10/8/18

Still working in DirtyHapkeEli.py

Realized that the spectrum files imported were pre-processing and that the k’s were post processing so they were different lengths. This is due to DirtyHapke not to HappyHapke

My fix after downloading the processed spectra is:

import csv

with open("SmallGrain.csv", 'rb') as csvfile:

sml\_data = list(csv.reader(csvfile))

with open("MediumGrain.csv", 'rb') as csvfile:

med\_data = list(csv.reader(csvfile))

with open("LargeGrain.csv", 'rb') as csvfile:

big\_data = list(csv.reader(csvfile))

sml\_data = np.asarray(sml\_data, dtype=np.float64)

med\_data = np.asarray(med\_data, dtype=np.float64)

big\_data = np.asarray(big\_data, dtype=np.float64)

self\_spectra = {'sml': sml\_data, 'med': med\_data,

'big': big\_data}

The next thing to figure out was what needs to go into MasterHapke2\_PP

Sticking point was figuring out hapke\_vector\_isow,

This, clearly comes form if we are isometric or Lambertian in our reflectance standard. However, I couldn’t figure out where this term was defined.

This goes back to prog\_state.py where **initialize** function defaults scatter\_type to isotropic. We should really change that to Lambertian.

If it is set to isotropic, then the recurring statement if self.hapke\_scalar.needs\_isow is True, otherwise, it is false.

We want the false case for now.

Then we get down past MasterHapke1\_PP and find that self.hapke\_vector\_isow is set to equal self.hapke\_scalar.copy()

Huh, currently self\_hapke\_isow is a hapke\_model.HapeModel object. I don’t know what that means. But, it can be copied, so that is how we get self\_hapke\_vector\_isow. This is still False in the .needs\_isow department

OK, so what we are trying to figure out is everything that goes into

tmp = snalysis.MasterHapke2\_PP(self.hapke\_vector\_isow, self.pp\_spectra, guesses, lb, ub, ff, tr\_solver=’lsmr’, verbose=2, spts=int(num\_solns))

We dealt with self\_hapke\_vector\_isow. Now we need to deal with self\_pp\_spectra

OK, that comes in with the little pre-processing thing in analysis.py

For key, traj in self.spectra.items():

self.pp\_spectra[key] = analysis.prerocess\_traj(traj, low, jhigh, UV, fit\_order=fit\_order)

Since I already preprocessed these files before bringing them into DirtyHapkeEli, I just need to rename things, I think. Alternatively, I could define low, high, UV, and fit\_order and send self\_spectra through analysis.preprocess\_traj

OK, I think I have all the elements I need to go over to analysis.py

To make sure we have everything, one can use a line like:

print(len(lb), lb.dtype, lb.shape) as equivalent for whos in MatLab

Now I am into analysis.MasterHapke2\_PP

Def MasterHapke2\_PP(hapke, spectra, coefg, lb, up, ff, spts=1, \*\*kwargs)

For lack of a better ideal, I simply renamed self\_pp\_spectra as spectra. I have no clue what hapke variable does.

This is what Aish writes:

1. self.pp\_spectra is the value in the prog\_state.py file that gets passed to the MasterHapke2\_PP in the analysis.py file where it is called spectra. Both are the same thing.

Consider it like -- you have a number 10. When you write your code you call it A. but when I write my code I call it x. When you call my code (call the function) - u pass A, but internally my code still calls it X. but we use the same number 10 regardless.

2. hapke - is the name for the HapkeModel class object. Code for which is found in hapke\_model.py - the two methods are defined there

I guess the programs just know…based on the placement in the order.

Working on the obj\_fcn definition loop. It seems that it is pulling out bcsD ok, cycling through I values. So I guess I should work through each scenario

Grrrr I can’t make any function work. Picking it up in the morning

10/9/18

OK, so I can get the numbers for the small grain size to run through all the equations by hand and the results look reasonable. However, this does not mean that the way they are called by the program is the same as the way I call them.

So now I need Aish’s help to figure out how to feed the program program values and get out output values. She is really busy so I don’t know when that will happen

One thing that I do notice is that spts is a variable and when we choose 1 as spts, does that actually mean 2??? Because 0 would be just coefg, right????

Again, need to talk to Aish. Will continue to work through for the other grain sizes and see if I can get an unreasonable error or something.

I must have copied some formulas wrong because I can’t find a single place where k is used in all of these equations. I am going to try to paste in just the functions and not the classes to see if I can get a more realistic run of things and then just call the functions and name the variables. This means I take out self but I think that shouldn’t matter…..maybe…ironic that I have no concept of self.

OK, may be way to tired to do this but am starting on the pasting in of actual function pieces.

Note to self – make a data folder in Happy Hapke so that this can just run as it is a little more.

Other note to self…still do not understand self

Resaving as DirtyHapkeEli2 but still need to figure out where k goes in DirtyHapkeEli

10/10/18

More things that concern me – what the fuck is a scalar isow and why is it a thing. I really need to change the defaults on that puppy eventually or at least learn how to change the defaults on that puppy.

Apparently when I pull in just

self\_hapke\_scalar = HapkeModel(thetai, thetae, float(n1), float(Bg))

self\_hapke\_scalar.needs\_isow = True….hmmmm

in optimize\_global\_k, default for opt\_strategy = ‘slow’ not ‘fast’

Running into errors checking file inputs:

self\_spectra10 = {}

for key, infile in [('sml', small\_file),

('med', medium\_file),

('big', large\_file),

('file1', file1),

('file2', file2),

('file3', file3),

('file4', file4),

('file5', file5),

('file6', file6),

('file7', file7)]:

#Checks if the infile variable has a string -- sanity check if not all ten files are uploaded

#self.spectra has the number of grain files included in the process

if not infile == '':

self\_spectra10[key] = analysis.loadmat\_single(infile)

\_\_main\_\_:14: FutureWarning: elementwise comparison failed; returning scalar instead, but in the future will perform elementwise comparison

Traceback (most recent call last):

File "<ipython-input-54-29b79644f05a>", line 15, in <module>

self\_spectra10[key] = analysis.loadmat\_single(infile)

File "analysis.py", line 14, in loadmat\_single

return np.loadtxt(path\_or\_file\_obj)

File "C:\Users\ecskl\Anaconda2\lib\site-packages\numpy\lib\npyio.py", line 1045, in loadtxt

first\_vals = split\_line(first\_line)

File "C:\Users\ecskl\Anaconda2\lib\site-packages\numpy\lib\npyio.py", line 987, in split\_line

line = regex\_comments.split(line, maxsplit=1)[0]

TypeError: buffer size mismatch

\*\*\*\*Is infile a command or an input file???

Eck, hoping this is just a me problem:

self\_spectra10 = {}

for key, infile in [('sml', small\_file),

('med', medium\_file),

('big', large\_file),

('file1', file1),

('file2', file2),

('file3', file3),

('file4', file4),

('file5', file5),

('file6', file6),

('file7', file7)]:

#Checks if the infile variable has a string -- sanity check if not all ten files are uploaded

#self.spectra has the number of grain files included in the process

if not infile == '':

print('Not in file must load')

Not in file must load

Not in file must load

Not in file must load

\_\_main\_\_:14: FutureWarning: elementwise comparison failed; returning scalar instead, but in the future will perform elementwise comparison

Something fun to remember would be self\_spectra.keys() and self\_spectra.keys are different. The first one works. I really don’t understand these empty parentheses

OK, by changing that one enumerate to an enumerate(sorted with a reverse=True, I have changed the error from whatever it was to X0 is infeasible.

According to the interwebs, this simply means that X0 is not in lb ub range. This is something to go on.

I need to figure out where a bunch of these defaults are coming from because they are all wrong. Here goes nothing with editing files I have no business editing. I guess that makes me a real programmer. All new files are in Documents/Python Scripts/my copy 101018

I can’t find where the defaults are. I am trying to work through the functions and programs as they are written from the .py files, but this is very tricky because I have to essentially delete all the functions and hope I do it right. But, as far as I can see, the defaults for a lot of the parameters are incorrect and I can’t figure out why.

First 94 lines of prog\_state.py run but spectra.keys() is still med, sml, big

To fix this, I think I have to figure out where it brings it in from the web.

Mayeb we really just need everything to be file 1:10 without the small medium and big

10/11/18

Down to line 128 works. Changing all *self.* To self\_ and taking self out of functions and those are the only changes I am making to run this section by section. So….maybe it works this way but I may be strange in variable space.

OK, So sometimes I also have to run the functions. But, then nothing exists outside of the functions which makes checking all parameters really hard…..

OK, so what I did for lines 1-94 was first to run it as a defined function, and then run the function and then run all lines as though they were outside of the function to make sure that all variables worked the way they should.

Renaming all to file 1: file10. The problem is that dictionaries do not have order but I do not know when to bring in the sorting function….Right now they show up in reverse

Have also made a second file in function form… I take out the classes and self gets transformed but I am hoping I can run this from the command line and see what is going on with the plotting.

It looks like k is not minimizing correctly but that may be because I am running everything weirdly

OK, nevermind, that doesn’t work because it keeps giving me indent errors.

But I can run it as prog\_state\_functions.initialize. However, it doesn’t plot

I really don’t understand plots

OK, I am finally figuring things out. For instance, I am now seeing where the defaults in the website are recorded: They are in ui.html. I have changed some of them but have run into a problem with the way that upper and lower bounds are placed for the multi minimization – lines 336-366. It looks like the bounds are a group so that we can’t change the grain size bounds independently for each size and have it pre-display. However, some other things have been fixed so maybe that will help.

Ooooo got new error message: built in function or method object is not iteritable.

I think we are getting closer to figuring out the problem

I bet the x0 is infeasible is about the fact that we never assign guesses for med and large grain sizes. I think that going back one generation in some of the html coding may fix this.

10/12/18

Today I am going to try pasting some of the old html file into the new one to see if I can fix some of the issues we are getting. Cause breaking stuff more is what I do.

10/15/18

Worked through html file and found it to be very different. Working through it more today

Decided I will start small. I am replacing the option to choose small medium and big from the single k lookup table back to having three sections.

I did some cut and paste. Let’s see if it works.

Those sections work fine but the next one does not because ddlGuessKey is no longer there. I need to have someone help me reintroduce that.

But frankly I just want to see if it is the html section or the py section that is screwed up so I am going to paste back in the old stuff from CJs program

OK, we still get the error that x0 is infeasible. This is good information. This means that the problem is with something in the py code. I will go back to working with that now.

Back to program state. I have debunked the thought that we are getting the x0 is infeasible because we are not assigning med and large variables…kind of. CJs code certainly does assign what it needs to …unless there is some strange mismatch that I am not accounting for.

I should bring up CJs prog state and see if I can see where it is changed in Aish’s version and how to make it work again.

Oh, I think I might finally understand classes. For example, self\_hapke\_scalar is defined in initialize but not in preprocess but when both of them are in the class Program\_State, *self.*hapke\_scalar makes perfect sense to the computer. The class extends the scope of the variable using the term *self.*

Aaaaaand I still need Aish’s help calling the damn function as it is written \*\*\*\*\*

Changing line 137 to reflect CJ’s code since I mucked with the html.

Changing from:

solved\_k, scat\_eff = analysis.MasterHapke1\_PP(

self.hapke\_scalar, traj, b, c, ff, s, D, debug\_plots=True)

to: self.ks[key] = analysis.MasterHapke1\_PP(

self.hapke\_scalar, traj, b, c, ff, s, D, debug\_plots=True)

nope, nevermind. That should not have mattered because defined below….

Eh, trying it anyways. What is with this bug in my face…I swear I have killed it three times and it is still here.

New part:

def solve\_for\_k(self, key='file2', b=0, c=0, ff=0.000000001, s=0, D=0):

b, c, s, D, ff = map(float, (b, c, s, D, ff))

self.guesses[key] = (b, c, s, D, ff)

traj = self.pp\_spectra[key]

plt.close('all') # hack!

#The hidden treasure where all the brains are hidden

self.ks[key] = analysis.MasterHapke1\_PP(

self.hapke\_scalar, traj, b, c, ff, s, D, debug\_plots=True)

#self.ks[key] = solved\_k

#self.scat\_eff\_grain[key] = scat\_eff

figures = [plt.figure(i) for i in plt.get\_fignums()]

return 'Solved for k: ', 'sk-' + key, figures

That created a whole new error message: Setting an array element with a sequence.

Also going to replace

def optimize\_global\_k(self, guess\_key='file2', opt\_strategy='slow', num\_solns=1,lowk=0, upk=0, lowb\_sml=-1.7, lowb\_med=-1.7, lowb\_big=-1.7, upb\_sml=1.7, upb\_med=1.7, upb\_big=1.7, lowc\_sml=-1, lowc\_med=-1, lowc\_big=-1, upc\_sml=1, upc\_med=1, upc\_big=1, lows\_sml=0, lows\_med=0, lows\_big=0, ups\_sml=0.6, ups\_med=0.6, ups\_big=0.6, lowD\_sml=21, lowD\_med=36, lowD\_big=50, upD\_sml=106, upD\_med=150, upD\_big=180,

\*\*kwargs):

With

def optimize\_global\_k(self, guess\_key='file2', opt\_strategy='slow',

lowb1=0, lowb2=0, lowb3=0, upb1=0, upb2=0, upb3=0,

lowc1=0, lowc2=0, lowc3=0, upc1=0, upc2=0, upc3=0,

lows1=0, lows2=0, lows3=0, ups1=0, ups2=0, ups3=0,

lowD1=0, lowD2=0, lowD3=0, upD1=0, upD2=0, upD3=0,

lowk=0, upk=0, num\_solns=1, \*\*kwargs):

my reasoning is that the variables are in a different order and I am lead to believe that in python, order matters.

Same error message. First thing tomorrow I will track down that pesky array and sequence….somehow…

10/16/18

Found a problem, not all of the problems. When you run line by line through CJs code, what you get for self.ks[key] is different than what you get in Aish’s code. And I mean really, really different. You get actual answers in CJs code and just an ascending array from 0.03-0.254 in Aish’s code. Time to figure out why that is.

10/17/18

Nope! That is only the case if you forget and put zeros in for grain size etc….

Although I did just discover that self\_spectra is 224 points long where self\_ks is 225. This could be a problem

Nope, that’s the same in CJs program

10/19/18

OK, I am working in the main optimization function and I am seeing that if you only run one grain size, your guesses matrix has a lot of empties in it because of:

for i, key in enumerate(self\_guesses.keys()):

g = self\_guesses[key]

# Unpacks the b,c,s,D values for each grain size into one large array. g holds b,c,s,D,f -- we take only the first four

guesses[i:total\_guesses:no\_of\_grain\_samples] = g[:4]

# guesses Example:

# for sml, med anf big grain sizes

# [sml-b, med-b, big-g, sml-C, med-C, big-C, sml-S, med-S, big-S, sml-D, med-D, big-D, 215 values of K]

# total with the length of K - 4 values for each grain size -- this is the magic 12

ff[i] = g[4]

guesses[total\_guesses:] = k

But, when I added back in the other grain sizes to the html file, I still had the problem with x0 being infeasible so I am not sure this is the problem. Maybe a problem but not the problem

OK, in line by line Happy Hapke, I ran with sml med and large as key which makes self\_guesses.keys() complete for all three files

OK now we have populated different guesses into self\_guesses to make it easier to track

Ok, guesses populates but in reverse because that is the way the keys list (file3, file2, file1)

As I suspected, guesses are opposite in order to lb and up with respect to files 1, 2, and 3.

I am adding sorting to the enumerate in the for loop for populating guesses

I THINK I JUST SUCCESSFULLY RAN THE FUCKING PROGRAM!!!! (only in the line by line not in the webshell)

But still an error in the web version… Now:

Setting an array element with a sequence

The internet has this to say:

From the code you showed us, the only thing we can tell is that you trying to create an array from a list that isn't shaped like a multi-dimensional array. For example

numpy.array([[1,2], [2, 3, 4]])

or

numpy.array([[1,2], [2, [3, 4]]])

will yield this error message, because the shape of the input list isn't a (generalised) "box" that can be turned into a multidimensional array. So probably UnFilteredDuringExSummaryOfMeansArray contains sequences of different lengths.

**Edit**: Another possible cause for this error message is trying to use a string as an element in an array of type float:

numpy.array([1.2, "abc"], dtype=float)

That is what you are trying according to your edit. If you really want to have a NumPy array containing both strings and floats, you could use the dtype object, which enables the array to hold arbitrary Python objects:

numpy.array([1.2, "abc"], dtype=object)

Without knowing what your code shall accomplish, I can't judge if this is what you want.

[share](https://stackoverflow.com/a/4675383)[improve this answer](https://stackoverflow.com/posts/4675383/edit)

[edited Jan 13 '11 at 22:31](https://stackoverflow.com/posts/4675383/revisions)

answered Jan 12 '11 at 23:51

[[https://www.gravatar.com/avatar/2dceea858ad8f1577bec6ddaa0485d15?s=32&d=identicon&r=PG](https://stackoverflow.com/users/279627/sven-marnach)](https://stackoverflow.com/users/279627/sven-marnach)

[Sven Marnach](https://stackoverflow.com/users/279627/sven-marnach)

331k75733681

* Thank you, but I do not think that is the answer. I have added the contents of the array when it throws the error above. And it seems to me that it is a box when I paste it into notepad and examine it line by line. Any other ideas? – [MedicalMath](https://stackoverflow.com/users/573404/medicalmath" \o "512 reputation) [Jan 13 '11 at 21:06](https://stackoverflow.com/questions/4674473/valueerror-setting-an-array-element-with-a-sequence#comment5166775_4675383)
* 1

Your edit seems to have solved my problem. I needed to set the dtype=object. Thank you very much. – [MedicalMath](https://stackoverflow.com/users/573404/medicalmath" \o "512 reputation) [Jan 14 '11 at 7:53](https://stackoverflow.com/questions/4674473/valueerror-setting-an-array-element-with-a-sequence#comment5171771_4675383)

* This question has been completely answered. – [MedicalMath](https://stackoverflow.com/users/573404/medicalmath" \o "512 reputation) [Jan 14 '11 at 7:54](https://stackoverflow.com/questions/4674473/valueerror-setting-an-array-element-with-a-sequence#comment5171778_4675383)
* Another possibility could be an issue in 1.9 when building an array of objects (not necessarily lists) that implement \_\_getitem\_\_ as specified here:[github.com/numpy/numpy/issues/5100](https://github.com/numpy/numpy/issues/5100) – [dashesy](https://stackoverflow.com/users/311567/dashesy" \o "1,391 reputation) [Nov 21 '14 at](https://stackoverflow.com/questions/4674473/valueerror-setting-an-array-element-with-a-sequence#comment42650367_4675383)

The code:

print(X)

pred\_X = np.array(pred\_X)

pred\_Y = np.array(pred\_Y)

X = np.array(X)

Y = np.array(Y)

X = preprocessing.scale(X)

pred\_X = preprocessing.scale(pred\_X)

print(x):

[[547180.0, 120.0, 113.0, 456701.0, 1.0, 6.43, -1.0, 0.313, 0.42, 0.267 3.0, 11800.0, 607208.0, 120.0, 113.0, 456701.0, 1.0, 0.273, 0.331, 0.154, 6.0, 10300.0, 458015.0, 113.0, 120.0, 45328 6.0, 1.0, 2.54, -1.0, 0.32, 0.443, 0.257, 3.0, 92000.0, 543685.0, 120.0, 113.0, 456701.0, 1.0, 6.43, 1.0, 0.296, 0.4, 0.234, 2.0, 8800.0, 594809.0, 475582.0, 120.0, 113.0, 456701.0, 1.0, 1.0, 0.295, 0.384, 0.264, 4.0, 7700.0],

[547180.0, 120.0, 113.0, 456701.0, 1.0, 6.43, -1.0, 0.313, 0.42, 0.267, 3.0, 11800.0, 607208.0, 120.0, 113.0, 456701.0, 1.0, 0.273, 0.331, 0.154, 6.0, 10300.0, 458015.0, 113.0, 120.0, 453286.0, 1.0, 2.54, -1.0, 0.32, 0.443, 0.257, 3.0, 92000.0, 543685.0, 120.0, 113.0, 456701.0, 1.0, 6.43, 1.0, 0.296, 0.4, 0.234, 2.0, 8800.0, 594809.0, 435062.0, 120.0, 113.0, 456701.0, 1.0, 1.0, 0.312, 0.364, 0.154, 5.0, 6900.0],

[547180.0, 120.0, 113.0, 456701.0, 1.0, 6.43, -1.0, 0.313, 0.42, 0.267, 3.0, 11800.0, 607208.0, 120.0, 113.0, 456701.0, 1.0, 0.273, 0.331, 0.154, 6.0, 10300.0, 458015.0, 113.0, 120.0, 453286.0, 1.0, 2.54, -1.0, 0.32, 0.443, 0.257, 3.0, 92000.0, 543685.0, 120.0, 113.0, 456701.0, 1.0, 6.43, 1.0, 0.296, 0.4, 0.234, 2.0, 8800.0, 594809.0, 446308.0, 120.0, 113.0, 456701.0, 1.0, 0.0, 0.221, 0.28e, 0.115, 8.0, 6400.0]]

The Error:

Traceback (most recent call last):

File "sampleSVM.py", line 46, in <module>

X = preprocessing.scale(X)

File "/home/user/.local/lib/python3.5/site-packages/sklearn/preprocessing/data.py", line 133, in scale

dtype=FLOAT\_DTYPES)

File "/home/user/.local/lib/python3.5/site-packages/sklearn/utils/validation.py", line 433, in check\_array

array = np.array(array, dtype=dtype, order=order, copy=copy)

ValueError: setting an array element with a sequence.

Your input array X is malformed. There are 59 elements in row 1, and 58 in rows 2 & 3. When you convert to a numpy array it becomes an array of shape (3,) with dtype=Object.

The solution is to check and fix your input data. Each row in X must be the same length.

I can’t figure out what is wrong here. It looks like all I have is a numpy.ndarray for guesses but it is making the program endlessly unhappy.

OK, working in a combination of prog\_state\_functions and line by line Happy Hapke.

I first define the function from prog\_state\_functions, then define the terms it is to take, then call the function on those defined variable. Then I paste in the function from line by line because the next function is dependent upon things from the previous and I can’t figure out how to make that work cause I have no class. Hehe.

I get through to optimize\_global\_k and can run through the calling of MasterHapke2 and it iterates (but I am sure I need to fix Xtol so it iterates more) and then I seem to have a problem with unpacking variables in the next lines. This is very different from when I run it from the web.

Nope… here is the problem:

for i, key in self\_spectra.keys():

axes[i,0].set\_ylabel(key)

which gives the error: ValueError: too many values to unpack

I commented it out for the moment and fixed all the other self.spectra.keys (made self.spectra.keys()) but am still getting the same error from the web for that same line. So why is it a problem there and not in the line by line????

I put back in the change Aish made:

solved\_k, scat\_eff = analysis.MasterHapke1\_PP(

self\_hapke\_scalar, traj, b, c, ff, s, D, debug\_plots=True)

self\_ks[key] = solved\_k

self\_scat\_eff\_grain[key] = scat\_eff

And then it worked. I guess all the other changes I made helped. I officially unbroke it.

10/20/18

OK, Now that that is done spending ridiculous amounts of time trying to figure out how to work with scipy.optimize.least\_squares. Finally found documentation that makes sense. Matlab really has python beat for documentation. Loos like I needed to find the damn thing first, which was in analysis.py.

for spt in start\_points:

res = least\_squares(obj\_fn, spt, bounds=bounds, xtol=1.0e-08, method='trf', \*\*kwargs)

solutions.append(res)

return solutions

OK, that is now changed to e-15 and it is scrolling like it used to in Matlab. Now I can actually start testing this puppy!

It is doing something really funny with k. I think I have to line by line through and figure out what it is comparing to what because the first point in k is getting really really big. I am not sure the maths are correct.

After running that and the alternating a few times I have decided that a) alternating doesn’t work and b) the data and spectra are mismatched somehow.

Tomorrow line by line to figure out why

10/21/18

Line by line HappyHapke

Initialize

Self\_spectra is file3, file2, file1

Preprocess

Function analysis.preprocess\_traj works off of keys and does not require order.

Self\_pp\_spectra in order file3, file2, file1

Solve for k

Key is only one file and the traj is derived from self\_pp\_spectra for one key only

Then traj and all guesses are fed to MasterHapke1\_PP

Note that I really need to check these Hapke programs because I am not sure they are the most recent versions

Self\_guesses(key) are b,c,s,d,ff

This delivers self\_ks dictionary with key associated k.

One note about this k is that the end point is really high – I think it has to do with

preprocessing

But that doesn’t matter here because these are just guesses

But yup, removing the UV point works on the endpoint k value

Global K

Because I sorted self\_guesses.keys(), the values in self\_guesses go 1, 2, 3. This may be what is screwing with the program solver. I will get to that directly. However, this is now making the bounds line up correctly I think.

Still need to fix this section:

#this needs to change for new number scheme - 'grain' won't work

for grain in self\_spectra.keys():

if grain not in ['file1', 'file2', 'file3']:

temp\_low\_bound.append(kwargs['lowb'+grain])

temp\_low\_bound.append(kwargs['lowc'+grain])

temp\_low\_bound.append(kwargs['lows'+grain])

temp\_low\_bound.append(kwargs['lowD'+grain])

So in the end we create a tuple: self\_bounds. Now if I remember correctly a tuple is ordered….

Analysis.MasterHapke2\_PP gets fed:

Self\_hapke\_vector\_isow, self\_pp\_spectra, guesses, lb, ub, ff, tr\_solver,

verbose, spts

corresponding in order to:

def MasterHapke2\_PP(hapke, spectra, coefg, lb, ub, ff, spts=1, \*\*kwargs):

So on command line making hapke = self\_hapke\_vector\_isow.copy()

Spectra = Self\_pp\_spectra.copy()

Coefg = guesses.copy() etc…

\*spectra gets transformed into actuals, which is a list….so…we don’t actually know which is which

If you change

actuals = [spectra[key][:,1] for key in spectra.keys()] to

actuals = [spectra[key][:,1] for key in sorted(spectra.keys())]

you flip the order – this way file 1 comes first, always..

\*\*\*WHERE DO YOU SAY HAPKE\_MODEL is HAPKE?????\*\*\*\*\*

Wait, is this because the first coeff is hapke??? So instead of self.blah, it is hapke.blah???

OK, self\_hapke\_vector\_isow has property of needs\_isow = False but I can’t seem to investigate the other attributes cause I don’t know how ☹

IN MasterHapke2\_PP, obj\_fn is defined but it is never called that I can see, so what is its point????

Coefg is never renamed coef so wtf??

In obj\_fn, scat calculates as it should. Equations are checked with Hapke book

Rc calls radiance coefficient, which calls single particle phase and \_Hu\_Hu0

Single\_particle\_phase calls self.cosg, which I think is in \_init\_angles

\*\*\*HOW DOES PYTHON KNOW TO LOOK THERE\*\*????

10/22/18

I found what I think is an error in the way CJ had mathed Hu. He had:

1/(1 - u\*scat\_eff\*(r0+np.log((1+u)/u)\*(0.5 - r0\*u)))

When the equation is:

#Hapke 1993 equation 8.57

#H(x) = {1/{1-[1-gamma]\*x\*[r0+(1-0.5\*r0-r0\*x)\*ln((1+x)/x)]}}

So I replaced it with:

gamma = np.sqrt(1 - scat\_eff)

return 1/(1-(1-gamma)\*u\*(r0 + (1 - 0.5\*r0 - u\*r0)\*np.log((1 + u)/u)))

While I suppose it is possible that it reduces to something similar to what CJ wrote, when you run the two through, the answers are different.

CJ: [1.34189223 1.49201633 1.6147388 1.68383401 1.72679917 1.7507955

1.74402596 1.76647254 1.82049068 1.84092576 1.84065544 1.85856516

1.8725959 1.88368766 1.90613256 1.92397027 1.93435275 1.9448722

1.95912646 1.97090854 1.98207049 1.99384801 2.00468069 2.01995705

2.03420979 2.05188088 2.06868179 2.08498048 2.09767945 2.10706204

2.11430474 2.12047795 2.12616241 2.130616 2.13714915 2.14299991

2.14939447 2.15348207 2.16002903 2.16599798 2.171012 2.17499296

2.17918108 2.18351361 2.18455314 2.18817933 2.18866828 2.18908257

2.19110908 2.18842372 2.18591307 2.18286552 2.18114727 2.17865015

2.17584459 2.17239385 2.17088666 2.16904764 2.16550355 2.16133363

2.15675025 2.15470855 2.15057093 2.15233284 2.14683659 2.14454723

2.13707467 2.13186245 2.12544078 2.11863269 2.10936235 2.10181919

2.0917145 2.08385845 2.0742844 2.06442874 2.05431516 2.04420796

2.0319571 2.02264672 2.01388275 2.0052556 1.99595994 1.987545

1.97855715 1.97321272 1.96450273 1.95936675 1.9560364 1.95113576

1.94679588 1.94286429 1.94196553 1.939229 1.9354863 1.9354863

1.93584695 1.93613575 1.93685714 1.93979273 1.94240544 1.94361816

1.945538 1.94999753 1.95579846 1.95671453 1.9592928 1.9608678

1.96133651 1.96060789 1.9613186 1.96450273 1.9656042 1.97188361

1.97742397 1.98504488 1.99140795 2.00172873 2.01109235 2.01888599

2.02644784 2.03378743 2.03799618 2.04563352 2.04919014 2.05392673

2.05838353 2.06179702 2.0671854 2.07033172 2.07370971 2.07603291

2.08006416 2.08566999 2.08738161 2.09274942 2.09851505 2.09990483

2.10390672 2.10397893 2.10620687 2.10582546 2.10466569 2.10589808

2.10533571 2.10718959 2.11237832 2.11437905 2.11562614 2.11689546

2.11708236 2.11834969 2.11760625 2.11586864 2.11164027 2.10054772

2.0809425 2.04041142 1.98381991 1.92992984 1.88505849 1.84593935

1.82343383 1.83084069 1.84795147 1.85876716 1.87405178 1.89523444

1.92074721 1.94407866 1.96378097 1.97645568 1.98675253 1.99628601

2.00286959 2.01444721 2.02454476 2.03277954 2.0403758 2.04729916

2.05500061 2.06090015 2.06338122 2.06771387 2.0680788 2.07195268

2.07566014 2.08027906 2.08548841 2.0907379 2.09764394 2.10033329

2.1011381 2.10403309 2.10700739 2.11071993 2.11020562 2.10424985

2.10144263 2.09408463 2.08919355 2.07943815 2.07432151 2.06302054

2.04806157 2.03106379 2.00844553 1.98575995 1.96232295 1.93833986

1.91935138 1.90772317 1.89998981 1.8936662 1.88831396 1.88111043

1.88378007 1.88125716 1.87961019 1.88217571 1.88309681 1.8806157

1.88092711]

Me: [1.34578312 1.49921102 1.62487645 1.69562301 1.73958854 1.76413015

1.7572079 1.78015715 1.83533607 1.85618929 1.85591352 1.87417913

1.88848123 1.89978264 1.92263795 1.94078791 1.9513462 1.96203914

1.97652082 1.98848402 1.99981165 2.01175758 2.02273922 2.03821567

2.05264429 2.07051846 2.0874966 2.10395196 2.11676231 2.12622104

2.13351887 2.13973655 2.14545987 2.14994249 2.15651597 2.16240056

2.16882959 2.17293783 2.17951558 2.18551016 2.19054388 2.19453927

2.19874144 2.20308726 2.20412979 2.20776586 2.20825608 2.20867142

2.21070294 2.20801088 2.20549353 2.20243727 2.20071382 2.1982088

2.19539386 2.19193086 2.19041806 2.188572 2.18501369 2.18082598

2.17622173 2.17417029 2.17001209 2.1717829 2.16625824 2.16395646

2.15644105 2.15119685 2.14473341 2.13787824 2.12853921 2.12093629

2.11074623 2.10281964 2.09315478 2.08320017 2.07297939 2.06275943

2.0503645 2.04093934 2.03206322 2.02332187 2.013899 2.00536526

1.99624678 1.99082287 1.98198051 1.97676487 1.97338222 1.96840373

1.96399403 1.95999847 1.95908499 1.95630345 1.95249866 1.95249866

1.95286532 1.95315893 1.95389231 1.95687648 1.95953211 1.96076466

1.96271575 1.96724727 1.97314052 1.97407104 1.97668976 1.97828932

1.97876531 1.97802536 1.97874713 1.98198051 1.98309892 1.98947379

1.99509687 2.00282919 2.00928318 2.01974724 2.0292363 2.03713097

2.04478787 2.05221687 2.0564756 2.06420123 2.06779789 2.07258673

2.07709157 2.08054111 2.08598506 2.08916308 2.09257448 2.09492026

2.09898999 2.10464777 2.10637488 2.11179017 2.1176049 2.11900623

2.1230407 2.12311349 2.12535915 2.12497473 2.12380574 2.12504792

2.12448109 2.12634958 2.1315781 2.13359373 2.13484999 2.13612853

2.13631679 2.13759322 2.13684445 2.13509426 2.13083449 2.11965442

2.09987659 2.0589191 2.0015865 1.94684894 1.90117908 1.86130351

1.8383402 1.84589943 1.86335578 1.87438509 1.88996488 1.91154292

1.93750938 1.96123267 1.98124763 1.99411422 2.00456144 2.01422961

2.02090362 2.03263503 2.04286114 2.05119686 2.05888307 2.0658857

2.07367228 2.07963483 2.08214182 2.08651889 2.08688752 2.09080015

2.09454389 2.09920691 2.10446453 2.10976105 2.1167265 2.11943822

2.12024964 2.12316809 2.12616596 2.12990717 2.12938894 2.12338658

2.12055666 2.11313693 2.10820304 2.09835807 2.09319225 2.08177738

2.06665669 2.04946038 2.02655447 2.00355458 1.97976707 1.95539962

1.9360894 1.92425691 1.91638478 1.90994598 1.90449508 1.8971571

1.89987678 1.89730658 1.89562862 1.89824237 1.89918073 1.89665306

1.89697033]

Now I just hope that doesn’t screw up everything else.

OK, can take it through to RC and it seems to be working. Wish I could plot it here.

OK, couldn’t plot in spyder and then did a few things to try to. I changed matplotlibrc to default to gtkagg at the backend

I edited ipython\_config.py in 2 profile\_data folders to read

InteractiveShellApp.matplotlib = ‘auto’

InteractiveShellApp.pyplot = ‘auto’

And, from spyder, I typed %matplotlib inline and %pyplot inline (not sure I like it inline but it is happening)

Also, plt.ion() but don’t know if that did anything. The webpage that advised me of all this is saved in bookmarks

Also a saved page on turning matplotlib inline on and off

AHHHHH plt.ioff() is inline off and plt.ion() is inline on!!!

OK, it works well to create rc from the k that is guessed. Now, let’s see if the whole thing works at console.

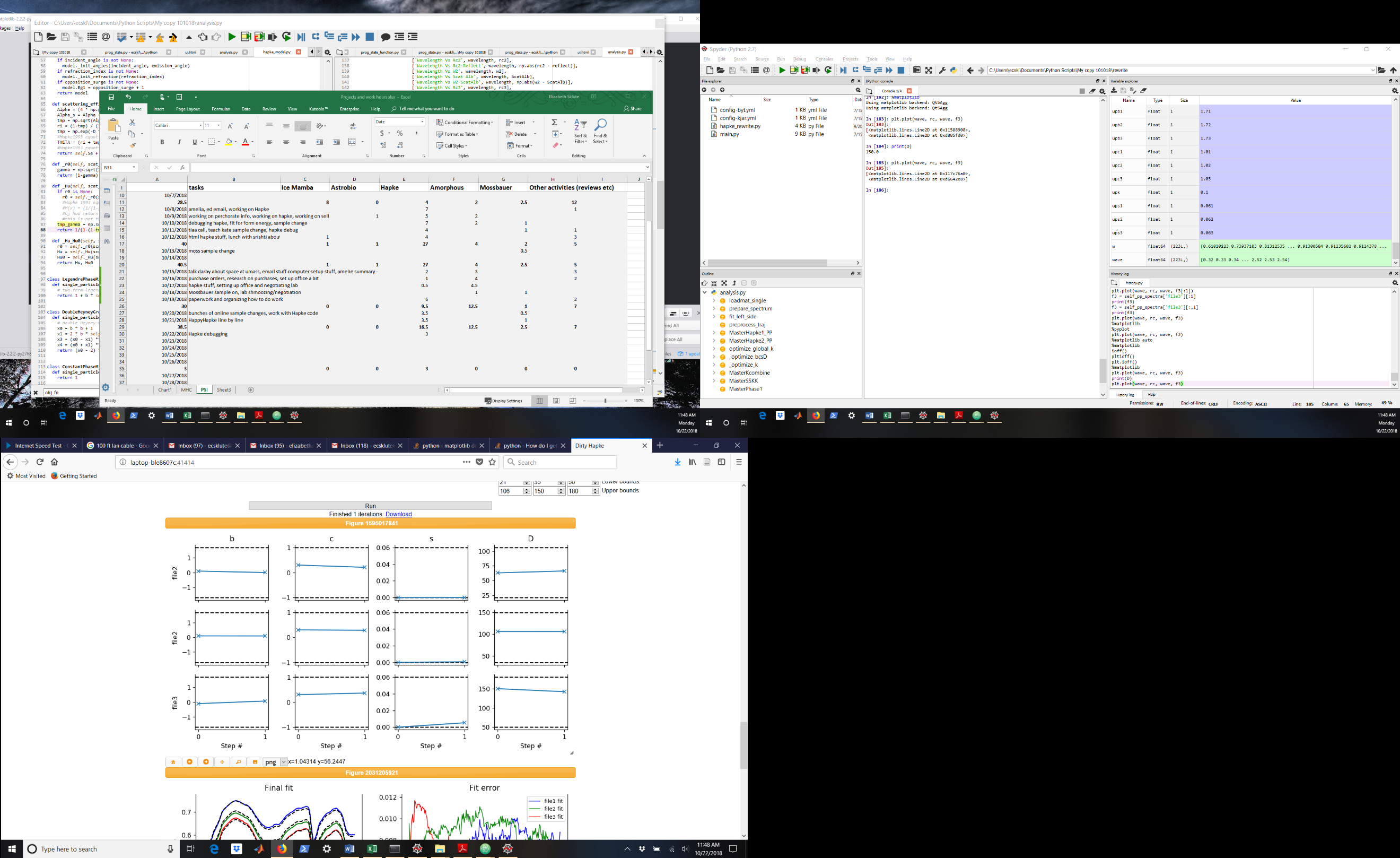
Damn. Broke plotting so web.py won’t launch. Ok, switching back to QT5Agg

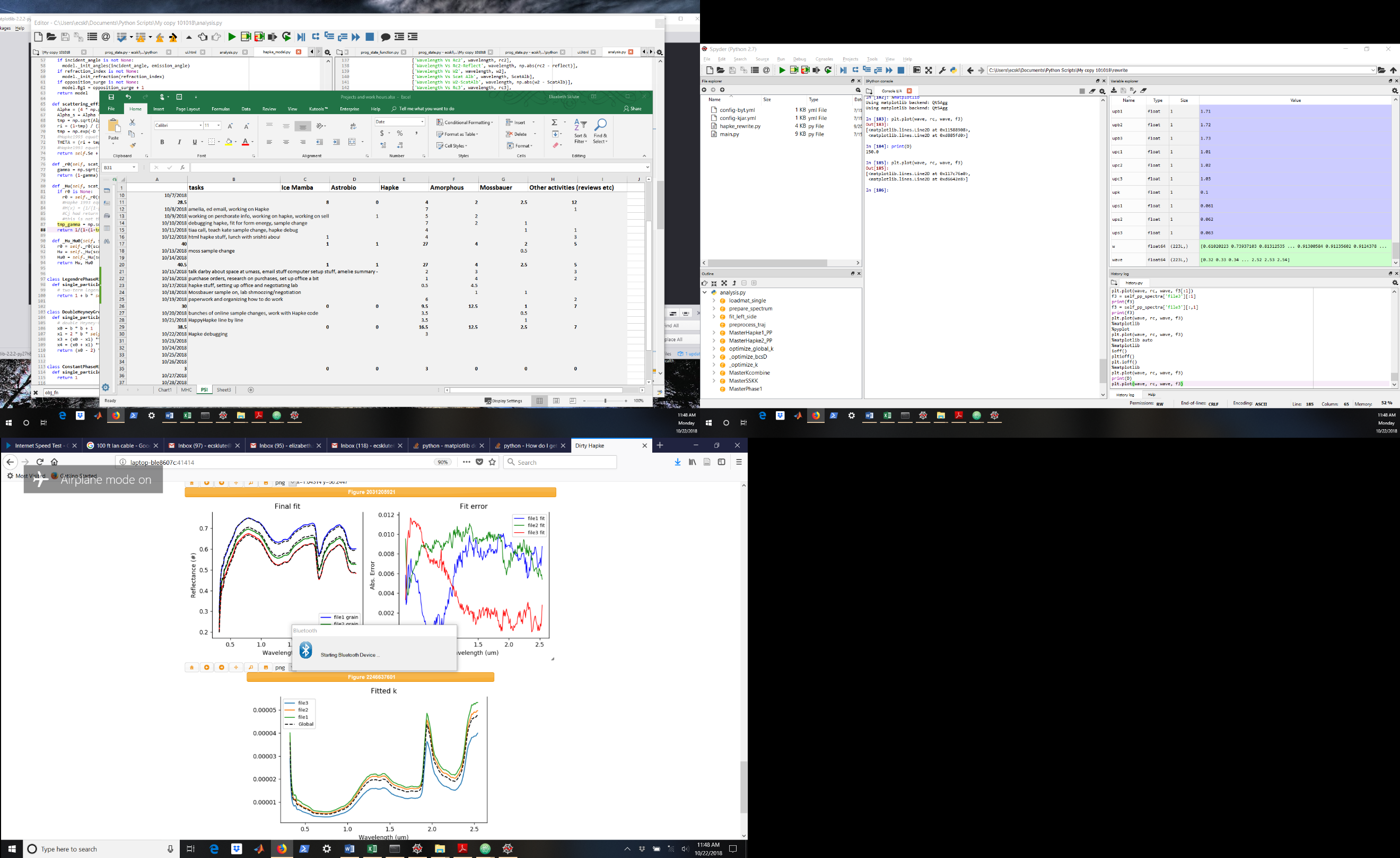
That fixed the problem. Wonder if I can still plot in Spyder – yup! Ok. Good.

OK, problem is local variable gamma referenced before assignment. How do I fix this???

It was an indent problem. I had put it in an if loop by mistake

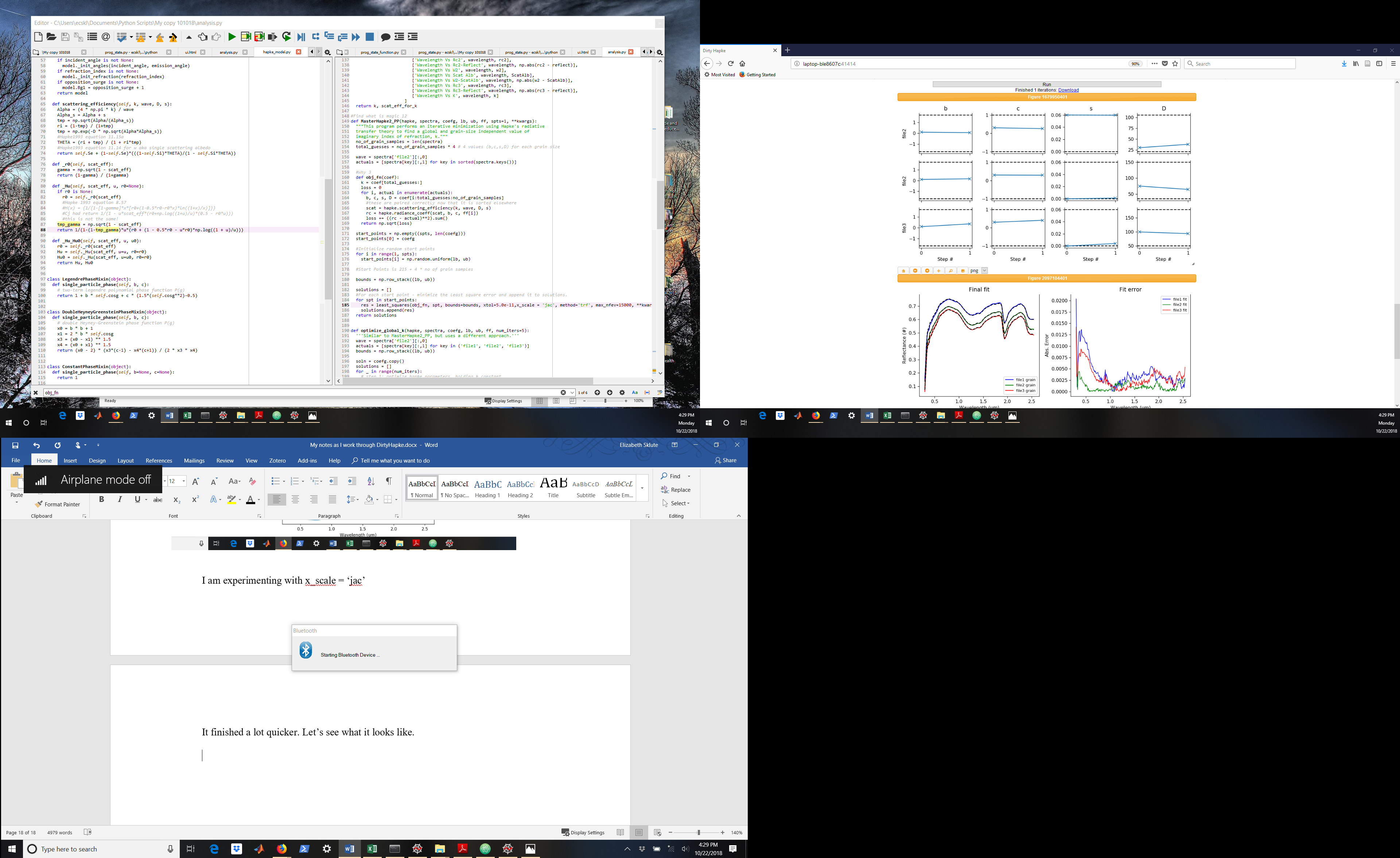
This is working but not finding a great solution. Also the parameters don’t seem to be changing all that much. Maybe the step size is wrong?





I am experimenting with x\_scale = ‘jac’

It finished a lot quicker. Let’s see what it looks like.



Well that’s a bit better. I am going to try bumping down x-tol to see if I can get an even better fit!

Another method that I am not trying but that apparently converges more quickly is method = ‘BFGS’ instead of ‘trf’

OK, it is finishing really quickly now but I am not sure that it is reaching the best answer

Ftol is getting satisfied so we need to shrink that as well. There was a note that xtol was too low. 2.22e-16 is min so I set above that. Set ftol as xtol. It is reaching xtol in only 822 iterations. I guess it is now time to work with the startpoints.

Ran 12 iterations. I do not think it plots the best one, only the last one. It was maxing out on b, c, and s and min on D in almost all cases. Done for the night, will play with it more on Thursday.

TO import – copy paste shift+enter to send class through