SpecMatch Synth Documentation

Release b1

Erik Perigura and BJ Fulton

Nov 14, 2017

Contents

Contents:

CHAPTER 1

Installation Instructions

1. Install dependencies: *pandas*, *astropy*
2. Download `specmatch-synth` from git repo
3. Run `python setup.py install` from within the main repo directory

CHAPTER 2

Quickstart Tutorial

1.

CHAPTER 3

API

Contents:

3.1 Model Library Grid

Module to define the Library class

This module defines the Library object used for specmatch-synth

class smsyn.library.**Library**(*header*, *model_table*, *wav*, *model_spectra*, *wavlim=None*)

The Library object

This object handles reading the model grid and associating the models with stellar parameters

Args:

header (dict): a dictionary containing metadata that describes the model library. 'model_name' and 'model_reference' are the only required keys. (e.g. {'model_name': 'coelho05', 'model_reference': 'Coelho et al. (2005)'})

model_table (DataFrame): Pandas DataFrame with the following columns: teff, logg, fe, model_index. The model_index column should give the index to the model spectrum in the *model_spectra* array that is associated with the given parameters.

wav (array): 1-d vector containng the wavelength scale for the model spectra

model_spectra (array): array containing all model spectra ordered so that the they can be referenced by the indicies contained in the *model_table*.

wavlim (2-element iterable): (optional) list, tuple, or other 2-element itarable that contains the upper and lower wavelengths limits to be read into memory

select_model(*pars*)

Select a model spectrum

Grab a model spectrum from the library that corresponds to a given set of stellar parameters.

Args:

pars (3-element iterable): A 3-element tuple containing teff, logg, and fe

Returns: array: model spectrum flux resampled at the new wavelengths

synth (*wav*, *teff*, *logg*, *fe*, *vsini*, *psf*, *rotation='rot'*, *interp_kw=None*)

Synthesize a model spectrum

For a given set of wavelengths teff, logg, fe, vsini, psf, compute a model spectrum by:

1. Determine the 8 coelho models surounding the (teff,logg,fe)
2. Perform trilinear interpolation
3. Resample onto new wavelength scale
4. Broaden with rot-macro turbulence
5. Broaden with PSF (assume gaussian)

Args: wav (array): wavelengths where the model will be calculated teff (float): effective temp (K) logg (float): surface gravity (logg) fe (float): metalicity [Fe/H] (dex) vsini (float): rotational velocity (km/s) psf (float): sigma for instrumental profile (pixels)

Returns:

array: synthesized model calculated at the wavelengths specified in the wav argument

to_hdf (*outfile*)

Save model library

Save a model library as an h5 file

Args: outfile (string): path to output h5 file

smsyn.library.**read_hdf** (*filename*, *wavlim=None*)

Read model library grid

Read in a model library grid from an h5 file and initialze a Library object.

Args:

filename (string): path to h5 file that contains the grid of stellar atmosphere models

wavlim (2-element iterable): upper and lower wavelength limits (in Angstroms) to load into RAM

Returns: Library object

smsyn.library.**trilinear_interp** (*c*, *v0*, *v1*, *vi*)

Trilinear interpolation

Perform trilinear interpolation as described here. http://en.wikipedia.org/wiki/Trilinear_interpolation

Args:

c (8 x n array): where C each row of C corresponds to the value at one corner

v0 (length 3 array): with the origin v1 (length 3 array): with coordinates on the diagonal vi (length 3 array): specifying the interpolated coordinates

Returns: interpolated value of c at vi

3.2 Spectrum Object

Defining Spectrum Class

class smsyn.io.spectrum.**Spectrum**
A light superclass on top of numpy record array that stores header information and and read and write to fits objects.

Args:

wav (array): wavelengths corresponding to each pixel in the flux array

flux (array): continuum-normalized flux as a function of rest wavelength

uflux (array): relative flux uncertainty header (dict): dictionary containing metadata associated with the

observed spectrum. Similar to a header from a fits file. Required keys: object, observation

to_fits(*outfile*, *clobber=True*)
Save to FITS

Save a Spectrum object as a mutli-extension fits file.

Args: outfile (string): name of output file name clobber (bool): if true, will overwrite existing file

smsyn.io.spectrum.**read_fits**(*filename*)
Read spectrum from fits file

Read in a spectrum as saved by the Spectrum.to_fits method into a Spectrum object

Args: filename (string): path to fits file

Returns: Spectrum object

3.3 Match Object

This module defines the Match class that is used in fitting routines.

class smsyn.match.**Match**(**args*, ***kwargs*)

masked_nresid(*params*, ***kwargs*)
Masked normalized residuals

Return the normalized residuals with masked wavelengths excluded

Args: params (lmfit.Parameters): see params in self.model

Returns: array: normalized residuals where self.wavmask == 1

model(*params*, *wav=None*, ***kwargs*)
Calculate model

Return the model for a given set of parameters

Args:

params (lmfit.Parameters): Parameters object containing at least teff, logg, fe, vsini, psf, and spline coefficients

wav (array): (optional) array of wavelengths at which to calculate the model. Useful for generating a more finely sampled model for plotting

**kwargs: extra keyword arguments passed to lib.synth

nresid(*params*, ***kwargs*)
Normalized residuals

Args: params (lmfit.Parameters): see params in self.model

Returns: array: model minus data divided by errors

resid(*params*, ***kwargs*)
Residuals

Return the residuals

Args: params (lmfit.Parameters): see params in self.model

Returns: array: model minus data

spline(*params*, *wav*)
Continuum model

Unpacks the params object and returns a spline evaluated at specified wavelengths.

Args: params (lmfit.Parameters): See params in self.model wav: array of wavelengths at which to calculate the continuum model.

Returns: array: spline

CHAPTER 4

Indices and tables for Python code

- genindex
- modindex
- search

Python Module Index

S

Index