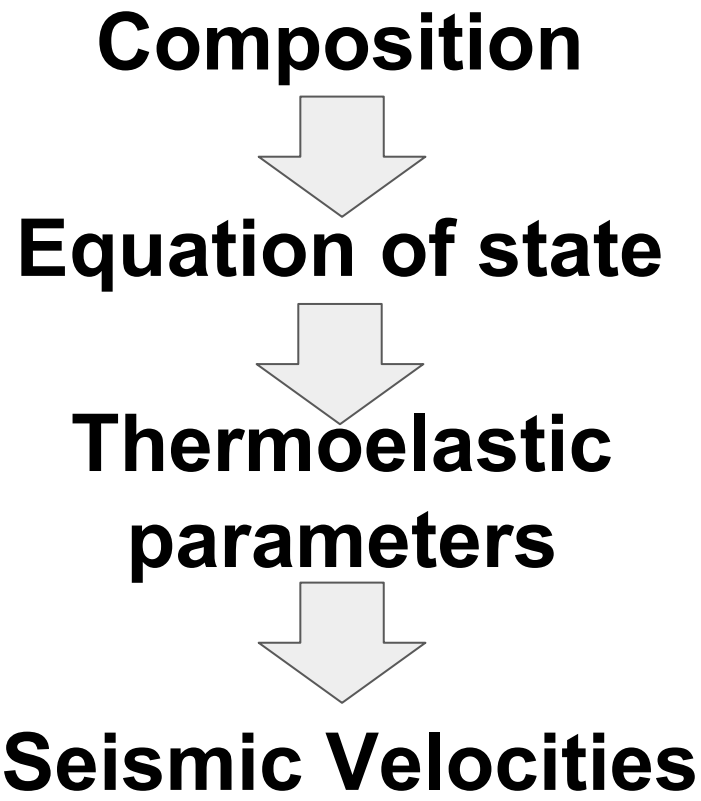
Introduction

- What is BurnMan?
  - thermoelastic toolkit
  - Python library + examples
  - open source
  - modular
  - easy scripting (no user interface)

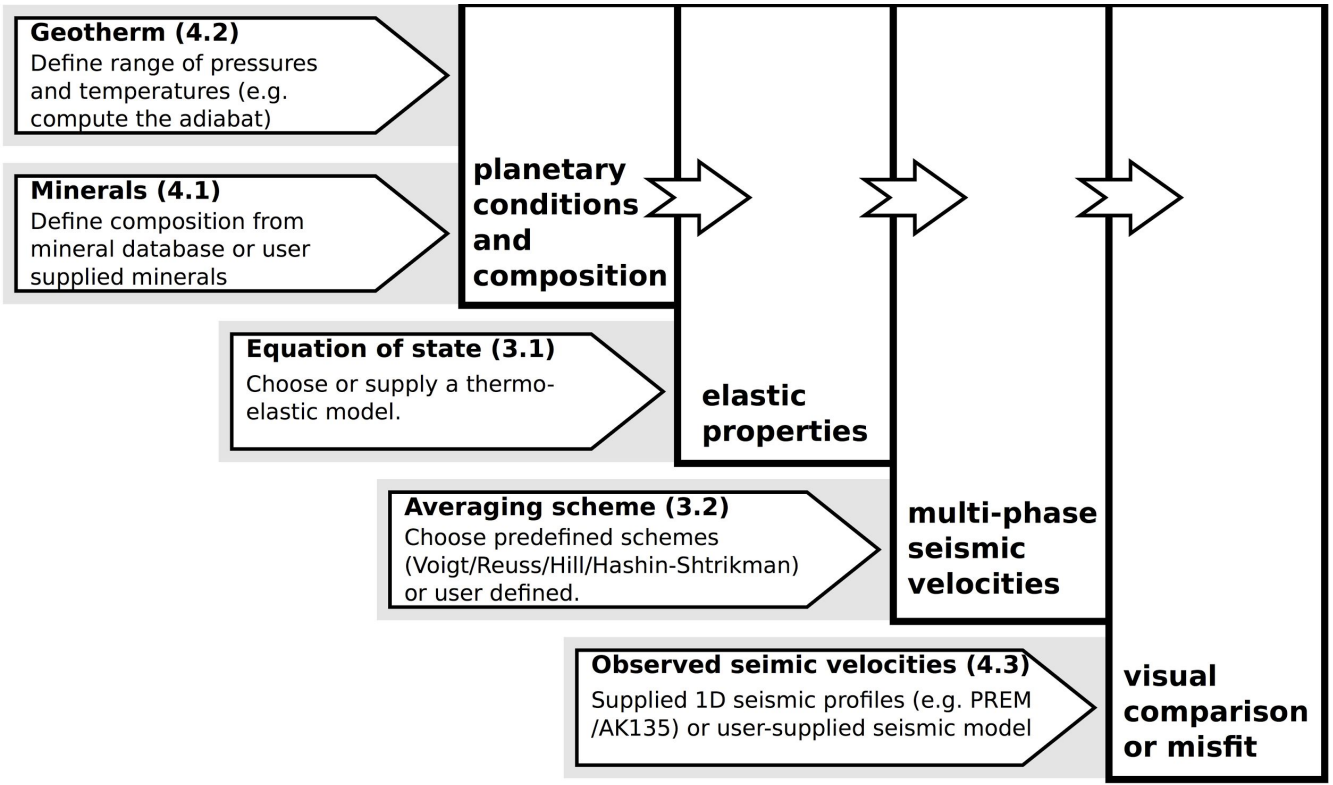
- GOAL: Self-consistently compute the mineral properties for the Earth (and other planets)



Toolbox overview

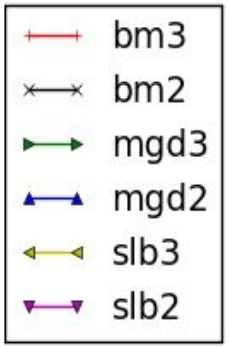


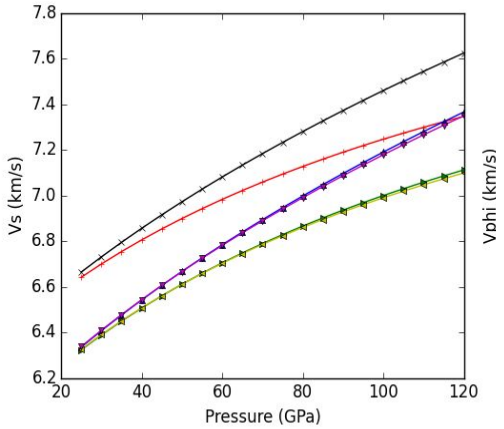
Detailed Overview

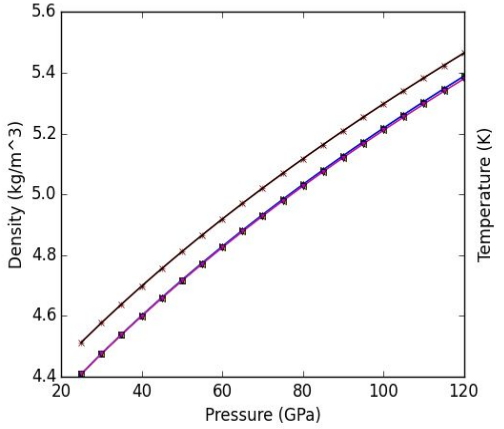


Equations of state

$$\rho = f(P,T,\boldsymbol{X})$$







Isothermal EoS:

- Birch-Murnaghan (2nd and 3rd order)
- Modified Tait (Holland & Powell, 2011)
- Vinet (Vinet et al. 1945)

Full thermal EoS:

- Birch-Murnaghan with a Mie-Gruneisen-Debye thermal part
- Stixrude and Lithgow-Bertelloni variant of the same
- Holland and Powell (2001) extension of the Modified Tait

examples/example_compare_all_methods.py

Equations of state: the bigger picture

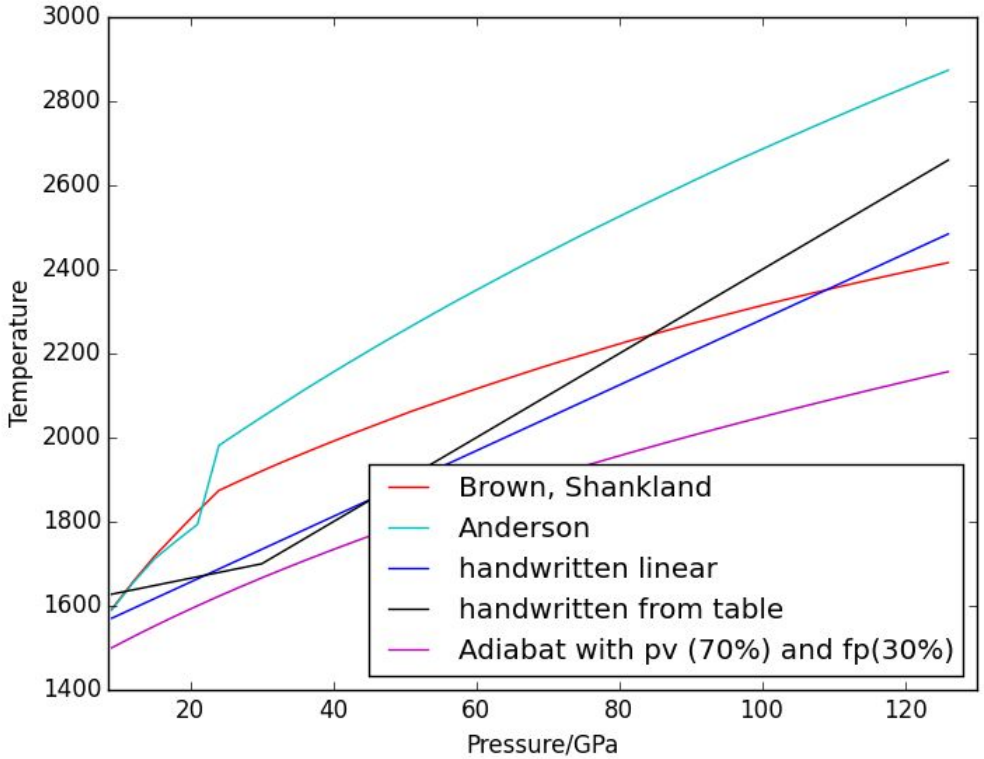
Inputs

Outputs

$$P, T, \chi \longrightarrow \mu, \gamma, G, H, F, S$$
$$\rho, \alpha, K_T, K_S, C_p, C_v$$

Geotherms

T = f(r or P)



examples/example_geotherms.py

Mineral libraries

Parameter libraries are (amongst others) included for mantle mineral from Stixrude and Lithgow-Bertelloni (2011) and Holland & Powell (2011, 2013).

Exact parameters depend on the equation of state.

Easy to add your own minerals.

Note: everything in BurnMan is in SI units.

```
class mg_perovskite (Mineral):
    def __init__(self):
        formula='MgSiO3'
        formula = dictionarize_formula(formula)
        self.params = {
            'name': 'Mg_Perovskite',
            'formula': formula,
            'equation_of_state': 'slb3',
            'F_0': -1368000.0 ,
            'V_0': 2.445e-05 ,
            'K_0': 2.51e+11 ,
            'Kprime_0': 4.1 ,
            'Debye_0': 905.0 ,
            'grueneisen_0': 1.57 ,
            'q_0': 1.1 ,
            'G_0': 1.73e+11 ,
            'Gprime_0': 1.7 ,
            'eta_s_0': 2.6 ,
            'n': sum(formula.values()),
            'molar_mass': formula_mass(formula, atomic_masses)}

        self.uncertainties = {
            'err_F_0': 1000.0 ,
            'err_V_0': 0.0 ,
            'err_K_0': 3000000000.0 ,
            'err_K_prime_0': 0.1 ,
            'err_Debye_0': 5.0 ,
            'err_grueneisen_0': 0.05 ,
            'err_q_0': 0.3 ,
            'err_G_0': 2000000000.0 ,
            'err_Gprime_0': 0.0 ,
            'err_eta_s_0': 0.3 }
        Mineral.__init__(self)
```

Solid solutions

Different models to compute the state of a solid solution:

- Ideal solution
- (A)symmetric regular solution (Holland & Powell, 2003)
- Subregular solution (Helffrich and Wood, 1993)

```
class mg_fe_perovskite(SolidSolution):
    def __init__(self, molar_fractions=None):
        self.name='magnesium silicate perovskite/bridgmanite'
        self.type='symmetric'
        self.endmembers = [[mg_perovskite(), '[Mg][Si]O3'],[fe_perovskite(), '[Fe][Si]O3'],[al_perovskite(), '[Al][Al]O3']]
        self.enthalpy_interaction=[[0.0, 116.0e3],[0.0]]

        SolidSolution.__init__(self, molar_fractions)
```

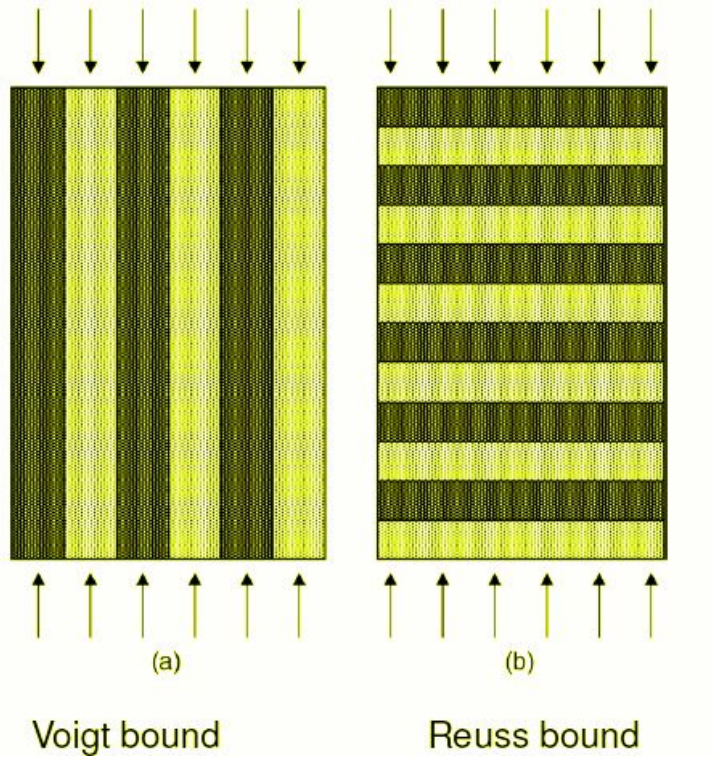
Composites (or rocks)

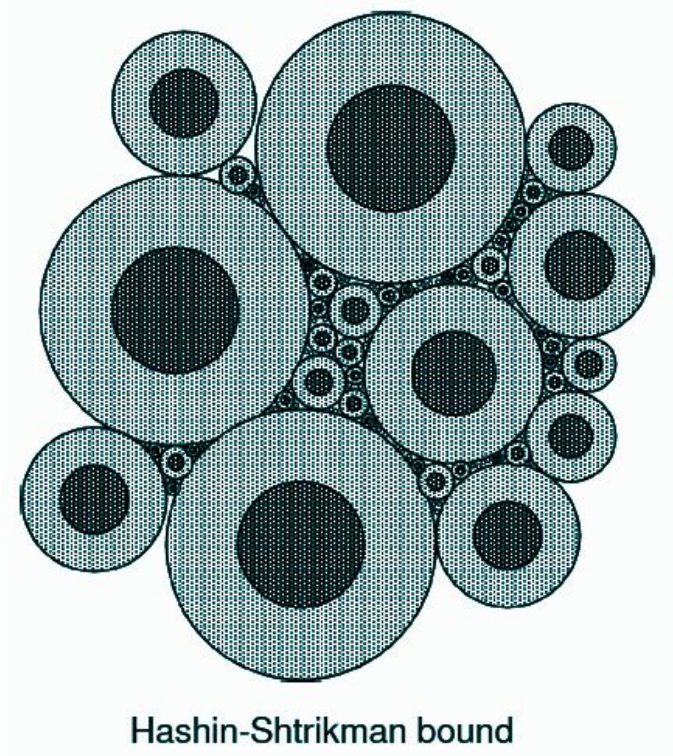
```
#Example: two simple fixed minerals
amount_perovskite = 0.95
rock = burnman.Composite([amount_perovskite, 1.0-amount_perovskite],
                         [minerals.SLB_2011.mg_perovskite(), minerals.SLB_2011.periclase()])


#Example: Mixing solid solutions
# Defining a rock using a predefined solid solution from the mineral library database.
preset_solidsolution=minerals.SLB_2011.mg_fe_perovskite()
# The line below is optional to see which endmembers (and in which order) are in the solid solution
preset_solidsolution.set_composition([0.9,0.1,0.]) # Set molar_fraction of mg_perovskite, fe_perovskite and al_perovskite
rock = burnman.Composite([0.8, 0.2], phases=[preset_solidsolution, minerals.SLB_2011.periclase()])
```

We have some combination of minerals/solid solutions (a rock) and would like to know its:

- density
- bulk modulus
- shear modulus
- seismic wavespeeds

Averaging schemes

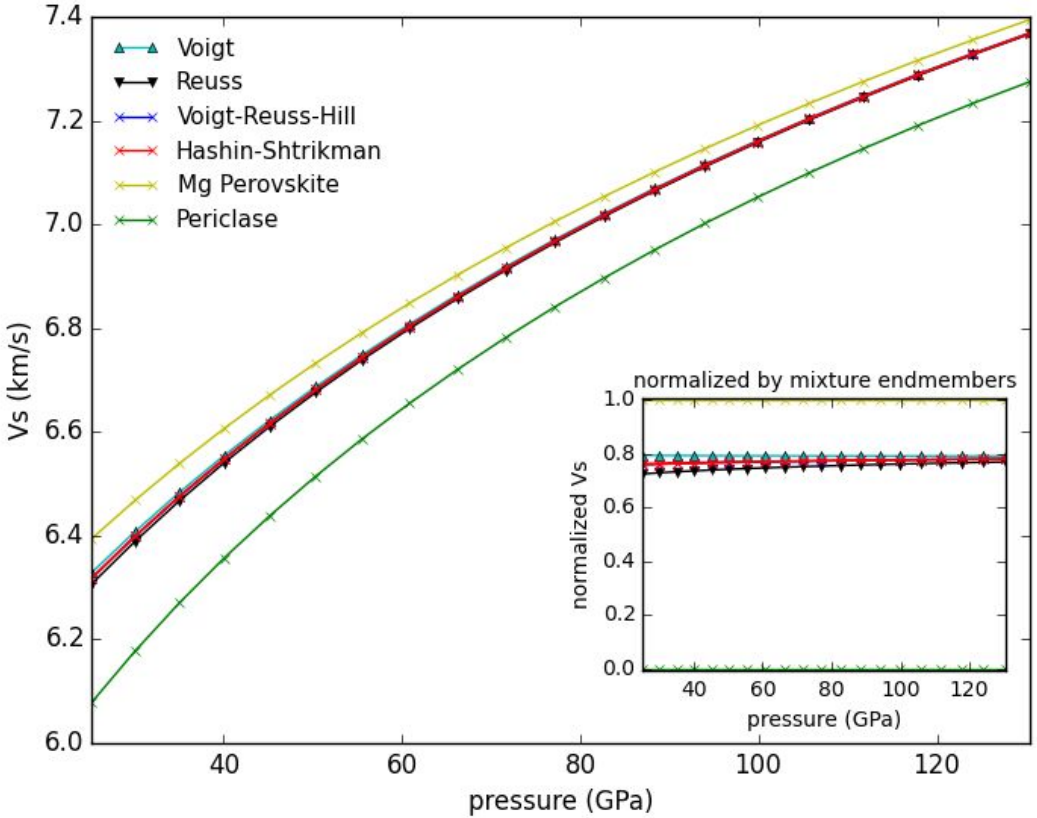




http://petrowiki.spe.org/

Averaging schemes

- Example: different ways to mix a rock made out of 40% periclase and 60% Mg-perovskite
- Does the choice of averaging scheme matter? Sometimes!



examples/example_averaging.py

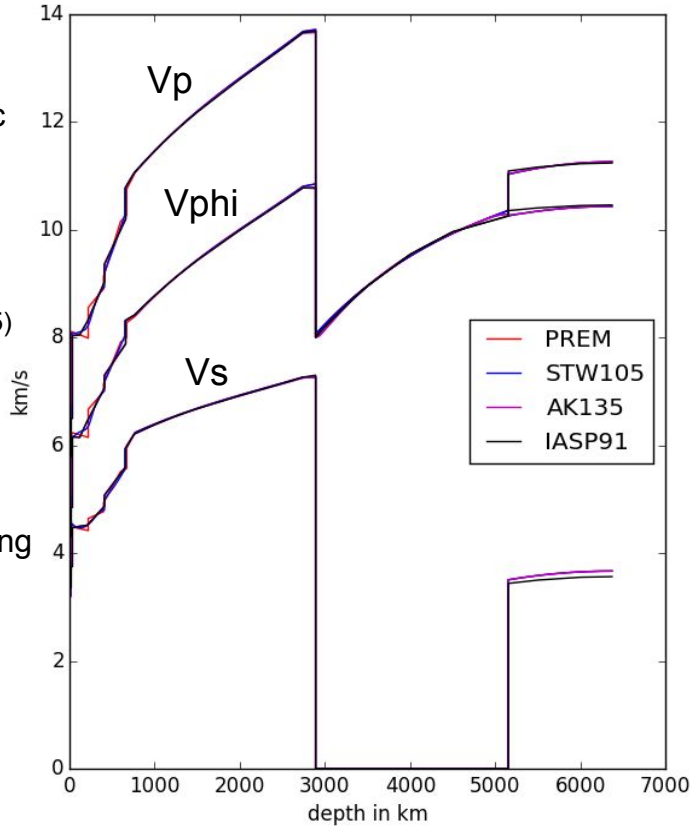
Seismic models

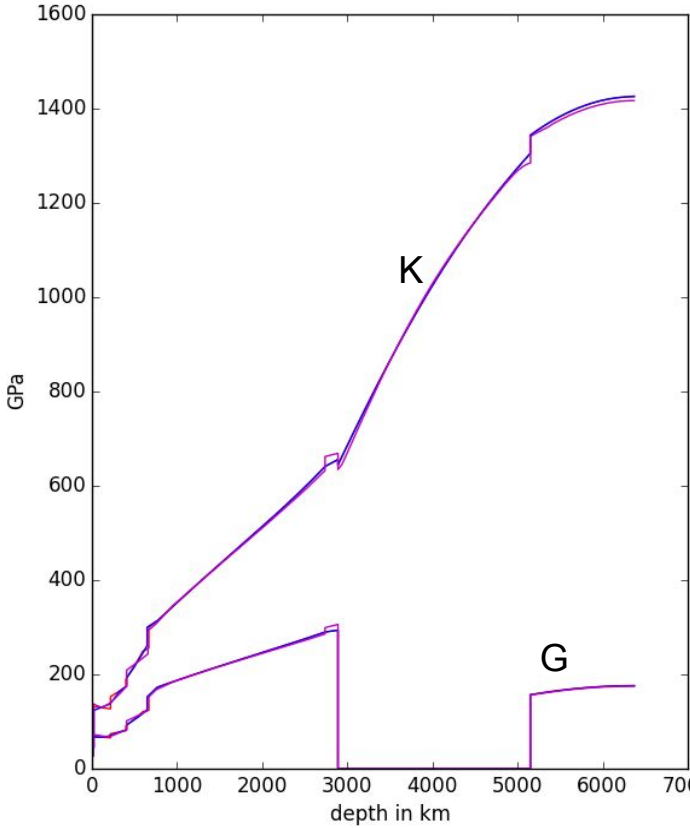
Currently 1D radial models only. Models can be queried for seismic moduli when density is included.

PREM Dziewonski & Anderson (1981)
STW105 Kustowski, Ekström and Dziewonski (2008)
AK135 Kennett, Engdahl & Buland (1995)
IASP91 Kennett & Engdahl (1991)

Note: models are in SI units in BurnMan, but here converted during plotting.

examples/example_seismic.py





User defined modules

- equations of state
- mineral and solid solution libraries
- solid solution models
- geotherms
- seismic models
- averaging schemes
- layers within planet

And contribute them to BurnMan for others to use!

Future plans and wishlist

- Gibbs free energy minimization to build phase diagrams
- Interaction with ASPECT: mantle geodynamics code.
- 3D seismic models as a reference to explore later velocity variations
- Output to codes to produce synthetic seismograms
- GUI?
- melts?
- ...

Some tips and warnings

- Python uses specific indentation. A script might fail if a code block is not indented correctly. We use four spaces and no tabs, mixing these can give trouble.
- Indices require square brackets and function or method calls parentheses (mainly different from Matlab).
- The first index of an array is 0 (e.g. x[0])
- Put dots after numbers to make them floats instead of integers (e.g. 5/3 will give 1 (Python 2.x rounds downward), while 5./3. will give 1.66666666667)
- Everything in BurnMan is in SI units, so *m* instead of *km* and *Pa* instead of *GPa*. These are generally converted to plot results

Getting started

Go to main folder with CIDER tutorials on the virtual machine.

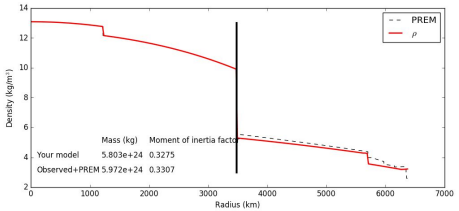
Type: git clone https://github.com/sannecottaar/Cider_buildaplanet

Go into the folder: cd Cider_buildaplanet

Run the code: python example_build_mantle.py

You can modify this file using gedit, vi or emacs: e.g.

gedit example_build_mantle.py &



- Add Fe to the lower mantle (fraction_fe in bridgemanite), upper mantle or both? Can you fit the mass and moment of inertia of the Earth?
- Include ferropericlase in the lower mantle to have the same composition as the upper mantle. One will have to choose a Fe partitioning between bridgemanite and periclase. (use minerals.SLB_2011.ferropericlase(), this is a solid solution of periclase and wuestite)
- Plot the seismic velocities for PREM (velocities are named ‘v_s’, ‘v_p’ and ‘v_phi’ in PREM) and your model. How do they fit?
- Include a mantle transition zone.

Or explore examples in burnman-0.9.0/examples/

The warning comes from the discontinuity between the upper and lower mantle, and integrating across that…. Ignore it.

Including ferropericlase:

```
periclase = minerals.SLB_2011.ferropericlase() #initialize solid solution

periclase.set_composition([1, 0.])  # set ratio of MgO vs FeO (here 20% FeO)

….Composite([bridgemanite, periclase], [0.5 , 0.5]) # define composite of both bridgemanite and periclase
```