Manual for FSPS v2.4

1. Overview

The collection of fortran routines contained in this package allows the user to compute simple stellar populations (SSPs) for a variety of IMFs and metallicities, and for a variety of assumptions regarding the morphology of the horizontal branch, the blue straggler population, the post–AGB phase, and the location in the HR diagram of the TP-AGB phase. A variety of simple and flexible prescriptions for attenuation by dust are also included, as are dust emission models based on the Draine & Li 2007 dust models. From these SSPs the user may then generate composite stellar populations (CSPs) for a variety of star formation histories (SFHs) and dust attenuation prescriptions. Outputs include the spectra and magnitudes of the SSPs and CSPs at arbitrary redshift. In addition to these fortran routines a collection of IDL routines are provided that allow easy manipulation of the output.

The user is strongly encouraged to read Conroy et al. 2009 for an overview of stellar population synthesis (SPS) and for details regarding this collection of routines. As of v2.0, this code has been extensively calibrated against a suite of observational data (for details see Conroy & Gunn 2010). The code package is a in essence a highly flexible SPS code and has therefore come to be called FSPS.

The rest of this manual is organized as follows. In §2 we briefly discuss the overall philosophy of the code structure. In §3 we present the routines, how they are used, and provide the user with a basic program that demonstrates the use of the routines. §4 discusses the IDL routines provided to read and manipulate the outputs from the fortran package. In §5 we discuss a variety of questions (ok, only one) the user may have.

For a description of revisions since the initial release of the code, see the file REVISION_HISTORY in the `doc` directory. Installation instructions are also provided in that directory.

1.1 Downloading the code

As of v2.2, FSPS is now kept in an SVN repository. In order to download the entire package, simply type the following at the command line: `svn checkout http://fsps.googlecode.com/svn/trunk/ fsps`

If you are only interested in the SSPs, a few standard SSP sets can be downloaded by executing the command: `svn checkout http://fsps.googlecode.com/svn/trunk/SSP/ fsps_ssp`

One of the features of having FSPS in SVN is that future updates can be easily downloaded by executing the command `svn update`. This will only download the differences between your local version and the updated version. Emails will be sent to the mailing list whenever new versions become available (this is one reason why it is essential for users of this code to be on the mailing list).

1.2 Environment variables

As mentioned on the webpage, you will need to set an environment variable called SPS_HOME that points to the root SPS directory (i.e. the directory that contains the `src`, `OUTPUTS`, etc. directories)

1.3 Inclusion of TP-AGB spectra

The TP-AGB empirical spectral library does not extend past the rest-frame K-band and so in previous versions the integrated spectra were not reliable beyond $\lambda \approx 2.4\mu m$. Up to v2.2 the empirical TP-AGB spectra were extrapolated with the BaSeL library for models at the same $T_{\rm eff}$ and the lowest surface gravities available in the BaSeL library. As of v2.3, the TP-AGB spectra are extrapolated blueward with simpler linear slopes (as advocated in Lancon & Wood 2002). The spectra of the carbon stars are now extrapolated redward with the Aringer et al. 2009 synthetic carbon star spectral library. The spectra of the oxygen-rich TP-AGB stars are now extrapolated with the latest version of the PHOENIX stellar spectral library (the BT-SETTL library).

1.4 Units

This code produces two types of files. The first are spectral files (*.spec). The spectra are in units of $L_\odot$/Hz, i.e. they are f_ν. Integrating over frequency generates the bolometric luminosity of the spectrum. Wavelengths are in angstroms *in vacuum*, and the wavelength array for the stellar libraries can be found in the files `BaSeL3.1/basel.lambda` and `MILES/miles.lambda`. The full wavelength array is also now printed

in the first line in the *.spec files. The second type of file contains magnitudes in a variety of filters. The magnitude zero point (i.e., AB or Vega) is set by the variable `compute_vega_mags` described below. Both of these output files contain information on the age, mass, bolometric luminosity, and star formation rate. All of these quantities are in the log (base-10), with units of years, $M_\odot$, $L_\odot$, and $M_\odot$ yr^{-1}, respectively.

1.5 Redshifting

The code allows the user to specify a redshift (see Section 3.1.2 below). As of v2.4, specifying a non-zero redshift affects *both* the computed magnitudes and spectra. In previous versions the spectrum was always returned in the restframe. There are two qualitatively different options for redshifting. See the parameter `redshift_colors` for more details. Notice that only the $(1 + z)$ stretching of the spectrum is taken into account when the user specifies a non-zero redshift. Other effects, such as the luminosity dimming for cosmological sources, is not taken into account.

1.6 Filters

The current release contains 105 filters. The filter names can be found in the file `FILTER_LIST` in the `data` directory. The actual filter definitions are in the file `allfilters.dat` in the same directory. The transmission profiles in this file include atmospheric absorption and are in units of relative response per photon (as opposed for example to relative response per unit power). The filters are normalized internally within the code. The output magnitudes are listed in the filter order displayed in the `FILTER_LIST` file. The user is cautioned to use whenever possible the exact filter transmission curve appropriate for the data being considered. There is, for example, no such thing as THE $B-$band filter. Differences of a few hundredths of a magnitude are common between different definitions of a given filter such as B.

Warning: Between v2.3 and v2.4 many new filters have been added (mostly *HST* filters) and the ordering of the filters has completely changed. Please keep this in mind when reading the *mags files! If you use the IDL routine `read_mags.pro` provided with FSPS then you do not need to worry about the filter reordering.

1.6.1 Computation of magnitudes

The AB magnitude through a filter b, m_b, is defined according to the following formuale:

$$\langle f_\nu \rangle_b = \frac{\int R_\gamma^b \, f_\nu \, \mathrm{d}\ln\nu}{\int (\nu/\nu_b)^\beta \, R_\gamma^b \, \mathrm{d}\ln\nu} \quad (1)$$

$$m_b = -2.5\log_{10}(\langle f_\nu \rangle_b) - 48.60 \quad (2)$$

where f_ν is the spectrum and R_γ^b is the relative response per photon of the filter. The factor $(\nu/\nu_b)^\beta$ in the demoninator of Equation 1 may surprise those used to working with optical photometry. Indeed, for UV, optical, and near-IR photometry, one usually adopts $\beta = 0$ (e.g., for the GALEX, SDSS, and 2MASS surveys). However, IR photometry typically assumes a different calibration. For example, the *IRAS*, *Spitzer* IRAC, and *Herschel* PACS and SPIRE magnitudes are frequently quoted assuming $\beta = 1$ while the *Spitzer* MIPS filters adopt $\beta = 2$ (i.e., a 10^4 K blackbody). The parameter ν_b is the central wavelength of the filter.

The motivation underlying this convention is that for sources with an intrinsic spectral slope β, the bandpass-convolved flux quoted at frequency ν_b will be precisely the flux at that frequency. For example, observing a 10^4K blackbody through the MIPS filters with the above calibration will return a flux density at the central frequency of the filter that is precisely the true flux at ν_b. In the end, this is merely a convention, but one that must be handled carefully when comparing data to models. See the routine `getmags.f90` for further details.

1.7 Interpretation of the Δ_L and Δ_T parameters

As described below, the user may modify the bolometric luminosity and effective temperature of the TP-AGB phase by applying overall shifts in $\log(L_{\rm bol})$ and $\log(T_{\rm eff})$ via the parameters Δ_L and Δ_T. In previous versions, these parameters were with respect to the default Padova model calculations circa 2008. As of v2.0, these parameters represent shifts with respect to the best-fit values found in Conroy & Gunn

2010. In other words, leaving these values set to 0.0 means that the user adopts the calibrations described in Section 3.1.3 in Conroy & Gunn 2010.

As of v2.3, these parameters have *not* been updated to reflect the results presented in Kriek et al. 2010.

1.8 Dust emission model

A new feature in FSPS as of v2.3 is the option of including a model for dust emission (see the parameter `add_dust_emission` below). Since there is currently no publication discussing this aspect of FSPS, a few words regarding the details and implementation of the dust emission model are in order.

We have adopted the dust emission model of Draine & Li 2007 (DL07), which is a silicate-graphite-PAH grain model. The model produces dust emission spectra from $1 - 10^4 \mu m$ as a function of the interstellar radiation field, U, expressed in units of the Milky Way radiation field. DL07 advocate constructing spectra for entire galaxies by summing up emission spectra over a range of radiation field strengths, $P(U)dU$ approximated by a delta function at $U_{\rm min}$ and a power-law component from $U_{\rm min} < U \leq U_{\rm max}$. As suggested by DL07, we have fixed $U_{\rm max} = 10^6$ and the power-law slope to be $\alpha = 2.0$. We therefore have: $P(U)dU = (1-\gamma)\delta(U - U_{\rm min}) + \gamma A U^{-2}$, where $(1-\gamma)$ is the fraction of dust mass exposed to starlight intensity $U_{\rm min}$ and A is a normalization constant. The DL07 model thus has three parameters, $U_{\rm min}$, γ, and $q_{\rm PAH}$, the latter parameter being the PAH fraction. These three parameters are contained in the parameter set and are called `duste_umin`, `duste_gamma`, `duste_qpah`. The user can define each of these parameters in FSPS.

2. How to use this code, generally speaking

The code provided in this package is optimally designed to be integrated into a larger fortran program. The code reads all of the libraries into memory and then utilizes various routines to compute SSPs and CSPs. The benefit of the structure of this package is that it allows the user to produce very large numbers of models relatively quickly (e.g., $\sim 10^5$ models in 30 minutes on a 2.66 GHz Intel processor). It also allows one to easily integrate SPS into particular science tasks.

For those more interested in generating quick results, rather than using the more flexible aspects of the code, we also provide a routine that generates results after prompting the user for input. This program is called `autosps.exe`. Note that all outputs are placed in the `OUTPUTS` directory, included in the tarball, by default.

Recently, several users have developed Python interfaces to FSPS, including Dan Foreman-Mackey (`https://github.com/dfm/python-fsps`), and Jonathan Sick (`https://github.com/jonathansick/pySPS`).

3. Details of Fortran Routines

3.1 The `sps_vars.f90` Module: Common Variables, Parameters, and Structure Set-up

It is recommended that the user read through the `sps_vars.f90` module. This module must be called at the beginning of every program that uses the routines described below. It sets up two types of variables: common and parameters. Both parameters and common variables can be seen by all routines and thus do not have to be explicitly passed during a call to a routine. The difference between the two is that the parameters are defined once in `sps_vars.f90` and thereafter cannot be changed. The common variables, by contrast, can be changed by various routines (see § 3.1.1). Be careful when changing common variables! The variables in this module are well-documented. Most parameters should not be changed.

The `sps_vars.f90` module also sets up structures (for those who are not familiar with fortran structures, they are similar in many ways to structures in IDL). There are two main structures that are used extensively in many routines. One of these is defined to conveniently handle the output of the `compsp.f90` routine; it should not be of much concern for most users. The second structure defines the parameter set. It is sufficiently important to warrant a more detailed discussion (see § 3.1.2).

3.1.1 Defining the isochrones and stellar libraries

At the top of the `sps_vars.f90` module there are lines that start with `#define`. These lines set switches that tell the code to compile certain portions of the module depending on which switches are set. The switches are binary, with '1'=on and '0'=off. So for example in the standard release of `sps_vars.f90` you will see:

```
#define BASEL 1
#define MILES 0
#define PADOVA 1
#define BASTI 0
```

which tells the code to use the BaSeL stellar library and the Padova isochrones. It is very important that you 'make clean' every time you change one of these switches.

3.1.1 Common Variables & Parameters

We briefly describe several of the most important and most used common variables and parameters. These parameters are defined in `sps_vars.f90`.

- `compute_vega_mags`

A switch that sets the zero points of the magnitude system:

- `compute_vega_mags`

A switch that sets the zero points of the magnitude system:

  - **0:** AB system
  - **1:** Vega system

- `dust_type`

Common variable defining the extinction curve for dust around old stars:

  - **0:** power–law with index dust_index set in the parameter structure
  - **1:** Milky Way extinction law (with $R \equiv A_V/E(B-V)$ value defined in parameter set; see below) parameterized by Cardelli et al. 1989.
  - **2:** Calzetti et al. 2000 attenuation curve. Note that if this value is set then the dust attenuation is applied to all starlight equally (not split by age), and therefore the only relevant parameter is `dust2` (defined below), which sets the overall normalization.
  - **3:** allows the user to access a variety of attenuation curve models from Witt & Gordon 2000. See the parameters `wgp1` and `wgp2` in §3.1.2.

- `add_dust_emission`

Parameter to turn on/off the dust emission model of Draine & Li 2007.

- `add_stellar_remnants`

Parameter to turn on/off the inclusion of stellar remnants in the calculation of stellar masses.

- `imf_type`

Common variable defining the IMF type:

  - **0:** Salpeter 1955
  - **1:** Chabrier 2003

- **2:** Kroupa 2001
- **3:** van Dokkum 2008
- **4:** Dave 2008
- **5:** tabulated piece-wise power-law IMF, specified in `imf.dat` file located in the `data` directory (or specified via the parameter `imf_filename`; see below).

- `redshift_colors`
  - **0:** Magnitudes are computed at a fixed redshift specified in the parameter set (see below)
  - **1:** Magnitudes are computed at a redshift that corresponds to the age of the output SSP/CSP (assuming a redshift–age relation appropriate for a WMAP5 cosmology). This switch is useful if the user wants to compute the evolution in *observed* colors of a SSP/CSP.

- `time_res_incr`

Parameter setting the factor by which the resolution of the default isochrone age array is increased. Default value is 2, which is sufficient for most purposes.

- `tuniv`

Common variable setting the age of the Universe. As of v2.1 this variable is set internally via the routine `get_tuniv`, and is determined via the cosmological parameters hard-wired into that routine. *As of v2.3 this variable is determined internally based on the specified cosmological parameters.*

- `verbose`

Parameter that controls output to screen. If set to 1 a lot of output will be printed to screen, if set to 0 the programs will be silent.

3.1.2 The Parameter Set

The parameter set is a structure defined in the `sps_vars.f90` module. It must be defined at the beginning of every program for the various routines to properly work. This structure acts as the primary interface between what the user would like to compute and the various subroutines that actually do the work. Simply defining the structure at the beginning of the program will set all structure elements to their default values (see the example program `simple.f90`). The elements of the structure, and the default values, are described below. The parameters are organized by topic, e.g., star formation history, IMF, dust, and stellar evolution parameters.

Basic Parameters

- `zred` Redshift. If this value is non-zero and if `redshift_colors`= 1, the magnitudes will be computed for the spectrum placed at redshift `zred`. *Default value is 0.0.*
- `zmet` Metallicity. The metallicity is specified as an integer ranging between 1 and 22 for the Padova isochrones and between 1 and 10 for the BaSTI isochrones. A lookup table for the actual metallicities corresponding to the integer values is provided at the end of this manual. Note that $Z_{\odot} = 0.0190$. *Default value is 1.*
- `imf1` Logarithmic slope of the IMF over the range $0.08 < M < 0.5\,M_{\odot}$. Only used if `imf_type`= 2. *Default value is 1.3.*
- `imf2` Logarithmic slope of the IMF over the range $0.5 < M < 1.0\,M_{\odot}$. Only used if `imf_type`= 2. *Default value is 2.3.*
- `imf3` Logarithmic slope of the IMF over the range $1.0 < M < 100\,M_{\odot}$. Only used if `imf_type`= 2. *Default value is 2.3.*
- `vdmc` IMF parameter defined in van Dokkum 2008. Only used if `imf_type`= 3. *Default value is 0.08.*
- `mdave` IMF parameter defined in Dave 2008. Only used if `imf_type`= 4. *Default value is 0.5.*
- `evtype` Only include isochrone points at a particular evolutionary phase specified in the isochrone table (see `data/ev_phases.tex` for phase options). Only available with the BaSTI isochrone option. All phases used when set to -1. *Default value is -1.*
- `masscut` Only include masses above `masscut` in the synthesis. *Default value is 150.0.*
- `imf_filename` Filename of the tabulated IMF file located in the `data` directory. If unset, the default `imf.dat` is assumed. Only applies if `imf_type`= 5.

SFH Parameters

- `sfh` defines the type of star formation history, normalized such that one solar mass of stars is formed over the full SFH. *Default value is 0.*
  - **0:** Compute an SSP
  - **1:** Compute a five parameter SFH, with parameters `tau`, `const`, `sf_start`, `tburst`, and `fburst` (see below).
  - **2:** Compute a tabulated SFH defined in a file called `sfh.dat` that must reside in the `data` directory (or a file specified via the parameter `sfh_filename`; see below). The file must contain three rows. The first column is time since the Big Bang in Gyr, the second is the SFR in units of solar masses per year, the third is the absolute metallicity. An example is provided in the `data` directory. The time grid in this file can be arbitrary (so long as the units are correct), but it is up to the user to ensure that the tabulated sfh is well-sampled so that the outputs are stable. Obviously, highly oscillatory data require dense sampling.

  - **3:** Reserved for special use of the tabulated SFH option. Allows the user to read in the tabulated SFHs directly into the necessary arrays, bypassing the need to create sfh.dat files. Email conroy@ucsc.edu if you would like further instructions for how to use this option.
  - **4:** Delayed tau-model. This is the same as option 1 except that the tau-model component takes the form $t\,e^{-t/\tau}$.
  - **5:** Currently under development.

- `tau` Defines e-folding time for the SFH, in Gyr. Only used if `sfh`= 1 or 4. The range is $0.1 < \tau < 10^2$.*Default value is 1.0.*

- `const` Defines the constant component of the SFH. This quantity is defined as the fraction of mass formed in a constant mode of SF; the range is therefore $0 \leq C \leq 1$. Only used if `sfh`= 1 or 4. *Default value is 0.0.*

- `sf_start` Start time of the SFH, in Gyr. *Default value is 0.0.*

- `sf_trunc` Truncation time of the SFH, in Gyr. If set to 0.0, there is no trunction. Only used if `sfh`= 5. *Default value is 0.0.*

- `tage` If set to a non-zero value, the `compsp` routine will compute the spectra and magnitudes only at this age, and will therefore only output one age result. The units are Gyr. (The default is for `compsp` to compute and return results from $t \approx 0$ to the maximum age in the isochrones). *Default value is 0.0.*

- `fburst` Defines the fraction of mass formed in an instantaneous burst of star formation. Only used if `sfh`= 1 or 4. *Default value is 0.0.*

- `tburst` Defines the age of the Universe when the burst occurs. If `tburst`>`tage` then there is no burst. Only used if `sfh`= 1 or 4. *Default value is 0.0.*

- `sfh_filename` Filename of the tabulated SFH file located in the `data` directory. If unset, the default `sfh.dat` is assumed. Only applies if `sfh`= 2.

Dust Parameters

- `dust_tesc` Stars younger than `dust_tesc` are attenuated by both `dust1` and `dust2`, while stars older are attenuated by `dust2` only. Units are log(yrs). *Default value is 7.0.*

- `dust1` Dust parameter describing the attenuation of young stellar light, i.e. where $t \leq$`dust_tesc` (for details, see Conroy et al. 2009a). *Default value is 0.0.*

- `dust2` Dust parameter describing the attenuation of old stellar light, i.e. where $t >$`dust_tesc` (for details, see Conroy et al. 2009a). *Default value is 0.0.*

- `dust_clumps` Dust parameter describing the dispersion of a Gaussian PDF density distribution for the old dust. Setting this value to -99.0 sets the distribution to a uniform screen. *Default value is -99.0.* See Conroy et al. 2009b for details.

- `frac_nodust` Fraction of starlight that is not attenuated by the diffuse dust component (i.e. that is not affected by `dust2`). *Default value is 0.0.*

- `frac_obrun` Fraction of starlight that is not attenuated by the birth cloud dust component (i.e. that is not affected by `dust1`). *Default value is 0.0.*

- `dust_index` Power–law index of the diffuse dust attenuation curve. Only used when `dust_type`=0. *Default value is -0.7.*

- `dust1_index` Power–law index of the birth cloud dust attenuation curve. Only used when `dust_type`=0. *Default value is -1.0.*

- `mwr` The ratio of total to selective absorption which characterizes the MW extinction curve: $R \equiv A_V/E(B-V)$. Only used when `dust_type`=1. *Default value is 3.1.*

- `uvb` Parameter characterizing the strength of the 2175Å extinction feature with respect to the standard Cardelli et al. determination for the MW. Only used when `dust_type`=1. *Default value is 1.0.*

- `wgp1` Integer specifying the optical depth in the Witt & Gordon 2000 (WG00) models. Values range from $1-18$, corresponding to optical depths of $0.25, 0.50, 0.75, 1.00, 1.50, 2.00, 2.50, 3.00, 3.50, 4.00, 4.50, 5.00, 5.50, 6.00, 7.00, 8.00, 9.00, 10.0$. Note that these optical depths are defined differently from the optical depths defined by the parameters `dust1` and `dust2`. See WG00 for details.

- `wgp2` Integer specifying the type of large–scale geometry and extinction curve. Values range from $1-6$, corresponding to MW+dusty, MW+shell, MW+cloudy, SMC+dusty, SMC+shell, SMC+cloudy. MW= Milky Way extinction; SMC= Small Magellanic Cloud extinction. Dusty, shell, and cloudy specify the geometry and are described in WG00.

- `wgp3` Integer specifying the local geometry for the WG00 dust models. A value of 0 corresponds to a homogeneous distribution, and a value of 1 corresponds to a clumpy distribution. See WG00 for details.

- `duste_gamma` Parameter of the Draine & Li (2007) dust emission model. Specifies the relative contribution of dust heated at a radiation field strength of $U_{\rm min}$ and dust heated at $U_{\rm min} < U \leq U_{\rm max}$. Allowable range is $0.0-1.0$. *Default value is 0.01.*

- `duste_umin` Parameter of the Draine & Li (2007) dust emission model. Specifies the minimum radiation field strength in units of the MW value. Valid range is between 0.1 and 25.0. *Default value is 1.0.*

- `duste_qpah` Parameter of the Draine & Li (2007) dust emission model. Specifies the grain size distribution through the fraction of grain mass in PAHs. This parameter has units of % and a valid range of $0.0-10.0$. *Default value is 3.5.*

Isochrone Parameters

- `redgb` Modify weight given to the RGB. Only available with BaSTI isochrone set. *Default value is 1.0.*

- `dell` Shift in $\log(L_{\rm bol})$ of the TP-AGB isochrones. Note that the meaning of this parameter and the one below has changed to reflect the updated calibrations presented in Conroy & Gunn 2009. That is, these parameters now refer to a modification about the calibrations presented in that paper. *Default value is 0.0.*

- `delt` Shift in $\log(T_{\rm eff})$ of the TP-AGB isochrones. *Default value is 0.0.*

- `fcstar` Fraction of stars that the Padova isochrones identify as Carbon stars that FSPS assigns to a Carbon star spectrum. Set this to 0.0 if for example the users wishes to turn all Carbon stars into regular M-type stars. Valid range is $0.0-1.0$. *Default value is 1.0.*

- `sbss` Specific frequency of blue straggler stars. See Conroy et al. 2009a for details and a plausible range. *Default value is 0.0.*

- `fbhb` Fraction of horizontal branch stars that are blue. The blue HB stars are uniformly spread in $\log(T_{\rm eff})$ to 10^4 K. See Conroy et al. 2009a for details and a plausible range. *Default value is 0.0.*

- `pagb` Weight given to the post–AGB phase. A value of 0.0 turns off post–AGB stars; a value of 1.0 implies that the Vassiliadis & Wood 1994 tracks are implemented as–is. *Default value is 1.0.*

3.2 Description of Fortran Routines

We now discuss the purpose and syntax of the routines in this package. §3.2.1 discusses the example routines provided in FSPS that demonstrate how many of these routines are used. Routines included in the `src` directory that are *not* discussed below are meant to only be accessed through other routines.

- `COMPSP(write_compsp,nzin,outfile,mass_ssp,lbol_ssp,spec_ssp,pset,ocompsp)`

This routine takes as input `mass_ssp`, `lbol_ssp`, and `spec_ssp`, which are the outputs of the routine `ssp_gen.f90`. The user must also provide the parameter set `pset` and filename for output in `outfile`, if output is desired (a blank string may be specified if no output is desired). There are four possible outputs, specified by `write_compsp`: No output (0), output magnitudes (1), output spectra (2), output magnitudes and spectra (3). The magnitude and spectra outputs are written to files with the output filename with ".mags" and ".spec" appended.

A variety of output from this routine is saved in the `ocompsp` variable, which must be a structure as defined in `sps_vars.f90`. Again, see the example routine below.

The `nzin` parameter sets the number of metallicity points passed. For standard, single metallicity calculations, set this value to one and simply pass the outputs from `ssp_gen.f90`. However, if one wants to compute spectra for an evolving metallicity (i.e. when computing a tabulated SFH, the metallicity history may be specified), one needs to set this parameter to the number of metallicity elements available (in the default release this is 22). One then must take care to pass the metallicity–dependent `ssp_gen.f90` outputs. Currently the code only allows the specification of a metallicity history when passing tabulated SFH.

- `GETMAGS(zred,spec,mags)`

The input is the redshift `zred` and spectrum `spec`. The output is an array of magnitudes `mags` for the redshifted spectrum.

- `GETINDX(lambda,spec,indices)`

This routine computes the spectral indices for the input spectrum `spec` with corresponding wavelength array `lambda`, returning the indices in an array `indices`. The `indices` array must be defined with an array length specified by the variable `nindsps`, which specifies the number of indices. The indices are defined in the file allindices.dat in the `data` directory.

- `PZ_CONVOL(yield,zave,spec_pz,lbol_pz,mass_pz)`

This routine convolves the full array of metallicity-dependent SSPs with a closed-box metallicity distribution function (MDF). The yield is the only input. Outputs include the average metallicity (`zave`) and the spectra, lbol, and mass integrated over the MDF. The full metallicity-dependent SSPs must have been previously set up. An example of how to use this routine is provided in `lesssimple.f90`.

- `SFHSTAT(pos,model,ssfr6,ssfr7,ssfr8,ave_age)`

This routine returns basic statistics for a given star formation history. The inputs are the parameter set (`pos`) and a single element output from the compsp routine (`model`). The outputs are the specific SFR (SSFR), averaged over 10^6, 10^7, and 10^8 yrs (`ssfr6, ssfr7, ssfr8`), and the mass-weighted average stellar age (`ave_age`). An example of how to use this routine is provided in `simple.f90`.

- `SPS_SETUP(zin)`

This routine must be called at least once before running any routines. It reads in all of the isochrones and spectral libraries and stores them in a common block. If the user requires only one metallicity, then that metallicity can be specified as `zin`. The metallicity must be specified as an integer corresponding to the look-up table at the end of this manual. If the user wishes to read in all metallicities, then `zin` should be set to -1.

- `SSP_GEN(pset,mass_ssp,lbol_ssp,spec_ssp)`

This routine takes as input the parameter set, `pset`, and outputs the time-dependent mass, `mass_ssp`, bolometric luminosity, `lbol_ssp`, and spectrum, `spec_ssp`, of an SSP defined by the parameter set. Each of these input variables must be properly defined at the beginning of the main routine. See the example routines for guidance.

- `VELBROAD(lambda,spec,sigma)`

This routine broadens the input spectrum `spec` with corresponding wavelength array `lambda` by a velocity dispersion `sigma` measured in km/s. Note that the velocity broadening is only approximate in that we are ignoring the variation in $d\lambda$ with λ within each integration step.

- `WRITE_ISOCHRONE(outfile,zz)`

This routine writes the isochrones for all ages at a single metallicity to a file with name `outfile`. The metallicity of the isochrone is specified via the variable `zz` in the internal integer metallicity units (see the Table at the end of this manual). At each age the isochrone is written for all of the magnitudes specified in the `FILTER_LIST` file.

3.2.1. Example Routines

The code package contains several routines that highlight many of the features of FSPS. The routine `simple.f90` demonstrates the basic syntax required to generate simple models. In addition, there is a less simple routine located in the `src` directory, called `lesssimple.f90`, that highlights some more advanced features of the code.

4. Description of IDL Routines

- `res = read_indx(file)`

This function takes as input an index file and reads it into a simple IDL structure. The index file must be created by the user (unlike the .spec and .mags files, which are automatically created in the compsp routine). The format is the age followed by all of the indices defined in the file `INDEX_LIST`.

- `res = read_mags(file)`

This function takes as input the magnitude file (*.mags) produced by `compsp.f90`. The output is an IDL structure with elements including the some of the magnitudes listed in `FILTER_LIST` and computed by `compsp.f90`. Note that not all magnitudes computed and listed in the *.mags file are contained in the resulting structure. This routine can be trivially modified to include other/all of the magnitudes computed.

- `res = read_spec(file)`

This function takes as input the spectra file (*.spec) produced by `compsp.f90`. The output is an IDL structure with elements including the time-dependent spectrum computed by `compsp.f90`.

5. How do I....

5.1 add additional filters?

Adding additional filters is straightforward. There are three things the user must do: 1) modify the `nbands` parameter in the `sps_vars.f90` routine; 2) add the filter to the `allfilters.dat` file located in the

Table 1: Lookup table of metallicity values depending on the combination of isochrone set and spectral library

zmet	Z [$\log(Z/Z_\odot)$] Padova+BaSeL	Z [$\log(Z/Z_\odot)$] BaSTI+BaSeL	Z [$\log(Z/Z_\odot)$] Padova+Miles	Z [$\log(Z/Z_\odot)$] BaSTI+Miles
1	0.0002 (-1.98)	0.0003 (-1.80)	0.0008 (-1.37)	0.0006 (-1.50)
2	0.0003 (-1.80)	0.0006 (-1.50)	0.0031 (-0.78)	0.0040 (-0.67)
3	0.0004 (-1.68)	0.0010 (-1.28)	0.0096 (-0.29)	0.0100 (-0.27)
4	0.0005 (-1.58)	0.0020 (-0.98)	0.0190 (0.000)	0.0200 (0.022)
5	0.0006 (-1.50)	0.0040 (-0.68)	0.0300 (0.198)	0.0300 (0.198)
6	0.0008 (-1.38)	0.0080 (-0.38)		
7	0.0010 (-1.28)	0.0100 (-0.28)		
8	0.0012 (-1.20)	0.0200 (+0.02)		
9	0.0016 (-1.07)	0.0300 (+0.20)		
10	0.0020 (-0.98)	0.0400 (+0.32)		
11	0.0025 (-0.89)			
12	0.0031 (-0.79)			
13	0.0039 (-0.69)			
14	0.0049 (-0.59)			
15	0.0061 (-0.49)			
16	0.0077 (-0.39)			
17	0.0096 (-0.30)			
18	0.0120 (-0.20)			
19	0.0150 (-0.10)			
20	0.0190 (+0.00)			
21	0.0240 (+0.10)			
22	0.0300 (+0.20)			

I have assumed here that $Z_\odot = 0.0190$. This information is also available in the ISOCHRONES directory.

`data` directory. Follow the format: there must be a line starting with a # sign, followed by two columns, the first being the wavelength in angstroms, the second being the total throughput. The filter can be of any resolution, and need not be properly normalized. 3) The user would be wise to add details of the filter to the `FILTER_LIST` file located in the `data` directory, although this is not, strictly speaking, necessary for proper functioning of the code.

We would appreciate it if the user would email us if they add a filter so that we can include this filter in later releases (thereby saving others the trouble of adding filters).