Lab Notebook

Date & Time: 01/28/2013
Location: Campus
Computing context: Macho Mac

Continuing from last time:

- Fixing the G2J problem is not optional.

Checking out the indexing in Helena's lookup and bilinear interpolation routines.

Peter's code does something like a *tri*-linear interpolation: between chemical compositions, temperature, and pressure.

In Peter's eos tables, the log(rho) values are specified to 6 significant digits. The conversion from log(rho) to rho (in Peter's code) adds on several more 'significant' figures. This is, of course, not physically realistic (I don't think), b/c you can't know 10^x to more sig figs than plain x... right? Lemme check this.

--> Yeah, if you have 10^x, and you vary the 4th sig fig of x, 10^x only changes at its fourth sig fig, too. Also, various internet math sources agree with this, too.

Figure 1: getting very different rho values from the same P and T inputs b/w the two codes.

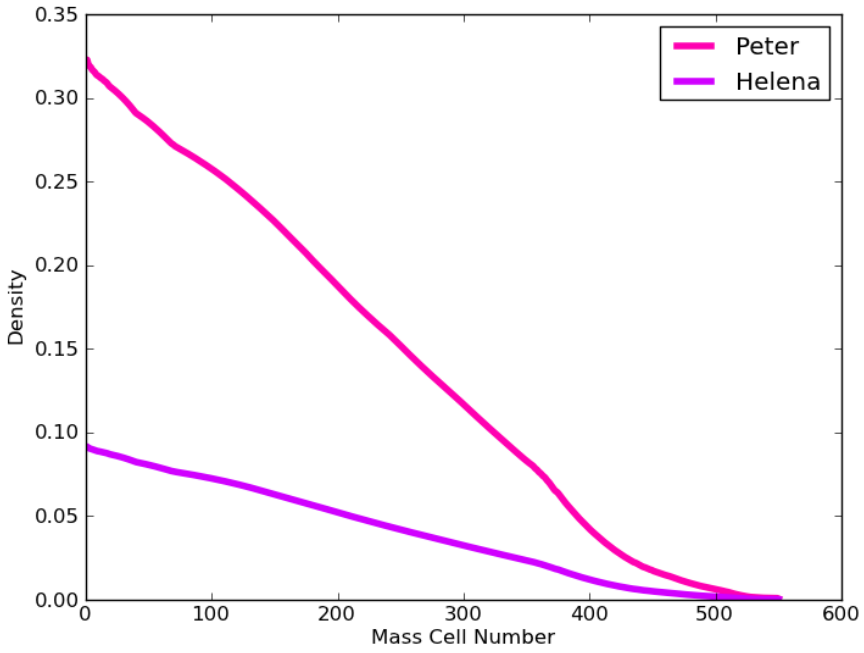

Figure 1:

I give up for today. Start here tomorrow. The difference in Figure 1 has something to do with which lookup table I use in Helena (rhotest vs. the one I'd been using before).

Also, calling 'start' in Peter's code seems to change the rho values that it looks up somehow, too, maybe?

Today's Work:

Figure out why Peter and Helena are getting really different rho values from their lookup tables, given identical P and T inputs (as in Figure 1).

I've figured out how to call the eostab subroutine in Peter's code from the main program (rather than from the invstate subroutine). Figuring out how to do it took several hours, b/c of the arcane rules in Fortran about how you have to specify variables and arrays that you want to pass b/w subroutines.

The advantage to calling eostab directly is: the logP/logT/logrho values get passed to/from the table lookup routine directly, rather than being converted back and forth b/w P/logP etc. values. This cuts down on rounding errors, which may be contributing slightly to the inaccuracies in Helena's eostable lookup proceedure.

To test whether this improves the rho-lookup value determination in Helena, I'll generate a rho lookup table from direct eostab calls, read it into Helena, and check the rho-values it calculates against the ones Peter's code gets for identical T and P values.

This lookup table is called: rho_test_2.txt (in the Helena eos directory).

Enter P value: 1e6

Enter T value: 8e3

Rho:

Helena = 1.93881e-06

Peter = 6.017400664859451E-006

There's a factor of ~6 difference b/w the Peter and Helena lookup values, here, b/w the two codes...

Making a new rho lookup table for Helena (rho_test_3.txt) where the P, T, and rho are all set to their log values. (Before, table had logP, logT, linear rho). Let's see if that makes any difference.

Got better agreement b/w the codes (like, not off by factors of 2 or 3 anymore) with rho_test_5.txt. This one had logP, logT, and logRho values. The resolution of the (Helena) table is the same as the eostable Peter's code is using, though it extends over a larger, uniform T/P range.

Now, the rho values Peter's code returns are slightly larger than the ones Helena returns. This might be a resolution issue.

To test, I'm going to decrease the resolution on the Helena table (make dT and dP bigger), and check to see how the results change. This courser resolution table is rho_test_6.txt

--> It makes a slight difference, but let's test it w/ even more extreme resolution differences.

Make dP and dT smaller this time. (rho_test_7.txt)

--> Increasing the table resolution by a factor of 10 (both in logT and logP) *does* seem to get the rho lookup values to agree (at least, to the print-out precision rounding limit for the fortran write(6,*) statements I'm using).

Lab Notebook

Next steps:

1. Check whether this precision level gets the rest of the lookup values (cP, delta, etc) to agree b/w the codes.

    (a) See if I can figure out why calling invstate returns different logRho values than calling eostab does...

    (b) Try to figure out precisely why Helena doesn't return identical rho-values when I use a lookup table w/ the same precision and inputs as in Peter's code.

    i. This probably has to do w/ the print-out precision his code is using/putting into the Helena lookup table. (I.e., writing out higher precision values than it should).

    A. Figure out how to specify the print-out format to only do a certain number of sig figs, and see whether using a lookup table based on those values can get Peter and Helena to agree on logRho values for identical T/dT P/dP ranges and values.

2. If all the lookup values agree w/ the higher-resolution tables, run each code and plot the resulting densities at each mass cell against each other.

(a) If those agree, then compare the G2J values from the two runs against each other and see whether that issue caused by the slight density differences has been fixed.