The AstFilter class

in the Astronomy & Astrophysics package for MATLAB

Description:

The AstFilter support excess and manipulation of astronomical telescope transmission filters. The class includes a structure-array like object that contains about 200 astronomical transmission filters. The user can add his/her own filters. Furthermore, the filters can be plotted, manipulated and other classes (e.g., AstSpec) can work with this class.

This file is accessible through "AstFilter.manual".

Credit

If you are using this code or products in your scientific publication please give a reference to Ofek (2014; ascl.soft 07005).

License

Instellation

This class is available as part of the *Astronomy and Astrophysics toolbox for matlab*. Furthermore, some of the functions in this class use functions in other packages in the toolbox, so full instellation is recomended.

See http://weizmann.ac.il/home/eofek/matlab/doc/install.html for instellation instruction and additional documentation.

The AstFilter class requires the AstFilterCat.mat file in the /data/Filters/ directory, as well as the @AstFilter directory in a directory contained in the MATLAB search path.

The AstFilter object properties

The AstFilter object contains about 200 elements with the following properties:

- family - Filter family name (e.g., 'SDSS', 'Johnson','2MASS',...)
- band - Band name (e.g., 'u','U','K',...)
- T - A two elemnt matrix of transmission curve [Wavelength(Ang), Transmission].
- Tunits - Transmission units in T. If empty then the units may be arbitrary.
- nT - Normalized transmission matrix such that the integral is 1.
- min_wl - Minimum wavelength
- max_wl - Maximum wavelength
- eff_wl - Effective wavelength
- half_width - Filter half width (kept for historical reasons)
- fwhm - Filter Full Width at Half Maximum
- comments - General comments
- source - Source

- UserData - Additional user data

Methods an examples:

To see all the properties and methods (functions) associated with the `AstFilter` object, first define an AstFilter object:

```
AstF=AstFilter
```

and than type:

```
%AstF.<tab>
```

Loading filters

The transmission filters are stored in a mat file named `AstFilterCat.m`.

To load the entire `AstFilter` object to memory use the static class:

```
F=AstFilter.get;
```

To see the content of F:

```
F
```

You can also upload specific filters.

Loading all the SDSS family filters

```
F=AstFilter.get('SDSS')
```

```
F = 
  5×1 AstFilter array with properties:

    family
    band
    T
    Tunits
    nT
    min_wl
    max_wl
    eff_wl
    half_width
    fwhm
    comments
    source
    UserData
```

or loading all the Filter named 'V'

```
AstFilter.get([],'V');
```

or a load several families

```
F = AstFilter.get({'PTF','sdss'});
```

You can search filters by a substring of family name:

```
F=AstFilter.get;
F=search(F,'sds')
```

or to get all available filter families:

```
F=AstFilter.get;
all_family(F)
```

Plot functions

You can plot the filter transmission curves:

```
F=AstFilter.get('SDSS')
plot(F)
% or
stairs(F)
```

Statistics

You can get some statistical properties of transmission curves using the

`integral`, `max_tran`, `max_wave`, and `norm`.

See help for more details

Shift

You can shift (by redshift) a transmission curve. For example:

```
F=AstFilter.get('WFC3','F160W')
F1=shift(F,0.3);
plot([F;F1])
```

Interpolation and sampling

Any operation of applying a transmission curve to e.g., a spectrum, or multiplication of several transmission curves requires that they will be resampled to the same grid.

The functions `interp`, `equalize_sampling`, `common_sampling` can be used for these purposes.

While equalize_sampling resample an AstFilter object to have the same sampling as another object, common_sampling is searching for the common wavelength range of two AstFilter objects and sample them to the same common wavelength range.

Examples:

```
F=AstFilter.get('SDSS');
Fu=interp(F(1),(3000:1:4000)')
```

Adding filters

To add your own filters, and populate the filters meta data information you can use the

add_filter, pop_wl, save functions.

See help for details.

Example:

```
% construct a new Filter
F=AstFilter;
% set the filter family and name
F.family='Ofek';
F.band  ='Eran';
% populate it with some filter shape (top-hat):
F.T = [4000 0; 4001 1; 4010 1; 4011 0];
% add the new filter F into the filters database in memory
F1  = add_filter(F);
% update the filter database on disk
% save(F1)
```

Operators

To apply an arbitrary operator (unary or binary) to an AstFilter object you can use the filter_fun1 and filter_fun2 functions.

In addition many overload functions like plus, minus, times, rdivide, lt, gt, le, ge, ne, eq are available.

These functions first uses the common_sampling function and than perform the operation.

Examples:

```
F=AstFilter.get('2MASS');
plot(F(1));
hold on;
plot(filter_fun1(F(1),@sqrt))
```

Interacation with other classes and external functions

The AstFilter class is optionally used by other classes and functions.

For example, the `synphot` function to calculate synthetic photometry can work with the `AstFilter` class.