atomneb Documentation

Release 0.3.3

Ashkbiz Danehkar

Mar 05, 2021

USER DOCUMENTATION

CHAPTER

ONE

INTRODUCTION

atomneb is a library written in Python for reading atomic data from a database containing atomic data stored in the Flexible Image Transport System (FITS) file format for *collisionally excited lines* and *recombination lines* typically observed in spectra of ionized gaseous nebulae. The AtomNeb database were generated for use in pyEQUIB, proEQUIB, and other nebular spectral analysis tools.

1.1 Collisional Excitation Unit

AtomNeb for collisionally excited lines contains sets of atomic datasets, which include energy levels (E_j), collision strengths (Ω_{ij}), and transition probabilities (A_{ij}) of the most ions commonly observed in ionized nebulae.

The atomic datasets for collisionally excited lines are as follows:

- Collection from the National Institute of Standards and Technology (NIST) Atomic Spectra Database, the CHIANTI atomic database, and some improved atomic data from Cloudy v13.04 and pyNeb v1.0. This collection was compiled according to the atomic data used in pyNeb v1.0.
- Chianti52 from the CHIANTI atomic database version 5.2. This dataset was compiled according to the atomic data used in MOCASSIN.
- Chianti60 from the CHIANTI atomic database version 6.0. This dataset was compiled according to the atomic data used in MOCASSIN.
- Chianti70 from the CHIANTI atomic database version 7.0. This dataset was compiled according to the atomic data used in MOCASSIN.
- Chianti90 from the CHIANTI atomic database version 9.0. This dataset was compiled according to the atomic data used in NEAT.

Each dataset contains the following atomic data FITS files: `AtomElj.fits` for *Energy Levels* (E_j), `AtomOmij.fits` for *Collision Strengths* (Ω_{ij}), and `AtomAij.fits` for *Transition Probabilities* (A_{ij}).

1.2 Recombination Unit

AtomNeb for recombination lines contains sets of effective recombination coefficients (α_{eff}) of recombination lines of H I, He I, He II, C I, C II, C III, C VI, N II, N III, N IV, N V, N VI, N VII, O II, O III, O IV, O V, O VI, O VIII, and Ne II ions typically observed in ionized nebulae, as well as Branching ratios (Br) of O II and N II lines.

The atomic datasets for recombination lines are as follows:

- RC Collection, effective recombination coefficients for C II (Davey et al. 2000), N II (Escalante and Victor 1990), O II (Storey 1994; Liu et al. 1995), and Ne II ions (Kisielius et al. 1998), including Branching ratios (Br) for O II and N II ions. This collection was compiled according to the atomic data used in MOCASSIN.

- SH95 Collection, hydrogenic ions for Z=1 to 8, namely H I, He II, Li III, Be IV, B V, C VI, N VII, and O VIII ions from Storey and Hummer (1995).
- PPB91 Collection, effective recombination coefficients for H, He, C, N, O, Ne ions from Pequignot, Petitjean and Boisson (1991).
- PFSD12 He I data, effective He I recombination coefficients from Porter et al (2012) and (2013a).
- FSL13 N II data, effective N II recombination coefficients (corrigendum) from Fang, Storey and Liu (2011) and (2013b).
- SSB17 O II data, effective O II recombination coefficients of 8889 recombination lines for Cases A, B, and C, and 2433 optical (3500-9000Å) recombination lines for Case B from Storey, Sochi and Bastin (2017).

CHAPTER TWO

INSTALLATION

To install the last version, all you should need to do is:

```
python setup.py install
```

To install the stable version, you can use the preferred installer program (pip):

```
pip install atomneb
```

or you can install it from the cross-platform package manager *conda*:

```
conda install -c conda-forge atomneb
```

To get this package with all the FITS file, you can simply use `git` command as follows:

```
git clone https://github.com/atomneb/AtomNeb-py
```

If you plan to use the recent O II recombination coefficients (Storey, Sochi and Bastin 2017), you need to unpack them:

```
cd AtomNeb-py/atomic-data-rc/
tar -xvf *.fits.tar.gz
```

This package requires the following packages:

- NumPy
- Astropy

CHAPTER

THREE

USAGE

The Documentation of the functions provides in detail in the *API Documentation* (atomneb.github.io/AtomNeb-py/doc). There are two main categories: *collisionally excited lines (CEL)* and *recombination lines (RC)*.

3.1 Collisional Excitation Unit

The atomic data for **collisional excitation unit (CEL)** contain Energy Levels (E_j), Collision Strengths (Ω_{ij}), and Transition Probabilities (A_{ij}). We have four atomic datasets for them: collection, chianti52, chianti60, and chianti70.

You need to load the **atomneb** library as follows:

```
import atomneb
```

Also:

```
import atomneb
```

Also:

```
import numpy as np
import os

atom_elj_file = os.path.join(base_dir,data_dir, 'AtomElj.fits')
atom_omij_file = os.path.join(base_dir,data_dir, 'AtomOmij.fits')
atom_aij_file = os.path.join(base_dir,data_dir, 'AtomAij.fits')
elj_data_list = atomneb.read_elj_list(atom_elj_file)
omij_data_list = atomneb.read_omij_list(atom_omij_file)
aij_data_list = atomneb.read_aij_list(atom_aij_file)
```

Now you have access to:

- *Energy Levels* (E_j):

```
atom='o'
ion='iii'
oiii_elj_data = atomneb.read_elj(atom_elj_file, atom, ion, level_num=6)
print(oiii_elj_data['j_v'])
print(oiii_elj_data['ej'])
```

which gives:

```
0.00000       1.00000       2.00000       2.00000       0.00000       2.00000
0.00000       113.200       306.200       20273.30      43185.69      60324.80
```

- *Collision Strengths* (Ω_{ij}):

```
atom='o'
ion='iii'
oiii_omij_data = atomneb.read_omij(atom_omij_file, atom, ion)
print(oiii_omij_data['level1'])
print(oiii_omij_data['level2'])
print(oiii_omij_data['strength'][0])
```

  which gives:

```
0        1        1        1        1        ...
0        2        3        4        5        ...
100.0       158.50       251.20       398.10       631.0       ...
```

- *Transition Probabilities* (A_{ij}):

```
atom='o'
ion='iii'
oiii_aij_data = atomneb.read_aij(atom_aij_file, atom, ion)
print(oiii_aij_data['aij'][0])
```

  which gives:

```
0.0000   2.5969e-05       0.0000   2.3220e-06      ...
```

3.2 Recombination Unit

The atomic data for **recombination unit (RC)** contain effective recombination coefficients (α_{eff}) of emission lines from different collections: RC Collection, SH95 Collection, PPB91 Collection, PFSD12 He I data, FSL13 N II data, and SSB17 O II data.

You need to load the **atomneb** libary:

```
import atomneb
```

Also:

```
import numpy as np
import os
```

Now you have access to effective recombination coefficients (α_{eff}) of the following collections:

- *RC Collection*:

```
atom_rc_file = os.path.join(base_dir,data_dir, 'rc_collection.fits')
atom='c'
ion='iii'
cii_rc_data = atomneb.read_aeff_collection(atom_rc_file, atom, ion)
n_line = len(cii_rc_data['wavelength'])
for i in range(0, n_line):
     print(cii_rc_data['wavelength'][i], cii_rc_data['a'][i],
           cii_rc_data['b'][i], cii_rc_data['c'][i],
           cii_rc_data['d'][i], cii_rc_data['f'][i])
```

  which gives:

```
914.00000      0.69280000     0.021400000    -0.016300000     -0.
↪24310000     -0.88000000
962.00000       1.0998000   -0.0042000000    -0.027900000     -0.
↪22940000     -0.96560000
997.00000      0.78210000     -0.36840000   0.00030000000     -0.
↪12170000     -0.78740000
...
```

- *SH95 Collection*:

```
atom_rc_file = os.path.join(base_dir,data_dir, 'rc_SH95.fits')
atom='h'
ion='ii'
hi_rc_data = atomneb.read_aeff_sh95(atom_rc_file, atom, ion)
print(hi_rc_data['aeff'][0])
```

which gives:

```
100.00000       500.00000       0.0000000   4.2140000e-27   1.7560000e-
↪27   1.0350000e-27
...
```

- *PPB91 Collection*:

```
atom_rc_file = os.path.join(base_dir,data_dir, 'rc_PPB91.fits')
atom='c'
ion='iii'
cii_rc_data = atomneb.read_aeff_ppb91(atom_rc_file, atom, ion)
n_line = len(cii_rc_data['wavelength'])
for i in range(0, n_line):
   print(cii_rc_data['ion'][i], cii_rc_data['case1'][i], cii_rc_data[
↪'wavelength'][i],
         cii_rc_data['a'][i], cii_rc_data['b'][i], cii_rc_data['c'][i],
         cii_rc_data['d'][i], cii_rc_data['br'][i], cii_rc_data['q'][i],␣
↪cii_rc_data['y'][i])
```

which gives:

```
C2+A       9903.4600      0.69700000     -0.78400000       4.2050000     ␣
↪ 0.72000000       1.0000000       1.6210000
C2+A       4267.1500       1.0110000     -0.75400000       2.5870000     ␣
↪ 0.71900000      0.95000000       2.7950000
...
```

- *PFSD12 He I data*:

```
atom_rc_file = os.path.join(base_dir,data_dir, 'rc_he_ii_PFSD12.fits')
atom='he'
ion='ii'
hei_rc_data = atomneb.read_aeff_he_i_pfsd12(atom_rc_file, atom, ion)
hei_rc_data_wave = atomneb.read_aeff_he_i_pfsd12(atom_rc_file, atom, ion,
↪ wavelength=True)
print(hei_rc_data['aeff'][0])
```

which gives:

```
5000.0000       10.000000      -25.379540      -25.058970      -25.
↪948440       -24.651820      -25.637660
...
```

- *FSL13 N II data*:

```
atom_rc_file = os.path.join(base_dir,data_dir, 'rc_n_iii_FSL13.fits')
atom='n'
ion='iii'
wavelength_range=[4400.0, 7100.0]
nii_rc_data = atomneb.read_aeff_n_ii_fsl13(atom_rc_file, atom, ion,␣
↪wavelength_range)
nii_rc_data_wave = atomneb.read_aeff_n_ii_fsl13(atom_rc_file, atom, ion,␣
↪wavelength_range, wavelength=True)
print(nii_rc_data.aeff[0])
n_line = len(hei_rc_data_wave['wavelength'])
for i in range(0, n_line):
   print(nii_rc_data_wave['wavelength'][i], nii_rc_data_wave['tr'][i],␣
↪nii_rc_data_wave['trans'][i])
```

which gives:

```
255.000      79.5000      47.3000      12.5000      6.20000      4.00000␣
↪      2.86000
258.000      54.4000      29.7000      7.92000      4.11000      2.72000␣
↪      2.00000
310.000      48.1000      23.7000      5.19000      2.55000      1.65000␣
↪      1.21000
434.000      50.3000      23.2000      4.71000      2.26000      1.45000␣
↪      1.05000

6413.23 6g - 4f2p6g G[9/2]o4 - 2p4f F[7/2]e3
6556.32 6g - 4f2p6g G[9/2]o5 - 2p4f G[7/2]e4
6456.97 6g - 4f2p6g G[9/2]o5 - 2p4f F[7/2]e4
6446.53 6g - 4f2p6g F[7/2]o3 - 2p4f D[5/2]e2
6445.34 6g - 4f2p6g F[7/2]o4 - 2p4f D[5/2]e3
...
```

- *SSB17 O II data*: You first need to unpack rc_o_iii_SSB17_orl_case_b.fits.tar.gz, e.g.:

```
tar -xvf rc_o_iii_SSB17_orl_case_b.fits.tar.gz
```

If you need to have access to the full dataset (entire wavelengths, case A and B):

```
tar -xvf rc_o_iii_SSB17.fits.tar.gz
```

Please note that using the entire atomic data will make your program very slow and you may need to have a higher memory on your system. Without the above comment, as default, the cose uses rc_o_iii_SSB17_orl_case_b.fits:

```
aatom_rc_file = os.path.join(base_dir,data_dir, 'rc_o_iii_SSB17_orl_case_
↪b.fits')
atom='o'
ion='iii'
case1='B'
wavelength_range=[5320.0, 5330.0]
oii_rc_data = atomneb.read_aeff_o_ii_ssb17(atom_rc_file, atom, ion,␣
↪case1, wavelength_range)
```

(continues on next page)

(continued from previous page)

```
oii_rc_data_wave = atomneb.read_aeff_o_ii_ssb17(atom_rc_file, atom, ion,␣
↪case1, wavelength_range, wavelength=True)
print(oii_rc_data['aeff'][0])
n_line = len(oii_rc_data_wave['wavelength'])
for i in range(0, n_line):
   print(oii_rc_data_wave['wavelength'][i], oii_rc_data_wave['lower_term
↪'][i], oii_rc_data_wave['upper_term'][i])
```

which gives:

```
1.64100e-30  1.60000e-30  1.56400e-30  1.54100e-30  1.52100e-30  1.
↪50900e-30
...

5327.17 2s22p2(1S)3p 2Po
5325.42 2s22p2(1S)3p 2Po
5327.18 2s22p2(1D)3d 2Ge
5326.84 2s22p2(1D)3d 2Ge
...
```

CHAPTER

FOUR

REFERENCES

- Danehkar, A. (2020). AtomNeb Python Package, an addendum to AtomNeb: IDL Library for Atomic Data of Ionized Nebulae. *J. Open Source Softw.*, **5**, 2797. doi:10.21105/joss.02797 ads:2020JOSS....5.2797D.
- Danehkar, A. (2019). AtomNeb: IDL Library for Atomic Data of Ionized Nebulae. *J. Open Source Softw.*, **4**, 898. doi:10.21105/joss.00898 ads:2019JOSS....4..898D.

CHAPTER FIVE

ATOMNEB.MAIN PACKAGE

5.1 atomneb main module

This module contains functions for Atomic Data of Ionized Nebulae

atomneb.**get_aeff_collection_reference_citation**(*atom_rc_file*, *atom*, *ion*, *br=None*, *reference=None*)

This function returns the reference citation for a recombination coefficient (Aeff) from the 2nd binary table extension of the FITS data file ('rc_collection.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data-rc')
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_collection.fits')
>> atom='c'
>> ion='iii' # C III
>> citation = atomneb.get_aeff_collection_reference_citation(atom_rc_file,
↪ atom, ion)
>> print(citation)
   Davey, A. R., Storey, P. J. and Kisielius, R., Astron.Astrophys.Suppl.,
↪ 142, 85, 2000
```

Returns This function returns the Citation.

Return type str

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_collection.fits')
- **atom** (*str*) – atom name e.g. 'c'
- **ion** (*str*) – ionic level e.g 'iii'
- **br** (*boolean, optional*) – set for the branching ratios (Br)
- **reference** (*str, optional*) – set for the reference

atomneb.**get_aeff_he_i_pfsd12_reference_citation**(*atom_rc_file*, *atom*, *ion*, *reference=None*)

This function returns the reference citation for a recombination coefficient (Aeff) from the 2nd binary table extension of the FITS data file ('rc_he_ii_PFSD12.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = ['atomic-data-rc']
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_he_ii_PFSD12.fits')
>> atom='he'
>> ion='ii' # He I
>> reference='PFSD13'
>> citation = atomneb.get_aeff_he_i_pfsd12_reference_citation(atom_rc_
↪file, atom, ion, reference=reference)
>> print(citation)
   Porter, R. L., Ferland, G. J., Storey, P. J. and Detisch, M. J., MNRAS,
↪ 433L, 89, 2013
```

Returns This function returns the Citation.

Return type str

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_he_ii_PFSD12.fits')
- **atom** (*str*) – atom name e.g. 'he'
- **ion** (*str*) – ionic level e.g 'ii'
- **reference** (*str, optional*) – set for the reference e.g. 'PFSD13'

atomneb.**get_aeff_n_ii_fsl13_reference_citation**(*atom_rc_file, atom, ion, reference=None*)

This function returns the reference citation for a recombination coefficient (Aeff) from the 2nd binary table extension of the FITS data file ('rc_n_iii_FSL13.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = ['atomic-data-rc']
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_n_iii_FSL13.fits')
>> atom='n'
>> ion='iii' # N II
>> reference='FSL13'
>> citation = atomneb.get_aeff_n_ii_fsl13_reference_citation(atom_rc_file,
↪ atom, ion)
>> print(citation)
   Fang X., Storey P.J., and Liu X.-W., R. 2011, Astron.Astrophys. 530,␣
↪A18; 2013, Astron.Astrophys. 550, C2
```

Returns This function returns the Citation.

Return type str

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_n_iii_FSL13.fits')

- **atom** (*str*) – atom name e.g. 'n'
- **ion** (*str*) – ionic level e.g 'iii'
- **reference** (*str, optional*) – set for the reference e.g. 'FSL13

atomneb.**get_aeff_o_ii_ssb17_reference_citation**(*atom_rc_file*, *atom*, *ion*, *reference=None*)

This function returns the reference citation for a recombination coefficient (Aeff) from the 2nd binary table extension of the FITS data file ('rc_o_iii_SSB17.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = ['atomic-data-rc']
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_o_iii_SSB17_orl_
↪case_b.fits')
>> atom='o'
>> ion='iii' # O II
>> reference='SSB17'
>> citation = atomneb.get_aeff_o_ii_ssb17_reference_citation(atom_rc_file,
↪ atom, ion)
>> print(citation)
   Storey, P.J., Sochi, T. and Bastin, R. 2017, MNRAS, 470, 379; VizieR␣
↪On-line Data Catalog: VI/150
```

Returns This function returns the Citation.

Return type str

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_o_iii_SSB17.fits')
- **atom** (*str*) – atom name e.g. 'o'
- **ion** (*str*) – ionic level e.g 'iii'
- **reference** (*str, optional*) – set for the reference e.g. 'SSB17

atomneb.**get_aeff_ppb91_reference_citation**(*atom_rc_file*, *atom*, *ion*, *reference=None*)

This function returns the reference citation for a recombination coefficient (Aeff) from the 2nd binary table extension of the FITS data file ('rc_PPB91.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = ['atomic-data-rc']
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_PPB91.fits')
>> atom='c'
>> ion='iii'
>> citation = atomneb.get_aeff_ppb91_reference_citation(atom_rc_file,␣
↪atom, ion)
```

(continues on next page)

(continued from previous page)

```
>> print(citation)
   Pequignot, D., Petitjean, P. and Boisson, C. Astron.Astrophys., 251,␣
↪680, 1991
```

Returns This function returns the Citation.

Return type str

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_PPB91.fits')
- **atom** (*str*) – atom name e.g. 'c'
- **ion** (*str*) – ionic level e.g 'iii'
- **reference** (*str, optional*) – set for the reference

atomneb.**get_aeff_sh95_reference_citation**(*atom_rc_file*, *atom*, *ion*, *reference=None*, *case1=None*)

This function returns the reference citation for a recombination coefficient (Aeff) from the 2nd binary table extension of the FITS data file ('rc_SH95.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = ['atomic-data-rc']
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_SH95.fits')
>> atom='h'
>> ion='ii' # H I
>> citation = atomneb.get_aeff_sh95_reference_citation(atom_rc_file, atom,
↪ ion)
>> print(citation)
   Storey, P. J. and Hummer, D. G., MNRAS, 272, 41S, 1995
```

Returns This function returns the Citation.

Return type str

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_SH95.fits')
- **atom** (*str*) – atom name e.g. 'h'
- **ion** (*str*) – ionic level e.g 'ii'
- **reference** (*str, optional*) – set for the reference
- **case1** (*str, optional*) – set for the case 'a' or 'b', defualt 'b'

atomneb.**get_aij_reference_citation**(*atom_aij_file*, *atom*, *ion*, *reference*)

This function returns the reference citation for a transition probability (Aij) from the 2nd binary table extension of the FITS data file ('AtoAij.fits')

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data', 'collection')
>> atom_aij_file = os.path.join(base_dir,data_dir, 'AtomAij.fits')
>> atom='o'
>> ion='iii'
>> reference='FFT04'
>> citation = atomneb.get_aij_reference_citation(atom_aij_file, atom, ion,
↪ reference)
>> print(citation)
   Froese Fischer et al 2004, ADNDT 87, 1
```

Returns This function returns the Citation.

Return type str

Parameters

- **Atom_Aij_file** (*str*) – the FITS data file name ('AtoAij.fits')
- **atom** (*str*) – atom name e.g. 'o'
- **ion** (*str*) – ionic level e.g 'iii'
- **reference** (*str*) – set for the reference e.g. 'FFT04'

atomneb.**get_elj_reference_citation**(*atom_elj_file*, *reference*)

This function returns the reference citation for energy levels (Ej) from the 2nd binary table extension of the FITS data file ('AtomElj.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data', 'collection')
>> atom_elj_file = os.path.join(base_dir,data_dir, 'AtomElj.fits')
>> reference='L7288'
>> citation=atomneb.get_elj_reference_citation(atom_elj_file, reference)
>> print(citation)
   C. E. Moore, in CRC Series in Evaluated Data in Atomic Physics, 339 pp.
↪ (CRC Press, Boca Raton, FL, 1993)
```

Returns This function returns the Citation.

Return type str

Parameters

- **atom_elj_file** (*str*) – the FITS data file name ('AtomElj.fits')
- **reference** (*str*) – set for the reference e.g. 'L7288'

atomneb.**get_omij_reference_citation**(*atom_omij_file*, *atom*, *ion*, *reference*)

This function returns the reference citation for collision strengths (Omega_ij) from the 2nd binary table extension of the FITS data file ('AtomOmij.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data', 'collection')
>> atom_omij_file = os.path.join(base_dir,data_dir, 'AtomOmij.fits')
>> atom='o'
>> ion='iii'
>> reference='SSB14'
>> citation = atomneb.get_omij_reference_citation(atom_omij_file, atom,␣
↪ion, reference)
>> print(citation)
   Storey, P. J., Sochi, T., and Badnell, N. R. 2014, Astron.Astrophys.,␣
↪441, 3028
```

Returns This function returns the Citation.

Return type str

Parameters

- **atom_omij_file** (*str*) – the FITS data file name ('AtomOmij.fits')
- **atom** (*str*) – atom name e.g. 'o'
- **ion** (*str*) – ionic level e.g 'iii'
- **reference** (*str*) – set for the reference e.g. 'SSB14'

atomneb.**list_aeff_collection_references**(*atom_rc_file*, *atom*, *ion*, *br=None*)

This function returns a list for all references of recombination coefficients (Aeff) for given element and ionic level from the FITS data file ('rc_collection.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data-rc')
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_collection.fits')
>> atom='c'
>> ion='iii' # C III
>> list_cii_aeff_reference = atomneb.list_aeff_collection_references(atom_
↪rc_file, atom, ion)
>> print(list_cii_aeff_reference)
```

Returns This function returns the references.

Return type an array of strings

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_collection.fits')
- **atom** (*str*) – atom name e.g. 'c'
- **ion** (*str*) – ionic level e.g 'iii'

- **br** (*boolean, optional*) – set for the branching ratios (Br)

atomneb.**list_aeff_he_i_pfsd12_references**(*atom_rc_file*, *atom*, *ion*)

This function returns a list for all references of recombination coefficients (Aeff) for given element and ionic level from the FITS data file ('rc_he_ii_PFSD12.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = ['atomic-data-rc']
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_he_ii_PFSD12.fits')
>> atom='he'
>> ion='ii' # He I
>> list_hei_aeff_reference = atomneb.list_aeff_he_i_pfsd12_
↪references(atom_rc_file, atom, ion)
>> print(list_hei_aeff_reference)
   PFSD12 PFSD13
```

Returns This function returns the references.

Return type an array of strings

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_he_ii_PFSD12.fits')
- **atom** (*str*) – atom name e.g. 'he'
- **ion** (*str*) – ionic level e.g 'ii'

atomneb.**list_aeff_n_ii_fsl13_references**(*atom_rc_file*, *atom*, *ion*)

This function returns a list for all references of recombination coefficients (Aeff) for given element and ionic level from the FITS data file ('rc_n_iii_FSL13.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = ['atomic-data-rc']
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_n_iii_FSL13.fits')
>> atom='n'
>> ion='iii' # N II
>> list_nii_aeff_reference = atomneb.list_aeff_n_ii_fsl13_references(atom_
↪rc_file, atom, ion)
>> print(list_nii_aeff_reference)
```

Returns This function returns the references.

Return type an array of strings

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_n_iii_FSL13.fits')
- **atom** (*str*) – atom name e.g. 'n'

- **ion** (*str*) – ionic level e.g 'iii'

atomneb.**list_aeff_o_ii_ssb17_references** (*atom_rc_file*, *atom*, *ion*)

This function returns a list for all references of recombination coefficients (Aeff) for given element and ionic level from the FITS data file ('rc_o_iii_SSB17.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = ['atomic-data-rc']
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_o_iii_SSB17_orl_
↪case_b.fits')
>> atom='o'
>> ion='iii' # O II
>> list_oii_aeff_reference = atomneb.list_aeff_o_ii_ssb17_references(atom_
↪rc_file, atom, ion)
>> print(list_oii_aeff_reference)
```

Returns This function returns the references.

Return type an array of strings

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_o_iii_SSB17.fits')
- **atom** (*str*) – atom name e.g. 'o'
- **ion** (*str*) – ionic level e.g 'iii'

atomneb.**list_aeff_ppb91_references** (*atom_rc_file*, *atom*, *ion*)

This function returns a list for all references of recombination coefficients (Aeff) for given element and ionic level from the FITS data file ('rc_PPB91.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = ['atomic-data-rc']
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_PPB91.fits')
>> atom='c'
>> ion='iii'
>> list_cii_aeff_reference = atomneb.list_aeff_ppb91_references(atom_rc_
↪file, atom, ion)
>> print(list_cii_aeff_reference)
```

Returns This function returns the references.

Return type an array of strings

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_PPB91.fits')
- **atom** (*str*) – atom name e.g. 'c'

- **ion** (*str*) – ionic level e.g 'iii'

atomneb.**list_aeff_sh95_references** (*atom_rc_file*, *atom*, *ion*)

This function returns a list for all references of recombination coefficients (Aeff) for given element and ionic level from the FITS data file ('rc_SH95.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = ['atomic-data-rc']
>> atom_rc_file = os.path.join(base_dir,data_dir,'rc_SH95.fits')
>> atom='h'
>> ion='ii' # H I
>> list_hi_aeff_reference = atomneb.list_aeff_sh95_references(atom_rc_
↪file, atom, ion)
>> print(list_hi_aeff_reference)
```

Returns This function returns the references.

Return type an array of strings

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_SH95.fits')
- **atom** (*str*) – atom name e.g. 'h'
- **ion** (*str*) – ionic level e.g 'ii'

atomneb.**list_aij_references** (*atom_aij_file*, *atom*, *ion*)

This function returns a list for all references of transition probabilities (Aij) for given element and ionic level from the FITS data file ('AtoAij.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data', 'collection')
>> atom_aij_file = os.path.join(base_dir,data_dir, 'AtomAij.fits')
>> atom='o'
>> ion='iii'
>> list_oiii_aij_reference = atomneb.list_aij_references(atom_aij_file,␣
↪atom, ion)
>> print(list_oiii_aij_reference)
   FFT04-SZ00 FFT04 GMZ97-WFD96 SZ00-WFD96
```

Returns This function returns the references.

Return type an array of data

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('AtoAij.fits')
- **atom** (*str*) – atom name e.g. 'o'

- **ion** (*str*) – ionic level e.g 'iii'

atomneb.**list_omij_references**(*atom_omij_file*, *atom*, *ion*)

This function returns a list for all references of collision strengths (Omega_ij) for given element and ionic level from the FITS data file ('AtomOmij.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data', 'collection')
>> atom_omij_file = os.path.join(base_dir,data_dir, 'AtomOmij.fits')
>> atom='o'
>> ion='iii'
>> list_oiii_omij_reference = atomneb.list_omij_references(atom_omij_file,
↪ atom, ion)
>> print(list_oiii_omij_reference)
   AK99 LB94 Pal12-AK99 SSB14
```

Returns This function returns the references.

Return type str

Parameters

- **atom_omij_file** (*str*) – the FITS data file name ('AtomOmij.fits')
- **atom** (*str*) – atom name e.g. 'o'
- **ion** (*str*) – ionic level e.g 'iii'

atomneb.**read_aeff_collection**(*atom_rc_file*, *atom*, *ion*, *br=None*, *reference=None*)

This function returns the effective recombination coefficients (Aeff) from the table extensions of the FITS data file ('rc_collection.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data-rc')
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_collection.fits')
>> atom='c'
>> ion='iii' # C III
>> cii_rc_data = atomneb.read_aeff_collection(atom_rc_file, atom, ion)
>> n_line = len(cii_rc_data['wavelength'])
>> for i in range(0, n_line):
>>       print(cii_rc_data['wavelength'][i], cii_rc_data['a'][i],
>>             cii_rc_data['b'][i], cii_rc_data['c'][i],
>>             cii_rc_data['d'][i], cii_rc_data['f'][i])
   914.00000      0.69280000     0.021400000    -0.016300000     -0.
↪24310000     -0.88000000
   962.00000       1.0998000   -0.0042000000    -0.027900000     -0.
↪22940000     -0.96560000
   ...
```

Returns This function returns the effective recombination coefficients. aeff_data (c_iii_aeff) { Wavelength:0.0,a: 0.0, b: 0.0, c: 0.0, d: 0.0, f: 0.0}, aeff_data (n_iii_aeff) { a: 0.0, b: 0.0, c: 0.0}, aeff_data (n_iii_br) {Wavelength: 0.0, BR: 0.0, g1:0, g2:0, Mult1:", LowerTerm:", UpperTerm:" }, aeff_data (o_iii_aeff) {Term: '', Case1: '', a2: 0.0, a4: 0.0, a5: 0.0, a6: 0.0, b: 0.0, c: 0.0, d: 0.0}, aeff_data (o_iii_br) {Wavelength:double(0.0), Br_A: 0.0, Br_B: 0.0, Br_C: 0.0, g1: 0, g2: 0, Mult1: '', LowerTerm: '', UpperTerm: ''}, aeff_data (ne_iii_aeff) {Wavelength:0.0, a: 0.0, b: 0.0, c: 0.0, d: 0.0, f: 0.0, br: 0.0},

Return type an array of data

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_collection.fits')
- **atom** (*str*) – atom name e.g. 'c'
- **ion** (*str*) – ionic level e.g 'iii'
- **br** (*boolean, optional*) – set for the branching ratios (Br)
- **reference** (*str, optional*) – set for the reference

atomneb.**read_aeff_collection_list**(*atom_rc_file*)

This function returns the list of effective recombination coefficients (Aeff) from the 1st binary table extension of the FITS data file ('rc_collection.fits')

Returns This function returns the aeff_data_list: { Aeff_Data:", Extension:0.0}

Return type an array of data

Parameters **atom_rc_file** (*str*) – the FITS data file name ('rc_collection.fits')

atomneb.**read_aeff_collection_references**(*atom_rc_file*)

This function returns the reference list of recombination coefficients (Aeff) from the 2nd binary table extension of the FITS data file ('rc_collection.fits').

Returns This function returns the aeff_data_reference: { Reference:", Citation:"}

Return type an array of data

Parameters **atom_rc_file** (*str*) – the FITS data file name ('rc_collection.fits')

atomneb.**read_aeff_he_i_pfsd12**(*atom_rc_file, atom, ion, wavelength=None, reference=None*)

This function returns the effective recombination coefficients (Aeff) from the table extensions of the FITS data file ('rc_he_ii_PFSD12.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data-rc')
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_he_ii_PFSD12.fits')
>> atom='he'
>> ion='ii' # He I
>> hei_rc_data = atomneb.read_aeff_he_i_pfsd12(atom_rc_file, atom, ion)
>> hei_rc_data_wave = atomneb.read_aeff_he_i_pfsd12(atom_rc_file, atom,␣
↪ion, wavelength=True)
```

(continues on next page)

(continued from previous page)

```
>> print(hei_rc_data['aeff'][0])
   5000.0000       10.000000      -25.379540      -25.058970      -25.
↪948440      ...
>> n_line = len(hei_rc_data_wave['wavelength'])
>> for i in range(0, n_line):
>>     print(hei_rc_data_wave['wavelength'][i],
>>           hei_rc_data_wave['lowerterm'][i], hei_rc_data_wave['upperterm
↪'][i])
   2945.00005p^{3}P2s^{3}S
   3188.00004p^{3}P2s^{3}S
   3614.00005p^{1}P2s^{1}S
   ...
```

Returns This function returns the effective recombination coefficients.

Return type an array of data

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_he_ii_PFSD12.fits')
- **atom** (*str*) – atom name e.g. 'he'
- **ion** (*str*) – ionic level e.g 'ii'
- **wavelength** (*boolean, optional*) – set for returning the wavelengths
- **reference** (*str, optional*) – set for the reference

atomneb.**read_aeff_he_i_pfsd12_list**(*atom_rc_file*)

This function returns the list of effective recombination coefficients (Aeff) from the 1st binary table extension of the FITS data file ('rc_he_ii_PFSD12.fits')

Returns This function returns the aeff_data_list: { Aeff_Data:", Extension:0.0}

Return type an array of data

Parameters **atom_rc_file** (*str*) – the FITS data file name ('rc_he_ii_PFSD12.fits')

atomneb.**read_aeff_he_i_pfsd12_references**(*atom_rc_file*)

This function returns the reference list of recombination coefficients (Aeff) from the 2nd binary table extension of the FITS data file ('rc_he_ii_PFSD12.fits').

Returns This function returns the aeff_data_reference:{ Reference:", Citation:"}

Return type an array of data

Parameters **atom_rc_file** (*str*) – the FITS data file name ('rc_he_ii_PFSD12.fits')

atomneb.**read_aeff_n_ii_fsl13**(*atom_rc_file, atom, ion, wavelength_range, wavelength=None, reference=None*)

This function returns the effective recombination coefficients (Aeff) from the table extensions of the FITS data file ('rc_n_iii_FSL13.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data-rc')
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_n_iii_FSL13.fits')
>> atom='n'
>> ion='iii' # N II
>> wavelength_range=[4400.0, 7100.0]
>> nii_rc_data = atomneb.read_aeff_n_ii_fsl13(atom_rc_file, atom, ion,␣
↪wavelength_range)
>> nii_rc_data_wave = atomneb.read_aeff_n_ii_fsl13(atom_rc_file, atom,␣
↪ion, wavelength_range, wavelength=True)
>> print(nii_rc_data['aeff'][0])
   255.000      79.5000      47.3000      12.5000      ...
>> n_line = len(nii_rc_data_wave['wavelength'])
>> for i in range(0, n_line):
>>     print(nii_rc_data_wave['wavelength'][i], nii_rc_data_wave['tr'][i],
↪ nii_rc_data_wave['trans'][i])
   6413.236g - 4f2p6g G[9/2]o4 - 2p4f F[7/2]e3
   6556.326g - 4f2p6g G[9/2]o5 - 2p4f G[7/2]e4
   6456.976g - 4f2p6g G[9/2]o5 - 2p4f F[7/2]e4
   ...
```

Returns This function returns the effective recombination coefficients.

Return type an array of data

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_n_iii_FSL13.fits')
- **atom** (*str*) – atom name e.g. 'n'
- **ion** (*str*) – ionic level e.g 'iii'
- **wavelength** (*boolean, optional*) – set for returning the wavelengths
- **reference** (*str, optional*) – set for the reference

atomneb.**read_aeff_n_ii_fsl13_list**(*atom_rc_file*)

This function returns the list of effective recombination coefficients (Aeff) from the 1st binary table extension of the FITS data file ('rc_n_iii_FSL13.fits')

Returns This function returns the aeff_data_list: {Aeff_Data:", Extension:0, IND:long(0), Wavelength: float(0.0), Tr:", Trans: '', T_X: ''}

Return type an array of data

Parameters **atom_rc_file** (*str*) – the FITS data file name ('rc_n_iii_FSL13.fits')

atomneb.**read_aeff_n_ii_fsl13_references**(*atom_rc_file*)

This function returns the reference list of recombination coefficients (Aeff) from the 2nd binary table extension of the FITS data file ('rc_n_iii_FSL13.fits').

Returns This function returns the aeff_data_reference: { Reference:", Citation:"}

Return type an array of data

Parameters **atom_rc_file** (*str*) – the FITS data file name ('rc_n_iii_FSL13.fits')

atomneb.**read_aeff_o_ii_ssb17**(*atom_rc_file*, *atom*, *ion*, *case1*, *wavelength_range*, *wavelength=None*, *reference=None*)

This function returns the effective recombination coefficients (Aeff) from the table extensions of the FITS data file ('rc_o_iii_SSB17.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data-rc')
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_o_iii_SSB17_orl_
↪case_b.fits')
>> atom='o'
>> ion='iii' # O II
>> case1='B'
>> wavelength_range=[5320.0, 5330.0]
>> oii_rc_data = atomneb.read_aeff_o_ii_ssb17(atom_rc_file, atom, ion,␣
↪case1, wavelength_range)
>> oii_rc_data_wave = atomneb.read_aeff_o_ii_ssb17(atom_rc_file, atom,␣
↪ion,
>>                                                case1, wavelength_
↪range, wavelength=True)
>> print(oii_rc_data['aeff'][0])
   1.64100e-30  1.60000e-30  1.56400e-30  1.54100e-30 ...
>> n_line = len(oii_rc_data_wave['wavelength'])
>> for i in range(0, n_line):
>>      print(oii_rc_data_wave['wavelength'][i], oii_rc_data_wave['lower_
↪term'][i], oii_rc_data_wave['upper_term'][i])
   5327.172s22p2(1S)3p 2Po
   5325.422s22p2(1S)3p 2Po
   5327.182s22p2(1D)3d 2Ge
   ...
```

Returns This function returns the effective recombination coefficients.

Return type an array of data

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_o_iii_SSB17.fits')
- **atom** (*str*) – atom name e.g. 'o'
- **ion** (*str*) – ionic level e.g 'iii'
- **case1** (*str*) – set for the case 'a' or 'b', defualt 'b'
- **wavelength_range** (*array*) – wavelength range e.g. [5320.0, 5330.0]
- **wavelength** (*boolean, optional*) – set for returning the wavelengths
- **reference** (*string, optional*) – set for the reference

atomneb.**read_aeff_o_ii_ssb17_list**(*atom_rc_file*)

This function returns the list of effective recombination coefficients (Aeff) from the 1st binary table extension of the FITS data file ('rc_o_iii_SSB17.fits')

Returns This function returns the aeff_data_list: {Aeff_Data:", Extension:0, IND:long(0), Wavelength: float(0.0), Case1:", lower_term: '', upper_term: ''}

Return type an array of data

Parameters **atom_rc_file** (*str*) – the FITS data file name ('rc_o_iii_SSB17.fits')

atomneb.**read_aeff_o_ii_ssb17_references** (*atom_rc_file*)

This function returns the reference list of recombination coefficients (Aeff) from the 2nd binary table extension of the FITS data file ('rc_o_iii_SSB17.fits').

Returns This function returns the aeff_data_reference: { Reference:", Citation:"}

Return type an array of data

Parameters **atom_rc_file** (*str*) – the FITS data file name ('rc_o_iii_SSB17.fits')

atomneb.**read_aeff_ppb91** (*atom_rc_file*, *atom*, *ion*, *reference=None*)

This function returns the effective recombination coefficients (Aeff) from the table extensions of the FITS data file ('rc_PPB91.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data-rc')
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_PPB91.fits')
>> atom='c'
>> ion='iii' # C II
>> cii_rc_data = atomneb.read_aeff_ppb91(atom_rc_file, atom, ion)
>> n_line = len(cii_rc_data['wavelength'])
>> for i in range(0, n_line):
>>          print(cii_rc_data['ion'][i], cii_rc_data['case1'][i], cii_rc_
↪data['wavelength'][i],
>>                cii_rc_data['a'][i], cii_rc_data['b'][i], cii_rc_data[
↪'c'][i],
>>                cii_rc_data['d'][i], cii_rc_data['br'][i], cii_rc_data[
↪'q'][i], cii_rc_data['y'][i])
   C2+A       9903.4600      0.69700000     -0.78400000      ...
   C2+A       4267.1500       1.0110000     -0.75400000      ...
   ...
```

Returns This function returns the effective recombination coefficients: {Ion: ' ', Case1:", Wavelength:0.0, a: 0.0, b: 0.0, c: 0.0, d: 0.0, br: 0.0, y: 0.0}

Return type an array of data

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_PPB91.fits')
- **atom** (*str*) – atom name e.g. 'c'
- **ion** (*str*) – ionic level e.g 'iii'
- **reference** (*str, optional*) – set for the reference

atomneb.**read_aeff_ppb91_list** (*atom_rc_file*)

This function returns the list of effective recombination coefficients (Aeff) from the 1st binary table extension of the FITS data file ('rc_PPB91.fits')

Returns This function returns the aeff_data_list: { Aeff_Data:", Extension:0.0}

Return type an array of data

Parameters **`atom_rc_file`** (*`str`*) – the FITS data file name ('rc_PPB91.fits')

`atomneb.`**`read_aeff_ppb91_references`**(*atom_rc_file*)

This function returns the reference list of recombination coefficients (Aeff) from the 2nd binary table extension of the FITS data file ('rc_PPB91.fits').

Returns This function returns the aeff_data_reference: { Reference:", Citation:"}

Return type an array of data

Parameters **`atom_rc_file`** (*`str`*) – the FITS data file name ('rc_PPB91.fits')

`atomneb.`**`read_aeff_sh95`**(*atom_rc_file*, *atom*, *ion*, *reference=None*, *case1=None*)

This function returns the effective recombination coefficients (Aeff) from the table extensions of the FITS data file ('rc_SH95.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data-rc')
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_SH95.fits')
>> atom='h'
>> ion='ii' # H I
>> hi_rc_data = atomneb.read_aeff_sh95(atom_rc_file, atom, ion)
>> print(hi_rc_data['aeff'][0])
   100.00000       500.00000       0.0000000    4.2140000e-27    1.
↪7560000e-27   ...
   ...
```

Returns This function returns the effective recombination coefficients.

Return type an array of data

Parameters

- **`atom_rc_file`** (*`str`*) – the FITS data file name ('rc_SH95.fits')
- **`atom`** (*`str`*) – atom name e.g. 'h'
- **`ion`** (*`str`*) – ionic level e.g 'ii'
- **`reference`** (*`str, optional`*) – set for the reference
- **`case1`** (*`boolean, optional`*) – set for the case 'a' or 'b', defualt 'b'

`atomneb.`**`read_aeff_sh95_list`**(*atom_rc_file*)

This function returns the list of effective recombination coefficients (Aeff) from the 1st binary table extension of the FITS data file ('rc_SH95.fits')

Returns This function returns the aeff_data_list: { Aeff_Data:”, Extension:0.0}

Return type an array of data

Parameters **atom_rc_file** (*str*) – the FITS data file name (‘rc_SH95.fits’)

atomneb.**read_aeff_sh95_references** (*atom_rc_file*)

This function returns the reference list of recombination coefficients (Aeff) from the 2nd binary table extension of the FITS data file (‘rc_SH95.fits’).

Returns This function returns the aeff_data_reference: { Reference:”, Citation:”}

Return type an array of data

Parameters **atom_rc_file** (*str*) – the FITS data file name (‘rc_SH95.fits’)

atomneb.**read_aij** (*atom_aij_file*, *atom*, *ion*, *reference=None*, *level_num=None*)

This function returns the transition probabilities (Aij) from the table extensions of the FITS data file (‘AtomAij.fits’).

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data', 'collection')
>> atom_aij_file = os.path.join(base_dir,data_dir, 'AtomAij.fits')
>> atom='o'
>> ion='iii'
>> reference='FFT04'
>> oiii_aij_data = atomneb.read_aij(atom_aij_file, atom, ion, reference)
>> print(oiii_aij_data['aij'][0])
   0.0000000   2.5960000e-05   3.0300000e-11   2.3220000e-06       0.
↪0000000    0.0021910000
   0.0000000       0.0000000   9.6320000e-05    0.0069510000      0.
↪22550000       230.80000
   0.0000000       0.0000000       0.0000000     0.020290000   0.
↪00069980000       576.50000
   0.0000000       0.0000000       0.0000000       0.0000000       1.
↪6850000    0.0057770000
   0.0000000       0.0000000       0.0000000       0.0000000       0.
↪0000000   3.7600000e-11
   0.0000000       0.0000000       0.0000000       0.0000000       0.
↪0000000       0.0000000
```

Returns This function returns the aij_data: { Aij:dblarr(n_level,n_level) }.

Return type an array of data

Parameters

- **atom_rc_file** (*str*) – the FITS data file name (‘AtoAij.fits’)
- **atom** (*str*) – atom name e.g. ‘o’
- **ion** (*str*) – ionic level e.g ‘iii’
- **reference** (*str, optional*) – set for the reference
- **level_num** (*str, optional*) – set for the maximum level number

atomneb.**read_aij_list**(*atom_aij_file*)

This function returns the list of transition probabilities (Aij) from the 1st binary table extension of the FITS data file ('AtomAij.fits').

Returns This function returns the aij_data_list: { Aij_Data:'', Extension:0.0}

Return type an array of data

Parameters **Atom_Aij_file** (*str*) – the FITS data file name ('AtomAij.fits')

atomneb.**read_aij_references**(*atom_aij_file*)

This function returns the reference list of transition probabilities (Aij) from the 1nd binary table extension of the FITS data file ('AtomAij.fits').

Returns This function returns the aij_data_reference: { Reference:'', Citation:''}

Return type an array of data

Parameters **Atom_Aij_file** (*str*) – the FITS data file name ('AtomAij.fits')

atomneb.**read_elj**(*atom_elj_file*, *atom*, *ion*, *level_num=None*)

This function returns the energy levels (Ej) from the table extensions of the FITS data file ('AtomElj.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data', 'collection')
>> atom_elj_file = os.path.join(base_dir,data_dir, 'AtomElj.fits')
>> atom='o'
>> ion='iii'
>> oiii_elj_data=atomneb.read_elj(atom_elj_file, atom, ion, level_num=6)
>> print(oiii_elj_data['j_v'])
   0.00000      1.00000      2.00000      2.00000      0.00000      2.
↪00000
>> print(oiii_elj_data['ej'])
   0.0000000       113.17800       306.17400       20273.270       43185.
↪740       60324.790
```

Returns This function returns the elj_data:{ Configuration:'', Term:'', J:'', J_v:0.0, Ej:0.0, Reference:''}.

Return type an array of data

Parameters

- **atom_elj_file** (*str*) – the FITS data file name ('AtomElj.fits')
- **atom** (*str*) – atom name e.g. 'o'
- **ion** (*str*) – ionic level e.g 'iii'
- **level_num** (*int, optional*) – set for the maximum level number.

atomneb.**read_elj_list**(*atom_elj_file*)

This function returns the list of energy levels (Ej) from the 1st binary table extension of the FITS data file ('AtomElj.fits')

Returns This function returns the elj_data_list: { Elj_Data:", Extension:0.0}

Return type an array of data

Parameters **atom_elj_file** (*str*) – the FITS data file name ('AtomElj.fits')

atomneb.**read_elj_references**(*atom_elj_file*)

This function returns the reference list of energy levels (Ej) from the 2nd binary table extension of the FITS data file ('AtomElj.fits').

Returns This function returns the aij_data_reference: { Reference:", Citation:"}

Return type an array of data

Parameters **atom_elj_file** (*str*) – the FITS data file name ('AtomElj.fits')

atomneb.**read_omij**(*atom_omij_file*, *atom*, *ion*, *reference=None*, *level_num=None*)

This function returns the collision strengths (omega_ij) from the table extensions of the FITS data file ('AtomOmij.fits').

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data', 'collection')
>> atom_omij_file = os.path.join(base_dir,data_dir, 'AtomOmij.fits')
>> atom='o'
>> ion='iii'
>> reference='SSB14'
>> oiii_omij_data=atomneb.read_omij(atom_omij_file, atom, ion,␣
↪reference=reference)
>> print(oiii_omij_data['level1'])
   0        1        1        1        1        2        2        2        3       ␣
↪ 3        4
>> print(oiii_omij_data['level1'])
   0        2        3        4        5        3        4        5        4       ␣
↪ 5        5
>> print(oiii_omij_data['strength'][0])
   100.00000       125.89254       158.48932       199.52623       251.
↪18864       ...
```

Returns This function returns the omij_data: { level1:0, level2:0, strength:array(temp_steps)}.

Return type an array of data

Params

atom_omij_file [in, required, type=string] the FITS data file name ('AtomOmij.fits')

atom [in, required, type=string] atom name e.g. 'o'

ion [in, required, type=string] ionic level e.g 'iii'

Parameters

- **reference** (*str, optional*) – set for the reference e.g. 'SSB14'
- **level_num** (*int, optional*) – set for the maximum level number.

atomneb.**read_omij_list** (*atom_omij_file*)

This function returns the list of collision strengths (omega_ij) from the 1st binary table extension of the FITS data file ('AtomOmij.fits').

Returns This function returns the omij_data_list: { Omij_Data:", Extension:0.0}

Return type an array of data

Parameters **atom_omij_file** (*str*) – the FITS data file name ('AtomOmij.fits')

atomneb.**read_omij_references** (*atom_omij_file*)

This function returns the reference list of collision strengths (omega_ij) from the 2nd binary table extension of the FITS data file ('AtomOmij.fits').

Returns his function returns the aij_data_reference: { Reference:", Citation:"}

Return type an array of data

Parameters **atom_omij_file** (*str*) – the FITS data file name ('AtomOmij.fits')

atomneb.**search_aeff_collection** (*atom_rc_file*, *atom*, *ion*, *br=None*)

This function searches effective recombination coefficients (Aeff) for given element and ionic levels in the FITS data file ('rc_collection.fits'), and returns the data entry.

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data-rc')
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_collection.fits')
>> atom='c'
>> ion='iii' # C III
>> list_cii_aeff_data = atomneb.search_aeff_collection(atom_rc_file, atom,
↪ ion)
>> print(list_cii_aeff_data)
   c_iii_aeff
```

Returns This function returns the Aeff_Data.

Return type an array of data

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_collection.fits')
- **atom** (*str*) – atom name e.g. 'c'
- **ion** (*str*) – ionic level e.g 'iii'
- **br** (*boolean, optional*) – set for the branching ratios (Br), may not necessary

atomneb.**search_aeff_he_i_pfsd12** (*atom_rc_file*, *atom*, *ion*)

This function searches effective recombination coefficients (Aeff) for given element and ionic levels in the FITS data file ('rec_he_ii_PFSD12.fits'), and returns the data entry.

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data-rc')
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_he_ii_PFSD12.fits')
>> atom='he'
>> ion='ii' # He I
>> list_hei_aeff_data = atomneb.search_aeff_he_i_pfsd12(atom_rc_file,␣
↪atom, ion)
>> print(list_hei_aeff_data)
   he_ii_aeff_PFSD12 he_ii_aeff_PFSD13
```

Returns This function returns the Aeff_Data.

Return type an array of data

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_he_ii_PFSD12.fits')
- **atom** (*str*) – atom name e.g. 'he'
- **ion** (*str*) – ionic level e.g 'ii'

atomneb.**search_aeff_n_ii_fsl13**(*atom_rc_file*, *atom*, *ion*, *wavelength*)

This function searches effective recombination coefficients (Aeff) for given element and ionic levels in the FITS data file ('rc_n_iii_FSL13.fits'), and returns the data entry.

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data-rc')
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_n_iii_FSL13.fits')
>> atom='n'
>> ion='iii' # N II
>> wavelength=5679.56
>> list_nii_aeff_data = atomneb.search_aeff_n_ii_fsl13(atom_rc_file, atom,
↪ ion, wavelength)
>> print(list_nii_aeff_data['wavelength'])
   5679.56
>> print(list_nii_aeff_data['aeff'])
   7810.00       1780.00       850.000       151.000       74.4000       53.
↪1000       47.4000
   7370.00       1700.00       886.000       206.000       110.000       80.
↪1000       70.8000
   7730.00       1680.00       900.000       239.000       138.000       103.
↪000       92.9000
   8520.00       1710.00       905.000       244.000       142.000       107.
↪000       97.0000
```

Returns This function returns the Aeff_Data.

Return type an array of data

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_n_iii_FSL13.fits')
- **atom** (*str*) – atom name e.g. 'n'
- **ion** (*str*) – ionic level e.g 'iii'
- **wavelength** (*int*) – set the wavelengths
- **reference** (*str, optional*) – set for the reference

atomneb.**search_aeff_o_ii_ssb17** (*atom_rc_file*, *atom*, *ion*, *case1*, *wavelength*)

This function searches effective recombination coefficients (Aeff) for given element and ionic levels in the FITS data file ('rc_o_iii_SSB17.fits'), and returns the data entry.

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data-rc')
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_o_iii_SSB17_orl_
↪case_b.fits')
>> atom='o'
>> ion='iii' # O II
>> case1='B'
>> wavelength=5325.42
>> list_oii_aeff_data = atomneb.search_aeff_o_ii_ssb17(atom_rc_file, atom,
↪ ion, case1, wavelength)
>> print(list_oii_aeff_data['wavelength'])
   5325.42
>> print(list_oii_aeff_data['aeff'])
   3.41800e-32  3.33300e-32  3.25700e-32  3.20900e-32  3.16800e-32 ...
```

Returns This function returns the Aeff_Data.

Return type an array of data

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_o_iii_SSB17.fits')
- **atom** (*str*) – atom name e.g. 'o'
- **ion** (*str*) – ionic level e.g 'iii'
- **case1** (*str*) – set for the case 'a' or 'b', defualt 'b'
- **wavelength** (*float*) – set the wavelengths

atomneb.**search_aeff_ppb91** (*atom_rc_file*, *atom*, *ion*)

This function searches effective recombination coefficients (Aeff) for given element and ionic levels in the FITS data file ('rec_PPB91.fits'), and returns the data entry.

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data-rc')
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_PPB91.fits')
>> atom='c'
>> ion='iii'
>> list_cii_aeff_data = atomneb.search_aeff_ppb91(atom_rc_file, atom, ion)
>> print(list_cii_aeff_data)
   c_iii_aeff
```

Returns This function returns the Aeff_Data.

Return type an array of data

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_PPB91.fits')
- **atom** (*str*) – atom name e.g. 'c'
- **ion** (*str*) – ionic level e.g 'iii'

atomneb.**search_aeff_sh95**(*atom_rc_file*, *atom*, *ion*)

This function searches effective recombination coefficients (Aeff) for given element and ionic levels in the FITS data file ('rec_SH95.fits'), and returns the data entry.

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data-rc')
>> atom_rc_file = os.path.join(base_dir,data_dir, 'rc_SH95.fits')
>> atom='h'
>> ion='ii' # H I
>> list_hi_aeff_data = atomneb.search_aeff_sh95(atom_rc_file, atom, ion)
>> print(list_hi_aeff_data)
   h_ii_aeff_a h_ii_aeff_b
```

Returns This function returns the Aeff_Data.

Return type an array of data

Parameters

- **atom_rc_file** (*str*) – the FITS data file name ('rc_SH95.fits')
- **atom** (*str*) – atom name e.g. 'h'
- **ion** (*str*) – ionic level e.g 'ii'

atomneb.**search_aij**(*atom_aij_file*, *atom*, *ion*)

This function searches transition probabilities (Aij) for given element and ionic levels in the FITS data file ('AtomAij.fits'), and returns the data entry.

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data', 'collection')
>> atom_aij_file = os.path.join(base_dir,data_dir, 'AtomAij.fits')
>> atom='o'
>> ion='iii'
>> list_oiii_aij_data = atomneb.search_aij(atom_aij_file, atom, ion)
>> print(list_oiii_aij_data)
   o_iii_aij_FFT04-SZ00 o_iii_aij_FFT04 o_iii_aij_GMZ97-WFD96 o_iii_aij_
↪SZ00-WFD96
```

Returns This function returns the Aij_Data.

Return type an array of data

Parameters

- **Atom_Aij_file** (*str*) – the FITS data file name (‘AtomAij.fits’)
- **atom** (*str*) – atom name e.g. ‘o’
- **ion** (*str*) – ionic level e.g ‘iii

atomneb.**search_omij**(*atom_omij_file*, *atom*, *ion*)

This function searches collision strengths (omega_ij) for given element and ionic levels in the FITS data file (‘AtomOmij.fits’), and returns the data entry.

For example:

```
>> import atomneb
>> import numpy as np
>> import os
>> base_dir = os.getcwd()
>> data_dir = os.path.join('..','atomic-data', 'collection')
>> atom_omij_file = os.path.join(base_dir,data_dir, 'AtomOmij.fits')
>> atom='o'
>> ion='iii'
>> list_oiii_omij_data = atomneb.search_omij(atom_omij_file, atom, ion)
>> print(list_oiii_omij_data)
   o_iii_omij_AK99 o_iii_omij_LB94 o_iii_omij_Pal12-AK99 o_iii_omij_SSB14
```

Returns This function returns the Omij_Data.

Return type an array of data

Parameters

- **atom_omij_file** (*str*) – the FITS data file name (‘AtomOmij.fits’)
- **atom** (*str*) – atom name e.g. ‘o’
- **ion** (*str*) – ionic level e.g ‘iii’

CHAPTER

SIX

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX