1. Introduction note

This purpose of this package is to make publicly available the supernova (SN) rate models used in Heringer et al. (2017) to compute the likelihood of parametrized delay time distributions (DTD) and assess the most likely DTD for a sample of galaxies and hosts surveyed by SDSS in the Stripe 82 region. Pertinent references are available in the original publication.

2. Installation requirements

It is suggested that a conda environment is created containing the packages specified under **env_specs.txt** (located at the top level of this repository). This package has only been tested under Ubuntu 17.10.

The codes here may be used as standalone routines, but they may depend on data produced by FSPS (v3.0) computed through the Python-FSPS package. A few test cases have been uploaded to this repository so that the user may bypass the installation of these packages.

If the user wishes to explore models which include star formation histories (SFH) or filters different than provided in the test case, the installation of both FSPS and Python-FSPS will be required. Note that for the user to be able to create new FSPS files, an older version (6.4.0) of the gfortran compiler will also be required, since the default version (7.2.0) conflicts with Python-FSPS. Installing gfortran 6 can be done via: **sudo apt-get install gfortran-6**. Then it is necessary to change the compiler set to run FSPS in the **$SPS_HOME/src/Makefile** so that **F90 = gfortran − > F90 = gfortran-6**.

A few notes on the installation of FSPS and Python-FSPS are given in §4.

3. Producing SN rate models

The routines available here can be called through by running `python master.py`. The relevant input parameters are set in the `input_params.py` file.

The default flags in `master.py` and the default parameters in `input_params.py` should allow for the master code to run even if the FSPS and Python-FSPS packages are not installed.

Note that at this moment, likelihoods calculations (which depend on the observational data) are only available if the selected filters are `filter_1 = ‘‘sdss_r’’` and `filter_2 = ‘‘sdss_g’’`. If FSPS and Python-FSPS are installed, the user may still build SN rate models for other colors.

3.1. input_params.py

This file stores the input parameters that are used by the `master.py` script. Specific combinations of parameters are stored as `cases`.

- filter_1 (str): Filter_1 and filter_2 determine the color to be used as (filter_2 - filter_1). A list of available filters can be shown by calling fsps.list_filters() under run_fsps.py
- filter_2 (str): Same as above.
- imf_type (str): Choice of initial mass function for the FSPS simulations. Accepts: `Salpeter`, `Chabrier` or `Kroupa`.
- sfh_type (str): Choice of star formation history for the FSPS simulations. Accepts: `exponential` (SFR $\propto e^{-t/\tau}$) or `delayed-exponential` (SFR $\propto t \times e^{-t/\tau}$).

- Z (float): Choice of metallicity for the FSPS simulations. Accepts 0.0002, 0.0003, 0.0004, 0.0005, 0.0006, 0.0008, 0.0010, 0.0012, 0.0016, 0.0020, 0.0025, 0.0031, 0.0039, 0.0049, 0.0061, 0.0077, 0.0096, 0.0120, 0.0150, 0.0190, 0.0250 or 0.0300.
- t_onset (astropy float (unit of time)): Sets the time past which SN may occur. Determined by the lifetime of stars that may contribute to the SN rate.
- t_cutoff (astropy float (unit of time)): Sets the time at which the slope of the DTD may change. Determined by theoretical models.
- Dcolor_min (float): Sets the lower Dcolor (color with respect to the red sequence) limit, below which galaxies are not taken into account. Set by the typical Dcolor at which different SFH may not converge in the sSNRL models.
- Dcolor_max (float): Dcolor cut to explode (a few) potential outliers that are much redder that the red sequence. Originally determined by the uncertainty in fitting the red sequence.
- slopes (np.array): Numpy array containing which DTD slopes to use to compute likelihoods. This package adopts the same array for slopes pre and post cutoff time.
- tau_list (astropy array (unit of time)): List containing the tau timescales for the selected SFH. E.g.: tau_list = np.array([1., 1.5, 2., 3., 4., 5., 7., 10.]) * 1.e9 * u.yr
- subdir (str): Name of the sub-directory where the outputs will be stored. For organization purposes only. Refereed henceforth as `$subdir`.

Note that a side routine (`util_tasks.py`) will create a LOG containing the input parameters for that run at:

```
$INSTALLATION_DIR/OUTPUT_FILES/RUNS/$subdir/record.dat
```

3.2. master.py

This code will call three routines to be executed:

1) Make_FSPS: controlled by `run_fsps_flag`.

- True: New FSPS files will be computed according to the parameters in `input_params.py`. REQUIRES FSPS AND PYTHON-FSPS TO BE INSTALLED.
- False: Pre-made FSPS files will be used, but this option is only available for the combination of `filter_1 = sdss_r, filter_2 = sdss_g, imf_type=Chabrier, Z=0.0190 and tau_list=[1, 1.5, 2, 3, 4, 5, 7, 10] Gyr`. A warning will be issued and these variables will be reset to these values. `sfh_type` may be `exponential` or `delayed-exponential`.

The synthetic stellar population files are stored at:

`$INSTALLATION_DIR/OUTPUT_FILES/RUNS/$subdir/fsps_FILES/`

2) Get_Likelihood and Plot_Likelihood: controlled by `likelihood_flag`.

- True: the routine will use the computed models and observed data to compute the likelihood of a set of DTD slopes. These likelihoods will then be plot as intensity as a function of the DTD slope pre and post cutoff. REQUIRES `filter_1 = sdss_r, filter_2 = sdss_g`. This may take several minutes.
- False: these tasks will simply be skipped.

The computed likelihoods are stored at:

`$INSTALLATION_DIR/OUTPUT_FILES/RUNS/$subdir/likelihood.csv`

3)Make_Panels: controlled by `panels_flag`.

- True: the routine will create several plots with panels showing the relationship between multiple relevant variables, such as age, color, mass formed and SN rates.
- False: this task will simply be skipped.

The produced plots are stored at:

`$INSTALLATION_DIR/OUTPUT_FILES/RUNS/$subdir/FIGURES/PANELS/`

4. Optional: installing FSPS and Python-FSPS

4.1. FSPS

Source code available at https://github.com/cconroy20/fsps. Pertinent references are Conroy et al. (2009) and Conroy et al. (2010).

Installation requires setting the `SPS_HOME` variable at the bash file. e.g.: `export SPS_HOME=‘‘...’’`

Note 1) For Python-FSPS to compile, it might be necessary to change the following at `$SPS_HOME/src/Makefile`: `F90FLAGS = -O -cpp` − > `F90FLAGS = -O -cpp -fPIC`

Note 2) The models computed in Heringer et al. (2017) adopted the *BaSeL* spectral library, which needs to be changed at `$SPS_HOME/src/sps_vars.f90`: `define MILES 1 define BASEL 0` − > `define MILES 0 define BASEL 1`

Finally, type `make` at the `$SPS_HOME/src` directory.

4.2. Python-FSPS

Source code available at https://github.com/dfm/python-fsps.

In the main directory, type `python setup.py install`.

REFERENCES

Conroy, C., Gunn, J. E., & White, M. 2009, ApJ, 699, 486

Conroy, C., White, M., & Gunn, J. E. 2010, ApJ, 708, 58

Heringer, E., Pritchet, C., Kezwer, J., et al. 2017, ApJ, 834, 15