Problem 4 (for ASTRON 441 Students)

You will need to install picaso on your own machines following the install instructions here. Note that you will need to download auxiliary files totaling around 6 GB. I recommend you follow the instructions in the link above, but if you are lazy, you can probably get away with running the code below. You will need to run the code below if you run this in Google Colab (note that in Colab, the downloading of files may take >= 1 hour depending on network traffic).

```
!pip install picaso
!git clone https://github.com/natashabatalha/picaso.git
!wget -O picaso/reference/opacities/opacities.db https://zenodo.org/record/3759675/files/opacities.db?download=1

!wget http://ssb.stsci.edu/trds/tarfiles/synphot3.tar.gz
!tar -xvf synphot3.tar.gz
!mv grp/redcat/trds/grid grid

import os
os.environ['picaso_refdata'] = os.path.join(os.getcwd(), 'picaso', 'reference')
os.environ['PYSYN_CDBS'] = os.getcwd()
print(os.environ['picaso_refdata'])
```

```
     grp/redcat/trds/grid/ck04models/ckp05/ckp05_12500.fits
     grp/redcat/trds/grid/ck04models/ckp05/ckp05_6250.fits
     grp/redcat/trds/grid/ck04models/ckp05/ckp05_13000.fits
     grp/redcat/trds/grid/ck04models/ckp05/ckp05_11250.fits
     grp/redcat/trds/grid/ck04models/ckp05/ckp05_21000.fits
     grp/redcat/trds/grid/ck04models/ckp05/ckp05_11000.fits
     grp/redcat/trds/grid/ck04models/ckp05/ckp05_11750.fits
     grp/redcat/trds/grid/ck04models/ckp05/ckp05_8250.fits
     grp/redcat/trds/grid/ck04models/ckp05/ckp05_9250.fits
     grp/redcat/trds/grid/ck04models/ckp05/ckp05_19000.fits
```

```
grp/redcat/trds/grid/ck04models/ckp05/ckp05_6000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_9000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_35000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_47000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_40000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_41000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_8750.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_6500.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_12750.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_26000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_4750.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_39000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_25000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_7500.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_45000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_46000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_33000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_20000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_50000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_4250.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_15000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_12250.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_14000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_16000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_22000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_9500.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_5250.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_17000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_7750.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_10250.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_23000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_30000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_24000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_10000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_11500.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_34000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_49000.fits
```

```
grp/redcat/trds/grid/ck04models/ckp05/ckp05_18000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_6750.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_5750.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_36000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_10500.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_43000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_27000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_48000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_44000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_29000.fits
grp/redcat/trds/grid/ck04models/ckp05/ckp05_5500.fits
/content/picaso/reference
```

We also need to download a few other auxiliary files for this assignment. Step 1: we need to download [these Sonora models](), untar them, stick them in a folder, and set a variable to point to that folder.

```
!wget -O $picaso_refdata/sonora_profile.tar https://zenodo.org/record/1309035/fil
!mkdir $picaso_refdata/sonora_profile
!tar -xvf $picaso_refdata/sonora_profile.tar -C $picaso_refdata/sonora_profile
import os
sonora_profile_db = os.path.join(os.environ['picaso_refdata'], 'sonora_profile')
```

```
t700g100nc_m0.0.cmp.gz
t700g1780nc_m0.0.cmp.gz
t700g178nc_m0.0.cmp.gz
t700g17nc_m0.0.cmp.gz
t700g3160nc_m0.0.cmp.gz
t700g316nc_m0.0.cmp.gz
t700g31nc_m0.0.cmp.gz
t700g562nc_m0.0.cmp.gz
t700g56nc_m0.0.cmp.gz
t750g1000nc_m0.0.cmp.gz
t750g100nc_m0.0.cmp.gz
t750g1780nc_m0.0.cmp.gz
```

```
t750g178nc_m0.0.cmp.gz
t750g17nc_m0.0.cmp.gz
t750g3160nc_m0.0.cmp.gz
t750g316nc_m0.0.cmp.gz
t750g31nc_m0.0.cmp.gz
t750g562nc_m0.0.cmp.gz
t750g56nc_m0.0.cmp.gz
t800g1000nc_m0.0.cmp.gz
t800g100nc_m0.0.cmp.gz
t800g1780nc_m0.0.cmp.gz
t800g178nc_m0.0.cmp.gz
t800g17nc_m0.0.cmp.gz
t800g3160nc_m0.0.cmp.gz
t800g316nc_m0.0.cmp.gz
t800g31nc_m0.0.cmp.gz
t800g562nc_m0.0.cmp.gz
t800g56nc_m0.0.cmp.gz
t850g1000nc_m0.0.cmp.gz
t850g100nc_m0.0.cmp.gz
t850g1780nc_m0.0.cmp.gz
t850g178nc_m0.0.cmp.gz
t850g17nc_m0.0.cmp.gz
t850g3160nc_m0.0.cmp.gz
t850g316nc_m0.0.cmp.gz
t850g31nc_m0.0.cmp.gz
t850g562nc_m0.0.cmp.gz
t850g56nc_m0.0.cmp.gz
t900g1000nc_m0.0.cmp.gz
t900g100nc_m0.0.cmp.gz
t900g1780nc_m0.0.cmp.gz
t900g178nc_m0.0.cmp.gz
t900g17nc_m0.0.cmp.gz
t900g3160nc_m0.0.cmp.gz
t900g316nc_m0.0.cmp.gz
t900g31nc_m0.0.cmp.gz
t900g562nc_m0.0.cmp.gz
t900g56nc_m0.0.cmp.gz
t950g1000nc_m0.0.cmp.gz
```

```
t950g100nc_m0.0.cmp.gz
t950g1780nc_m0.0.cmp.gz
t950g178nc_m0.0.cmp.gz
t950g17nc_m0.0.cmp.gz
t950g3160nc_m0.0.cmp.gz
t950g316nc_m0.0.cmp.gz
t950g31nc_m0.0.cmp.gz
t950g562nc_m0.0.cmp.gz
t950g56nc_m0.0.cmp.gz
```

Next we need to download [these cloud opacity files](), unzip them, stick them in a folder, and set a variable to point to that folder.

```
!wget -O $picaso_refdata/virga.zip https://zenodo.org/record/3992294/files/virga.
!mkdir $picaso_refdata/virga
!unzip $picaso_refdata/virga.zip -d $picaso_refdata/virga
import os
virga_dir = os.path.join(os.environ['picaso_refdata'], 'virga','virga_1e-7')
```

```
--2023-04-25 00:43:47--  https://zenodo.org/record/3992294/files/virga.zip?download=1
Resolving zenodo.org (zenodo.org)... 188.185.124.72
Connecting to zenodo.org (zenodo.org)|188.185.124.72|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4007293 (3.8M) [application/octet-stream]
Saving to: ‘/content/picaso/reference/virga.zip’

/content/picaso/ref 100%[===================>]   3.82M   267KB/s    in 15s

2023-04-25 00:44:03 (262 KB/s) - ‘/content/picaso/reference/virga.zip’ saved [4007293/4007293]

Archive:  /content/picaso/reference/virga.zip
   creating: /content/picaso/reference/virga/virga_1e-7/
  inflating: /content/picaso/reference/virga/virga_1e-7/H2O.refrind
  inflating: /content/picaso/reference/virga/virga_1e-7/Fe.refrind
   creating: /content/picaso/reference/virga/__MACOSX/
```

```
 creating: /content/picaso/reference/virga/__MACOSX/virga_1e-7/
inflating: /content/picaso/reference/virga/__MACOSX/virga_1e-7/._Fe.refrind
inflating: /content/picaso/reference/virga/virga_1e-7/NH3.refrind
inflating: /content/picaso/reference/virga/virga_1e-7/H2O.mieff
inflating: /content/picaso/reference/virga/virga_1e-7/MgSiO3.mieff
inflating: /content/picaso/reference/virga/virga_1e-7/TiO2.mieff
inflating: /content/picaso/reference/virga/virga_1e-7/CH4.mieff
inflating: /content/picaso/reference/virga/virga_1e-7/Na2S.mieff
inflating: /content/picaso/reference/virga/virga_1e-7/Al2O3.refrind
inflating: /content/picaso/reference/virga/virga_1e-7/MnS.refrind
inflating: /content/picaso/reference/virga/virga_1e-7/KCl.refrind
inflating: /content/picaso/reference/virga/__MACOSX/virga_1e-7/._KCl.refrind
inflating: /content/picaso/reference/virga/virga_1e-7/MgSiO3.refrind
inflating: /content/picaso/reference/virga/virga_1e-7/Fe.mieff
inflating: /content/picaso/reference/virga/virga_1e-7/Cr.mieff
inflating: /content/picaso/reference/virga/virga_1e-7/Na2S.refrind
inflating: /content/picaso/reference/virga/virga_1e-7/TiO2.refrind
inflating: /content/picaso/reference/virga/__MACOSX/virga_1e-7/._TiO2.refrind
inflating: /content/picaso/reference/virga/virga_1e-7/NH3.mieff
inflating: /content/picaso/reference/virga/virga_1e-7/CH4.refrind
inflating: /content/picaso/reference/virga/__MACOSX/virga_1e-7/._CH4.refrind
inflating: /content/picaso/reference/virga/virga_1e-7/Cr.refrind
inflating: /content/picaso/reference/virga/virga_1e-7/Al2O3.mieff
inflating: /content/picaso/reference/virga/virga_1e-7/KCl.mieff
inflating: /content/picaso/reference/virga/virga_1e-7/MnS.mieff
inflating: /content/picaso/reference/virga/virga_1e-7/ZnS.refrind
inflating: /content/picaso/reference/virga/__MACOSX/virga_1e-7/._ZnS.refrind
inflating: /content/picaso/reference/virga/virga_1e-7/Mg2SiO4.refrind
inflating: /content/picaso/reference/virga/virga_1e-7/ZnS.mieff
inflating: /content/picaso/reference/virga/virga_1e-7/Mg2SiO4.mieff
inflating: /content/picaso/reference/virga/__MACOSX/virga_1e-7/._Mg2SiO4.mieff
```

Skip to here once you are done installing

Import packages

```
import numpy as np
import picaso
import matplotlib.pylab as plt
%matplotlib inline
```

```
import warnings
warnings.filterwarnings('ignore')
#picaso
from picaso import justdoit as jdi
from picaso import justplotit as jpi

#plotting
jpi.output_notebook()
```

Define Planetary Parameters Here

```
# define bulk planet properties here
planet_temp = 1400
cloudy = False
```

Simulate Spectra

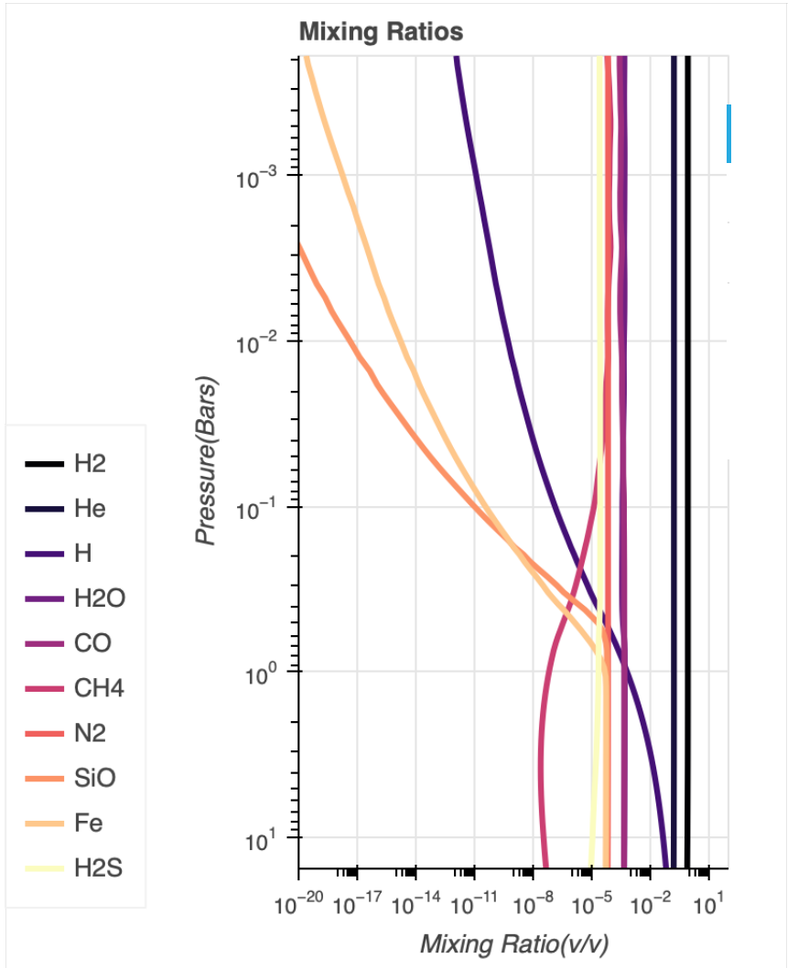
```
# load opacity database
opa = jdi.opannection(wave_range=[1,5])

# create a new case to simulate
case1 = jdi.inputs()

case1.phase_angle(0) # we are not dealing with reflected light, so we can skip this

# here we are going to have to specify gravity through R and M since we need it in the Flux calc
case1.gravity(mass=1, mass_unit=jdi.u.Unit('M_jup'),
              radius=1., radius_unit=jdi.u.Unit('R_jup'))

# set the star so the code doesn't crash. We are doing thermal emission spectra and don't actually use the star in any way.
case1.star(opa, 5800,0.0122,4.437,radius=1, radius_unit = jdi.u.Unit('R_sun') )

# load in the atmospheric parameters given the effective temperature we chose
# someone did all the hard work to figure this out already
case1.sonora(sonora_profile_db,  planet_temp)

# if cloudy, turn on clousd
if cloudy:
    metallicity = 1 #1xSolar
    mean_molecular_weight = 2.2
    fsed=1
    gas_condensates = ['H2O','MnS','Mg2SiO4','Al2O3']

    #for the cloud code we have to supply a kzz value, which describes the degree of mixing
    p=case1.inputs['atmosphere']['profile']['pressure']
```

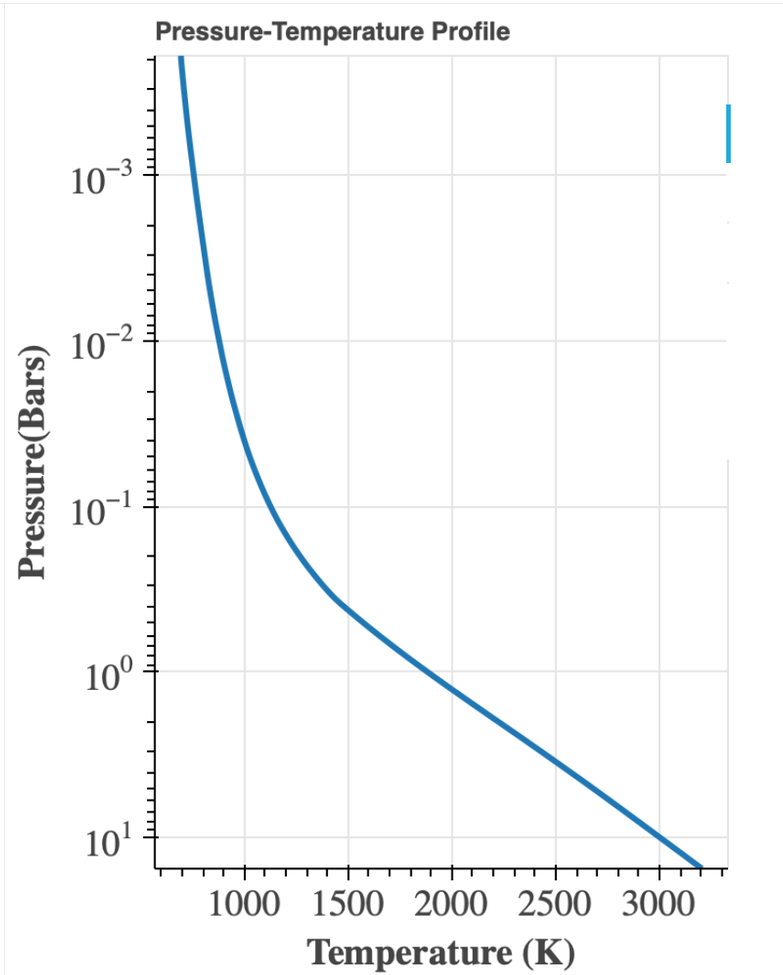
```
    case1.inputs['atmosphere']['profile']['kz'] = [1e9]*len(p)

    case1.virga(gas_condensates, virga_dir, fsed=fsed,mh=metallicity,
                mmw = mean_molecular_weight,full_output=True)

# simulate the spectrum and store in a dataframe
df = case1.spectrum(opa, full_output=True,calculation='thermal')

# pull out some useful parameters
wno, fpfs , fp = df['wavenumber'] , df['fpfs_thermal'], df['thermal']
wno_bin, fpfs_bin = jdi.mean_regrid(wno, fpfs, R=500) # downsample the spectrum
wno_bin, fp_bin = jdi.mean_regrid(wno, fp, R=500) # downsample the spectrum
full_output = df['full_output']
```

Produce plots

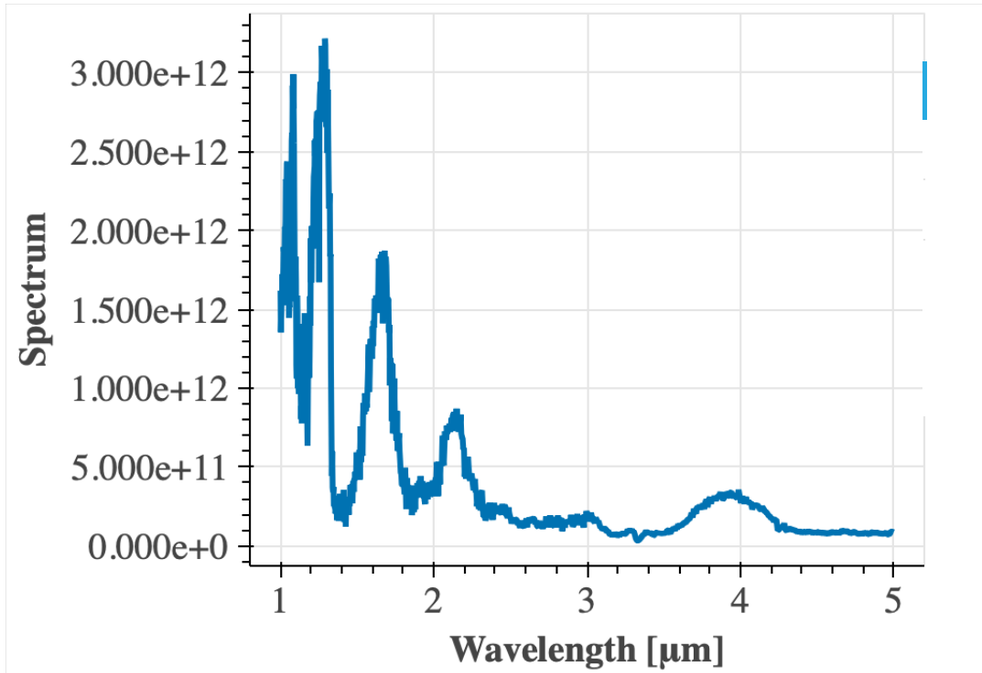
Use these plots to answer your homework.

```
# Left: plot the pressure vs mixing ratio for various constitutents in the atmosp
# Right: plot the pressure-temperature profile

jpi.show(jpi.row(
    jpi.mixing_ratio(full_output, plot_height=500, limit=10),
    jpi.pt(full_output, plot_height=500)))
```


Mixing Ratios
H2
He
H
H2O
CO
CH4
N2
SiO
Fe
H2S
Pressure(Bars)
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{-20} 10^{-17} 10^{-14} 10^{-11} 10^{-8} 10^{-5} 10^{-2} 10^{1}
Mixing Ratio(v/v)

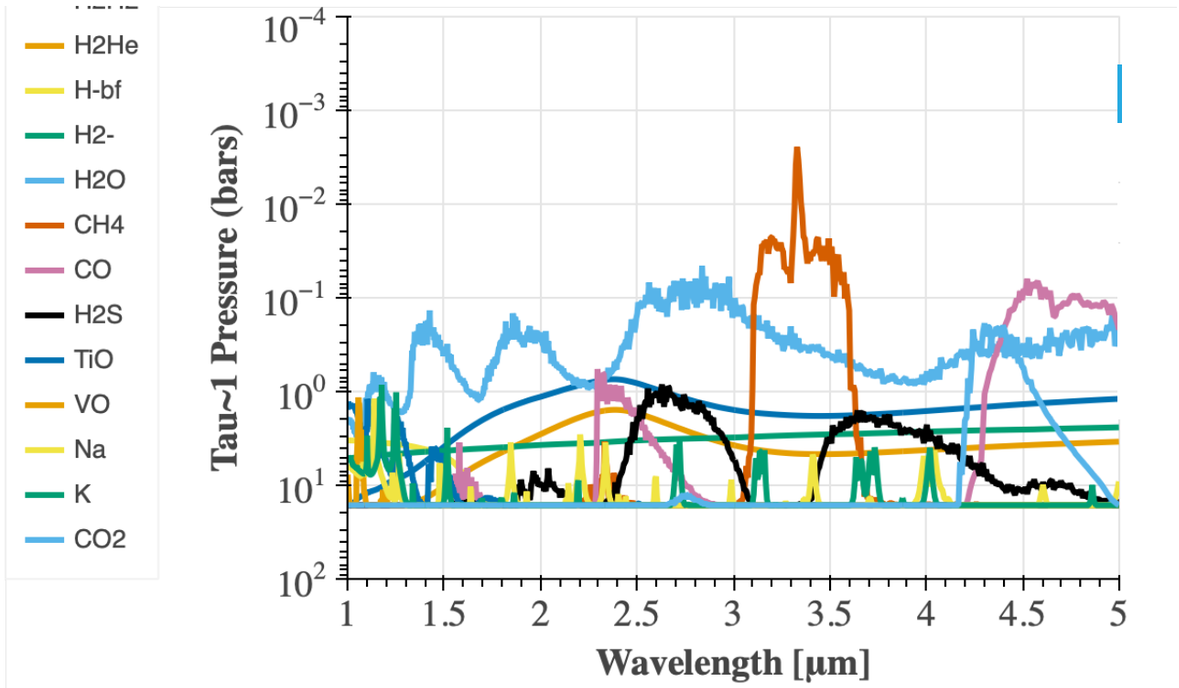

Pressure-Temperature Profile
Pressure(Bars)
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
1000 1500 2000 2500 3000
Temperature (K)

```
# Plot the actual spectrum. Zoom in to look at features carefully!

jpi.show(jpi.spectrum(wno_bin,fp_bin,plot_width=500))
```

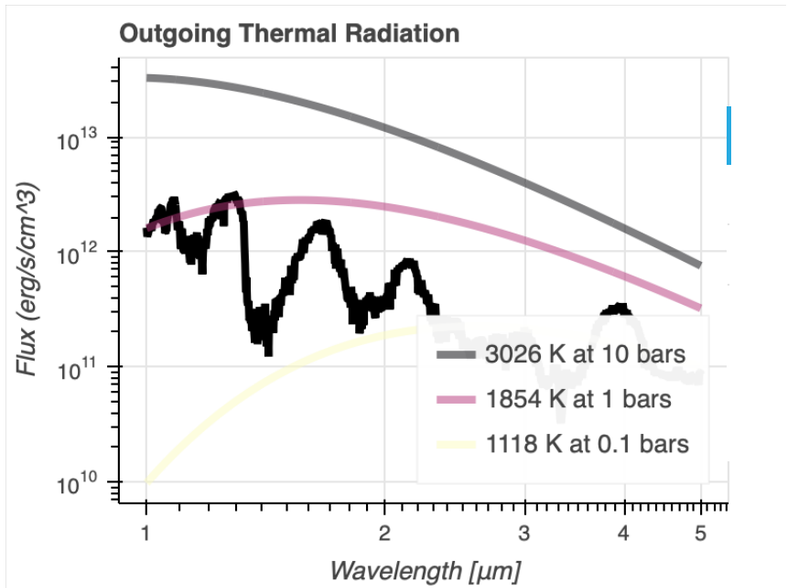



```
# Plot the pressure correspoding to tau=1 for photons that interact with each ind
# The deeper the tau=1 surface, the less absorption there is from that species at

contributors = jdi.get_contribution(case1, opa, at_tau=1)

contrib_species = contributors['tau_p_surface'].keys()

wno=[]
spec=[]
```

```
labels=[]
for j in contrib_species:
    x,y = jdi.mean_regrid(opa.wno, contributors['tau_p_surface'][j], R=500)
    if np.min(y)<5: # Bars
        wno+=[x]
        spec+=[y]
        labels +=[j]

fig = jpi.spectrum(wno,spec,plot_width=600,plot_height=350,y_axis_label='Tau~1 Pr
                   y_axis_type='log',x_range=[1,5],
                     y_range=[1e2,1e-4],legend=labels)

jpi.show(fig)
```



```
# Compare the spectrum with blackbody spectra.
# Each blackbody spectrum is labeled with the temperature and the corresponding p
fig = jpi.flux_at_top(df, pressures=[10,1,0.1],R=500)
fig.legend.location='bottom_right'
jpi.show(fig)
```



```
# For cloudy spectra, this plots the opacity as a function of wavelength and pres
if cloudy:
    fig, ax = plt.subplots(ncols=2,figsize=(15,5))

    for it, itau in enumerate(['taugas','taucld']):

        tau_bin = []
        for i in range(df['full_output'][itau].shape[0]):
            x,y = jdi.mean_regrid(df['wavenumber'],
                                  df['full_output'][itau][i,:,0], R=150)
            tau_bin += [[y]]

        tau_bin = np.array(np.log10(tau_bin))[:,0,:]
        X,Y = np.meshgrid(1e4/x,df['full_output']['layer']['pressure'])
        Z = tau_bin
        pcm=ax[it].pcolormesh(X, Y, Z)
        cbar=fig.colorbar(pcm, ax=ax[it])
        pcm.set_clim(-3.0, 3.0)
        ax[it].set_title(itau)
        ax[it].set_yscale('log')
        ax[it].set_ylim([1e2,1e-3])
        ax[it].set_ylabel('Pressure(bars)')
        ax[it].set_xlabel('Wavelength(um)')
        cbar.set_label('log Opacity')
```

Colab paid products - Cancel contracts here

✓ 0s completed at 7:53 PM