PyNeb_manual_7b

July 19, 2017

```
In [10]: import pyneb as pn
         import numpy as np
```

```
In [11]: obs = pn.Observation()
         obs.readData('observations1.dat', fileFormat='lines_in_rows', err_default=
         obs.def_EBV(label1="H1r_6563A", label2="H1r_4861A", r_theo=2.85)
         obs.correctData(normWave=4861.)
```

```
In [12]: obs.printIntens()
```

```
S4_10.5m      4.076
Ne2_12.8m     4.826
Ne3_15.6m    19.803
S3_18.7m      5.802
O2_3726A     46.576
O2_3729A     21.812
Ne3_3869A    21.722
Ne3_3968A     7.255
S2_4069A      0.950
S2_4076A      0.503
O3_4363A      4.687
H1r_4861A   100.000
O3_5007A    425.599
N2_5755A      0.454
S3_6312A      0.641
O1_6300A      1.428
O1_6364A      0.454
N2_6548A      5.657
H1r_6563A   285.000
N2_6584A     15.668
S2_6716A      0.995
S2_6731A      1.777
Ar3_7136A     3.882
O2_7319A+     5.106
O2_7330A+     4.034
```

```
In [13]: all_atoms = pn.getAtomDict(atom_list=obs.getUniqueAtoms())
         line_ab = {}
```

```
ion_ab = {}
temp = 12000.
dens = 1e4
for line in obs.getSortedLines():
    if line.atom != 'H1' and line.atom != 'He1' and line.atom != 'He2':
        line_ab[line.label] = all_atoms[line.atom].getIonAbundance(line.co
                                                          to_eval=line.to_
        if line.atom not in ion_ab:
            ion_ab[line.atom] = []
        ion_ab[line.atom].append(line_ab[line.label][0])
for line in sorted(line_ab):
    print('{:10} {:.2f}'.format(line, 12+np.log10(line_ab[line][0])))
```

```
warng _ManageAtomicData: rec data not available for Ar3
warng _ManageAtomicData: atom data not available for H1
warng _ManageAtomicData: coll data not available for H1
warng _ManageAtomicData: rec data not available for Ne2
warng _ManageAtomicData: rec data not available for Ne3
warng _ManageAtomicData: rec data not available for S2
warng _ManageAtomicData: rec data not available for S3
warng _ManageAtomicData: rec data not available for S4
Ar3_7136A  5.33
H1r_4861A  12.00
H1r_6563A  12.01
N2_5755A   6.36
N2_6548A   6.38
N2_6584A   6.36
Ne2_12.8m  6.77
Ne3_15.6m  7.11
Ne3_3869A  7.07
Ne3_3968A  7.12
O1_6300A   6.16
O1_6364A   6.16
O2_3726A   7.47
O2_3729A   7.48
O2_7319A+  7.32
O2_7330A+  7.29
O3_4363A   7.89
O3_5007A   7.92
S2_4069A   5.08
S2_4076A   5.29
S2_6716A   5.18
S2_6731A   5.12
S3_18.7m   5.93
S3_6312A   5.82
S4_10.5m   5.17
```

```
In [16]: for ion in sorted(ion_ab):
```

```
            print(ion, ion_ab[ion])

Ar3 [2.152894644253984e-07]
H1r [1.0, 1.0118929765886284]
N2 [2.2896520932105462e-06, 2.4132982669892579e-06, 2.2716848328034669e-06]
Ne2 [5.9051114501146047e-06]
Ne3 [1.2924607125398534e-05, 1.1873083428775325e-05, 1.3165090486499409e-05]
O1 [1.4587786123355056e-06, 1.4512515727307561e-06]
O2 [2.9484966486284868e-05, 3.0401998417595668e-05, 2.1076205142152512e-05, 1.94566
O3 [7.8241501660366078e-05, 8.3431919237520271e-05]
S2 [1.195881774170362e-07, 1.9578449298916088e-07, 1.5124222321153e-07, 1.331376274
S3 [8.5987288919891947e-07, 6.6551708910155244e-07]
S4 [1.4765564919152136e-07]
```

```
In [17]: for atom in ion_ab:
             mean = np.mean(np.asarray(ion_ab[atom]))
             ion_ab[atom] = mean
             print('{:4s}: {:4.2f}'.format(atom, 12+np.log10(mean)))

Ar3 : 5.33
H1r : 12.00
N2  : 6.37
Ne2 : 6.77
Ne3 : 7.10
O1  : 6.16
O2  : 7.40
O3  : 7.91
S2  : 5.18
S3  : 5.88
S4  : 5.17
```