Using_pyCloudy_1

June 14, 2017

In [1]:
```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
import pyCloudy as pc
```

In [3]:
```
# Define verbosity to high level (will print errors, warnings and messages)
pc.log_.level = 3
```

In [4]:
```
# The directory in which we will have the model
# You may want to change this to a different place so that the current directory
# will not receive all the Cloudy files.
dir_ = './'
```

In [5]:
```
# Define some parameters of the model:
model_name = 'model_1'
full_model_name = '{0}{1}'.format(dir_, model_name)
dens = 2. #log cm-3
Teff = 45000. #K
qH = 47. #s-1
r_min = 5e17 #cm
dist = 1.26 #kpc
```

In [6]:
```
# these are the commands common to all the models (here only one ...)
options = ('no molecules',
           'no level2 lines',
           'no fine opacities',
           'atom h-like levels small',
           'atom he-like levels small',
           'COSMIC RAY BACKGROUND',
           'element limit off -8',
           'print line optical depth',
           )
```

In [7]:
```
emis_tab_c13 = ['H  1  4861',
                'H  1  6563',
                'He 1  5876',
                'N  2  6584',
                'O  1  6300',
                'O II  3726',
                'O II  3729',
                'O  3  5007',
                'TOTL  4363',
                'S II  6716',
                'S II 6731',
```

```
                    'Cl 3 5518',
                    'Cl 3 5538',
                    'O  1 63.17m',
                    'O  1 145.5m',
                    'C  2 157.6m']
```

In [8]:
```
emis_tab = ['H  1  4861.36A',
            'H  1  6562.85A',
            'Ca B  5875.64A',
            'N  2  6583.45A',
            'O  1  6300.30A',
            'O  2  3726.03A',
            'O  2  3728.81A',
            'O  3  5006.84A',
            'BLND  4363.00A',
            'S  2  6716.44A',
            'S  2  6730.82A',
            'Cl 3  5517.71A',
            'Cl 3  5537.87A',
            'O  1  63.1679m',
            'O  1  145.495m',
            'C  2  157.636m']
```

In [9]:
```
abund = {'He' : -0.92, 'C' : 6.85 - 12, 'N' : -4.0, 'O' : -3.40, 'Ne' : -4.00,
         'S' : -5.35, 'Ar' : -5.80, 'Fe' : -7.4, 'Cl' : -7.00}
```

In [10]:
```
# Defining the object that will manage the input file for Cloudy
c_input = pc.CloudyInput(full_model_name)
```

In [11]:
```
# Filling the object with the parameters
# Defining the ionizing SED: Effective temperature and luminosity.
# The lumi_unit is one of the Cloudy options, like "luminosity solar", "q(H)", "ionization par
c_input.set_BB(Teff = Teff, lumi_unit = 'q(H)', lumi_value = qH)
```

In [12]:
```
# Defining the density. You may also use set_dlaw(parameters) if you have a density law define
c_input.set_cste_density(dens)
```

In [13]:
```
# Defining the inner radius. A second parameter would be the outer radius (matter-bounded nebu
c_input.set_radius(r_in=np.log10(r_min))
c_input.set_abund(ab_dict = abund, nograins = True)
c_input.set_other(options)
c_input.set_iterate() # (0) for no iteration, () for one iteration, (N) for N iterations.
c_input.set_sphere() # () or (True) : sphere, or (False): open geometry.
c_input.set_emis_tab(emis_tab) # better use read_emis_file(file) for long list of lines, where
c_input.set_distance(dist=dist, unit='kpc', linear=True) # unit can be 'kpc', 'Mpc', 'parsecs'
```

In [14]:
```
# Writing the Cloudy inputs. to_file for writing to a file (named by full_model_name). verbose
c_input.print_input(to_file = True, verbose = False)
```

```
CloudyInput: Input writen in ./model_1.in
```

In [15]:
```
# Printing some message to the screen
pc.log_.message('Running {0}'.format(model_name), calling = 'test1')
```

```
test1: Running model_1
```

```
In [16]: # Tell pyCloudy where your cloudy executable is:
         pc.config.cloudy_exe = '/usr/local/Cloudy/c17.00/source/cloudy.exe'

_Config: cloudy_exe set to /usr/local/Cloudy/c17.00/source/cloudy.exe

In [17]: # Running Cloudy with a timer. Here we reset it to 0.
         pc.log_.timer('Starting Cloudy', quiet = True, calling = 'test1')
         c_input.run_cloudy()
         pc.log_.timer('Cloudy ended after seconds:', calling = 'test1')

run_cloudy: running: cd . ;  /usr/local/Cloudy/c17.00/source/cloudy.exe
     run_cloudy: ending: cd . ;  /usr/local/Cloudy/c17.00/source/cloudy.exe
   test1: Cloudy ended after seconds: in 31.47327160835266

In [18]: # Reading the Cloudy outputs in the Mod CloudyModel object
         Mod = pc.CloudyModel(full_model_name)

CloudyModel ./model_1: Creating CloudyModel for ./model_1
     CloudyModel ./model_1: Li abundance not defined
     CloudyModel ./model_1: Be abundance not defined
     CloudyModel ./model_1: B abundance not defined
     CloudyModel ./model_1: Sc abundance not defined
     CloudyModel ./model_1: ./model_1.rad read
     CloudyModel ./model_1: Number of zones: 125
     CloudyModel ./model_1: ./model_1.phy read
     CloudyModel ./model_1: ./model_1.ele_H read
     CloudyModel ./model_1: filling H with 3 columns
     CloudyModel ./model_1: ./model_1.ele_He read
     CloudyModel ./model_1: filling He with 3 columns
     CloudyModel ./model_1: ./model_1.ele_C read
     CloudyModel ./model_1: filling C with 13 columns
     CloudyModel ./model_1: ./model_1.ele_N read
     CloudyModel ./model_1: filling N with 8 columns
     CloudyModel ./model_1: ./model_1.ele_O read
     CloudyModel ./model_1: filling O with 12 columns
     CloudyModel ./model_1: ./model_1.ele_Ne read
     CloudyModel ./model_1: filling Ne with 11 columns
     CloudyModel ./model_1: ./model_1.ele_Ar read
     CloudyModel ./model_1: filling Ar with 19 columns
     CloudyModel ./model_1: ./model_1.ele_S read
     CloudyModel ./model_1: filling S with 17 columns
     CloudyModel ./model_1: ./model_1.ele_Cl read
     CloudyModel ./model_1: filling Cl with 18 columns
     CloudyModel ./model_1: ./model_1.ele_Fe read
     CloudyModel ./model_1: filling Fe with 27 columns
     CloudyModel ./model_1: ./model_1.ele_Si read
     CloudyModel ./model_1: filling Si with 15 columns
     CloudyModel ./model_1: ./model_1.emis read
     CloudyModel ./model_1: Number of emissivities: 16
     CloudyModel ./model_1: ./model_1.cont read

In [19]: # Use TAB to know all the methods and variables for CloudyModel class
         # Mod.TAB
         dir(Mod) # This is the online answering way
         # Description of this class is available here: http://pythonhosted.org//pyCloudy/classpy_cloud
```

```
Out[19]: ['C3D_comments',
          'H0_mass',
          'H_mass',
          'H_mass_cut',
          'H_mass_full',
          'Hbeta',
          'Hbeta_cut',
          'Hbeta_full',
          'Hp_mass',
          'Phi',
          'Phi0',
          'Q',
          'Q0',
          'T0',
          'Teff',
          '_CloudyModel__H_mass_cut',
          '_CloudyModel__r_in_cut',
          '_CloudyModel__r_out_cut',
          '_CloudyModel__r_range',
          '__class__',
          '__delattr__',
          '__dict__',
          '__dir__',
          '__doc__',
          '__eq__',
          '__format__',
          '__ge__',
          '__getattribute__',
          '__gt__',
          '__hash__',
          '__init__',
          '__init_subclass__',
          '__le__',
          '__lt__',
          '__module__',
          '__ne__',
          '__new__',
          '__reduce__',
          '__reduce_ex__',
          '__repr__',
          '__setattr__',
          '__sizeof__',
          '__str__',
          '__subclasshook__',
          '__weakref__',
          '_get_H_mass_cut',
          '_get_Hbeta_cut',
          '_get_over_range',
          '_get_r_in_cut',
          '_get_r_out_cut',
          '_i_emis',
          '_i_line',
          '_init_all2zero',
          '_init_cont',
```

```
'_init_emis',
'_init_grains',
'_init_heatcool',
'_init_ionic',
'_init_lin',
'_init_opd',
'_init_phy',
'_init_rad',
'_l_emis',
'_quiet_div',
'_r_out_cut_doc',
'_read_stout',
'_res',
'_set_H_mass_cut',
'_set_Hbeta_cut',
'_set_r_in_cut',
'_set_r_out_cut',
'aborted',
'abund',
'add_emis_from_pyneb',
'calling',
'cautions',
'cloudy_version',
'cloudy_version_major',
'comments',
'cool',
'cool_full',
'date_model',
'depth',
'depth_full',
'distance',
'dr',
'dr_full',
'drff',
'dv',
'dv_full',
'dvff',
'emis_from_pyneb',
'emis_full',
'emis_is_log',
'emis_labels',
'emis_labels_13',
'emis_labels_17',
'empty_model',
'ff',
'ff_full',
'gabund',
'gabund_full',
'gabund_labels',
'gas_mass_per_H',
'gasize',
'gdgrat',
'gdgrat_full',
'gdgrat_labels',
```

```
'gdsize',
'get_EW',
'get_EW2',
'get_G0',
'get_Ha_EW',
'get_Hb_EW',
'get_Hb_SB',
'get_T0_emis',
'get_T0_emis_rad',
'get_T0_ion_rad',
'get_T0_ion_rad_ne',
'get_T0_ion_vol',
'get_T0_ion_vol_ne',
'get_ab_ion_rad',
'get_ab_ion_rad_ne',
'get_ab_ion_vol',
'get_ab_ion_vol_ne',
'get_cont_x',
'get_cont_y',
'get_emis',
'get_emis_rad',
'get_emis_vol',
'get_ionic',
'get_line',
'get_ne_emis',
'get_ne_ion_rad_ne',
'get_ne_ion_vol_ne',
'get_t2_emis',
'get_t2_ion_rad_ne',
'get_t2_ion_vol_ne',
'gmass',
'gsize',
'gtemp',
'gtemp_full',
'gtemp_labels',
'heat',
'heat_full',
'info',
'intens',
'ionic_full',
'ionic_names',
'is_valid_ion',
'line_is_log',
'lines',
'liste_elem',
'log_',
'log_U',
'log_U_mean',
'log_U_mean_ne',
'model_name',
'model_name_s',
'nH',
'nH_full',
'nH_mean',
```

```
 'nHff_full',
 'n_elements',
 'n_emis',
 'n_gabund',
 'n_gdgrat',
 'n_gtemp',
 'n_ions',
 'n_lines',
 'n_zones',
 'n_zones_full',
 'ne',
 'ne_full',
 'nenH',
 'nenH_full',
 'nenHff2_full',
 'opd_absorp',
 'opd_energy',
 'opd_scat',
 'opd_total',
 'out',
 'out_exists',
 'phi',
 'plan_par',
 'plot_spectrum',
 'print_lines',
 'print_stats',
 'r_in',
 'r_in_cut',
 'r_out',
 'r_out_cut',
 'r_range',
 'rad_integ',
 'rad_mean',
 'radius',
 'radius_full',
 'read_outputs',
 'rlines',
 'slines',
 't2',
 'te',
 'te_full',
 'tenenH',
 'tenenH_full',
 'theta',
 'thickness',
 'thickness_full',
 'vol_integ',
 'vol_mean',
 'warnings',
 'zones',
 'zones_full']
```

In [20]:
```
Mod.print_stats()
```

```
Name of the model: ./model_1
```

```
 R_in (cut) = 5.000e+17 (5.000e+17), R_out (cut) = 1.926e+18 (1.926e+18)
 H+ mass = 2.32e+00, H mass = 2.47e+00
 <H+/H> = 0.97, <He++/He> = 0.00, <He+/He> = 0.86
 <O+++/O> = 0.00, <O++/O> = 0.29, <O+/O> = 0.67
 <N+++/O> = 0.00, <N++/O> = 0.40, <N+/O> = 0.58
 T(O+++) = 7543, T(O++) = 7330, T(O+) = 7661
 <ne> = 104,  <nH> = 100, T0 = 7570, t2=0.0024
 <log U> = -2.79
```

```
In [21]: Mod.print_lines()
```

```
H__1_486136A 4.640013e+34
H__1_656285A 1.269049e+35
CA_B_587564A 7.194442e+33
N__2_658345A 6.205508e+34
O__1_630030A 1.318442e+33
O__2_372603A 4.102699e+34
O__2_372881A 5.519847e+34
O__3_500684A 4.796250e+34
BLND_436300A 9.520116e+31
S__2_671644A 7.473555e+33
S__2_673082A 5.712804e+33
CL_3_551771A 1.170528e+32
CL_3_553787A 8.356462e+31
O__1_631679M 8.674026e+32
O__1_145495M 8.461688e+31
C__2_157636M 1.682068e+32
```

```
In [22]: Mod.get_ab_ion_vol_ne('O',2)
```

```
Out[22]: 0.29376349482467973
```

```
In [23]: Mod.get_T0_ion_vol_ne('O', 2)
```

```
Out[23]: 7329.8824323550098
```

```
In [24]: Mod.log_U_mean
```

```
Out[24]: -2.7909736077998555
```

```
In [25]: Mod.log_U_mean_ne
```

```
Out[25]: -2.7732943196565465
```

```
In [26]: print('T0 = {0:7.1f}K, t2 = {1:6.4f}'.format(Mod.T0, Mod.t2))
```

```
T0 =  7570.2K, t2 = 0.0024
```

```
In [27]: print('Hbeta Equivalent width = {0:6.1f}, Hbeta Surface Brightness = {1:4.2e}'.format(Mod.get_
```

```
Hbeta Equivalent width = -715.5, Hbeta Surface Brightness = 9.36e-14
```

```
In [28]: Mod.emis_labels
```

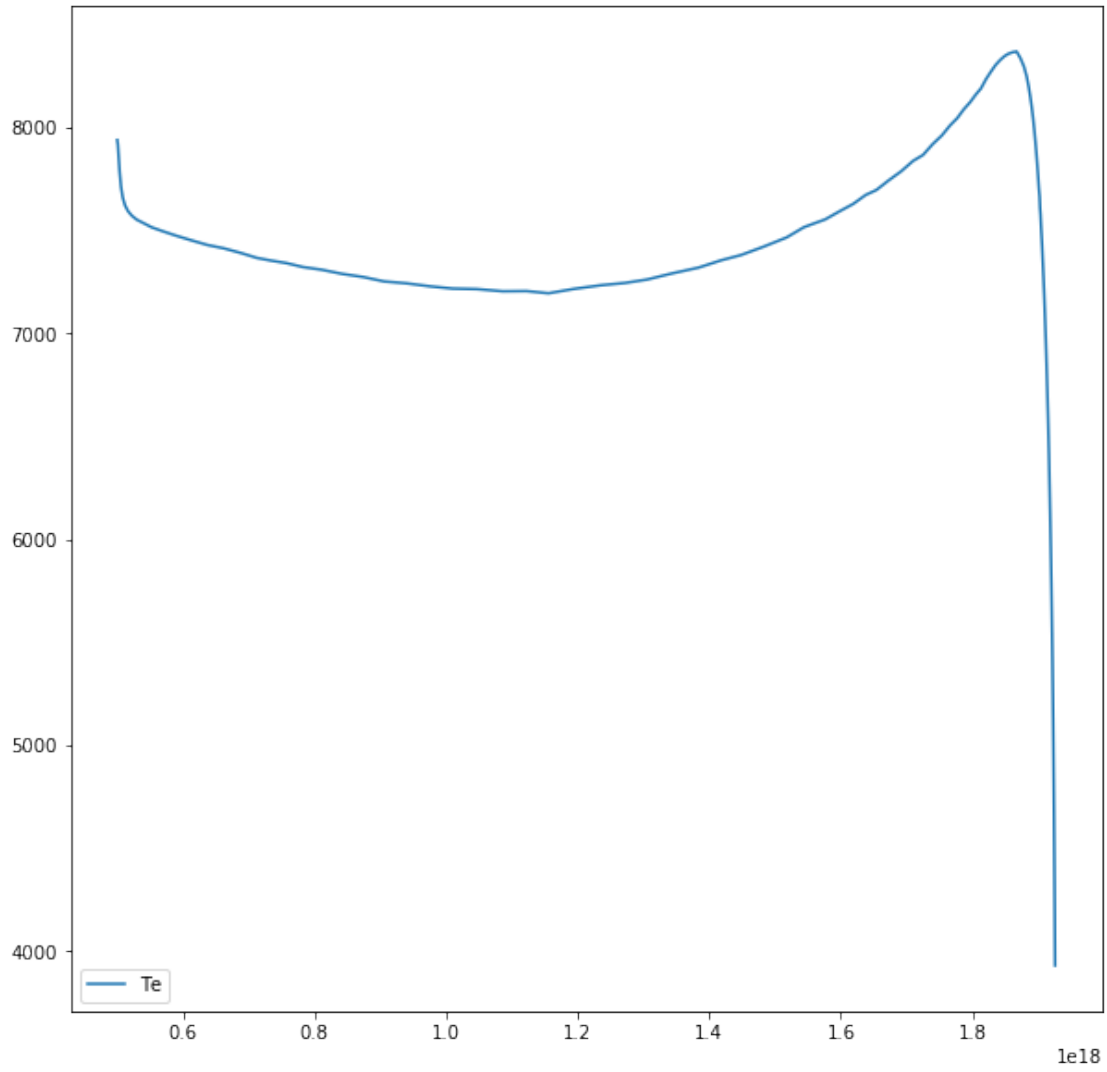
```
Out[28]: array(['H__1_486136A', 'H__1_656285A', 'CA_B_587564A', 'N__2_658345A',
                'O__1_630030A', 'O__2_372603A', 'O__2_372881A', 'O__3_500684A',
                'BLND_436300A', 'S__2_671644A', 'S__2_673082A', 'CL_3_551771A',
                'CL_3_553787A', 'O__1_631679M', 'O__1_145495M', 'C__2_157636M'],
               dtype='<U12')
```

In [29]:
```
# printing line intensities
for line in Mod.emis_labels:
    print('{0} {1:10.3e} {2:7.2f}'.format(line, Mod.get_emis_vol(line), Mod.get_emis_vol(line)
```

```
H__1_486136A  4.640e+34  100.00
H__1_656285A  1.269e+35  273.50
CA_B_587564A  7.194e+33   15.51
N__2_658345A  6.206e+34  133.74
O__1_630030A  1.318e+33    2.84
O__2_372603A  4.103e+34   88.42
O__2_372881A  5.520e+34  118.96
O__3_500684A  4.796e+34  103.37
BLND_436300A  9.520e+31    0.21
S__2_671644A  7.474e+33   16.11
S__2_673082A  5.713e+33   12.31
CL_3_551771A  1.171e+32    0.25
CL_3_553787A  8.356e+31    0.18
O__1_631679M  8.674e+32    1.87
O__1_145495M  8.462e+31    0.18
C__2_157636M  1.682e+32    0.36
```

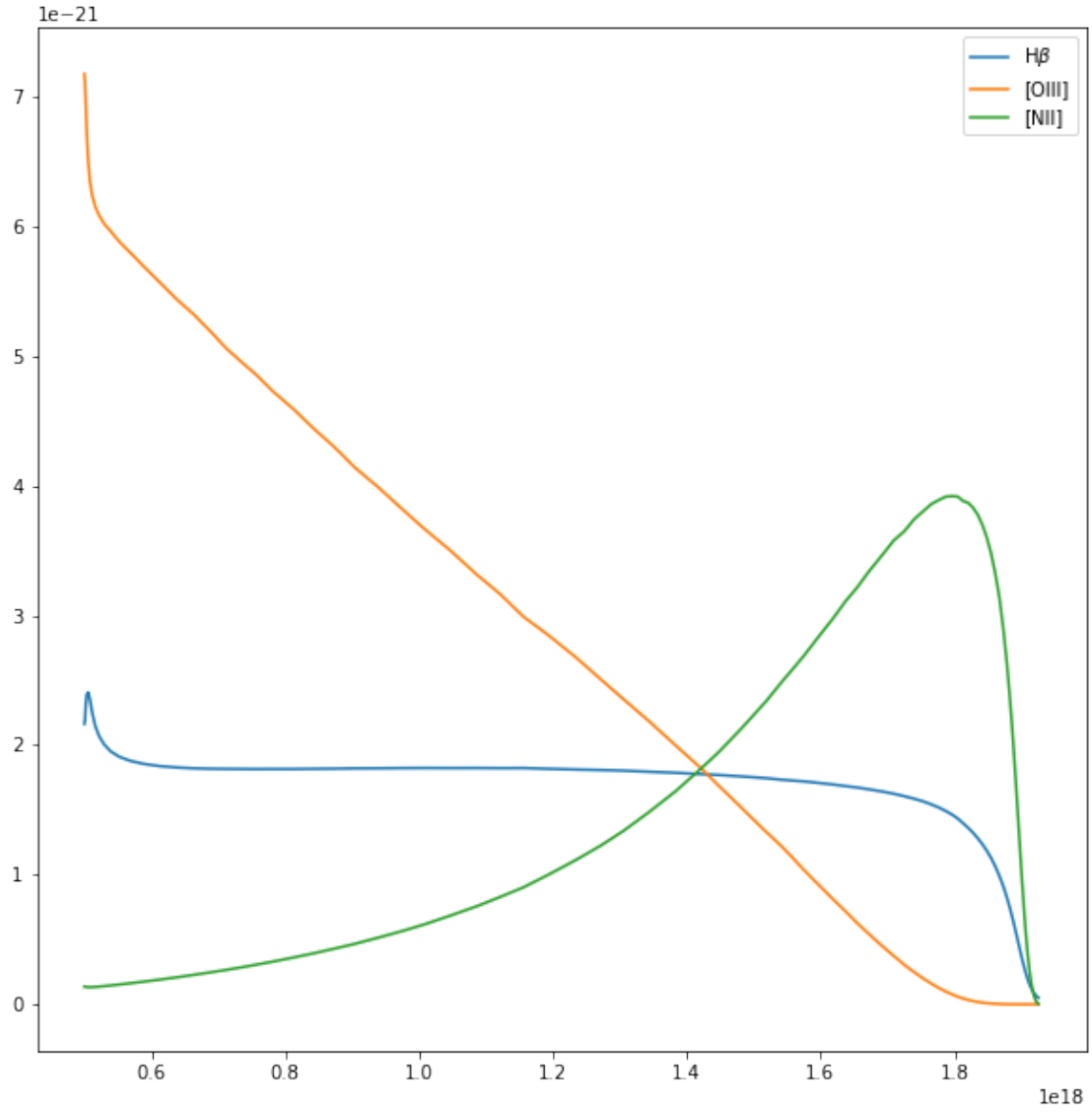
In [30]:
```
plt.figure(figsize=(10,10))
plt.plot(Mod.radius, Mod.te, label = 'Te')
plt.legend(loc=3)
```

Out[30]:
```
<matplotlib.legend.Legend at 0x7f242e7ce8d0>
```



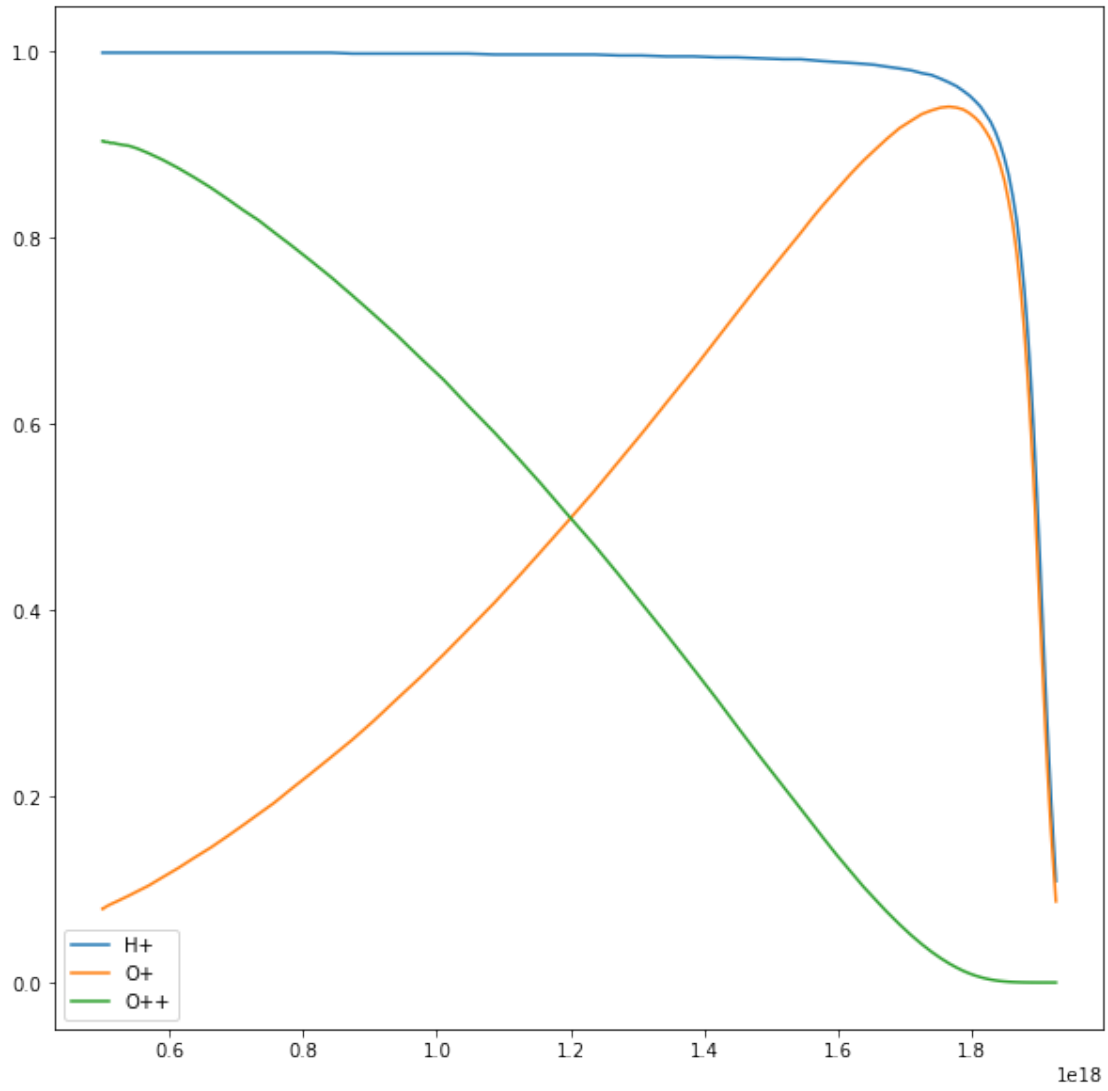

```
In [31]: plt.figure(figsize=(10,10))
         plt.plot(Mod.radius, Mod.get_emis('H__1_486136A'), label = r'H$\beta$')
         plt.plot(Mod.radius, Mod.get_emis('O__3_500684A'), label = '[OIII]')
         plt.plot(Mod.radius, Mod.get_emis('N__2_658345A'), label = '[NII]')
         plt.legend()

Out[31]: <matplotlib.legend.Legend at 0x7f242e3df908>
```



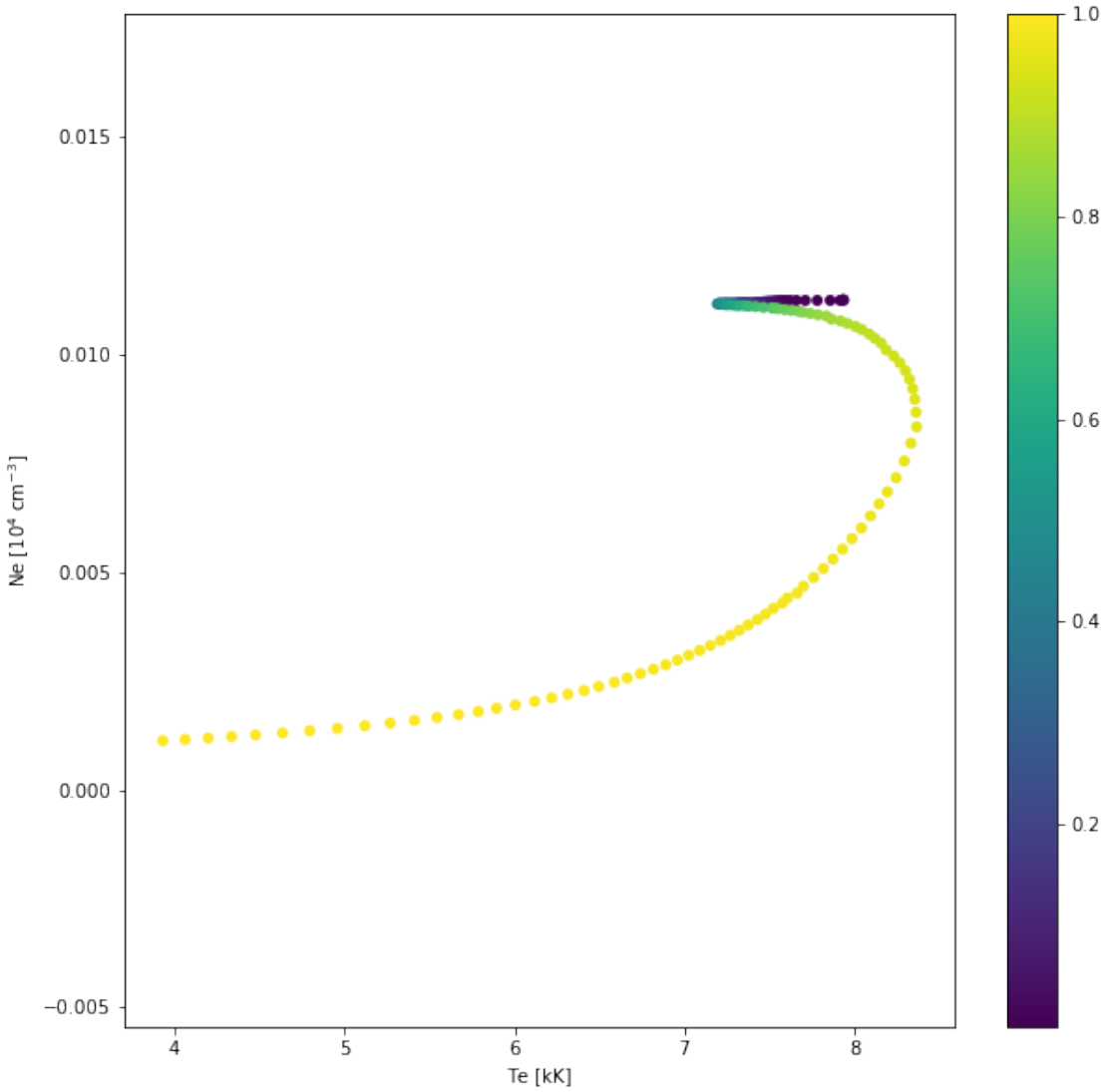
In [32]:
```
plt.figure(figsize=(10,10))
plt.plot(Mod.radius, Mod.get_ionic('H', 1), label = 'H+')
plt.plot(Mod.radius, Mod.get_ionic('O', 1), label = 'O+')
plt.plot(Mod.radius, Mod.get_ionic('O', 2), label = 'O++')
plt.legend(loc=3)
```

Out[32]: <matplotlib.legend.Legend at 0x7f242bdf2cf8>



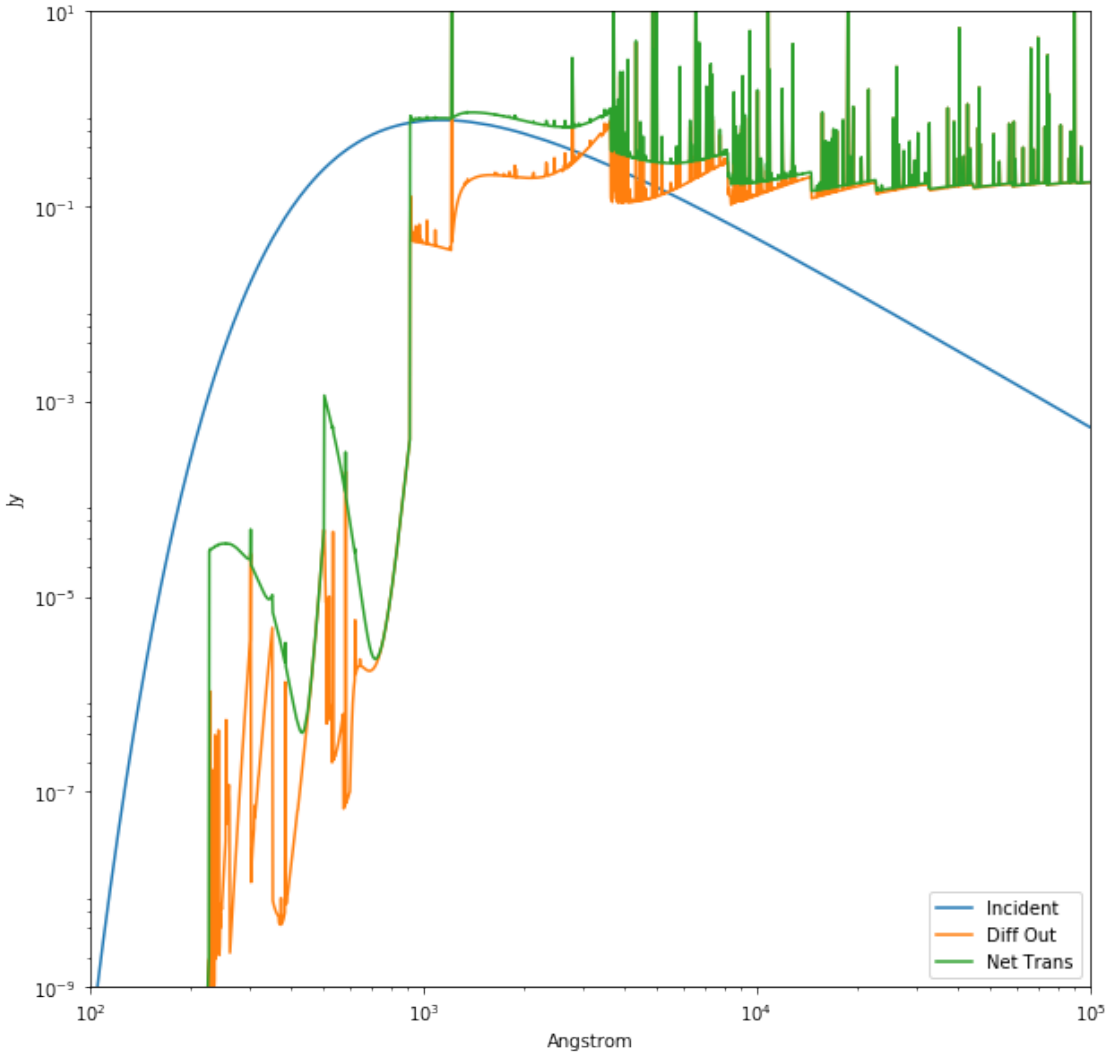
In [33]:
```
plt.figure(figsize=(10,10))
plt.scatter(Mod.te/1e3, Mod.ne/1e4, c = Mod.depth/np.max(Mod.depth), edgecolors = 'none')
plt.colorbar()
plt.xlabel('Te [kK]')
plt.ylabel(r'Ne [$10^4$ cm$^{-3}$]')
```

Out[33]: <matplotlib.text.Text at 0x7f242e4ec908>



In [34]:
```
plt.figure(figsize=(10,10))
plt.loglog(Mod.get_cont_x(unit='Ang'), Mod.get_cont_y(cont = 'incid', unit = 'Jy'), label = 'I
plt.loglog(Mod.get_cont_x(unit='Ang'), Mod.get_cont_y(cont = 'diffout', unit = 'Jy'), label =
plt.loglog(Mod.get_cont_x(unit='Ang'), Mod.get_cont_y(cont = 'ntrans', unit = 'Jy'), label = '
plt.xlim((100, 100000))
plt.ylim((1e-9, 1e1))
plt.xlabel('Angstrom')
plt.ylabel('Jy')
plt.legend(loc=4)
```

Out[34]:
```
<matplotlib.legend.Legend at 0x7f242bbdd5c0>
```


Incident
Diff Out
Net Trans
Angstrom
Jy