# observation_helper

***Release 0.0.1***

**Kain McCall**

**Jul 09, 2020**

# CONTENTS:

CHAPTER

ONE

# OBSERVATION_HELPER TUTORIAL

Here's a handy guide for your first steps with observation_helper.

## 1.1 Getting Started

In order to use observation_helper, you will need to have the following packages installed: matplotlib numpy scipy gatspy PyAstronomy

The tutorial file also uses os.path to read in some example data from included files. To get started, just open tutorial.py in your preferred editor to read the comments and follow along in the file!

## 1.2 The Basics

The observation_helper module aims to make the organization of an observational astronomy project easier by focusing on two types of objects: files and targets. Files refer to the .txt files full of data about a particular target. A target is a star or other object which was imaged and is now being studied.

## 1.3 tutorial.py

This file is a nice starting point for seeing how observation_helper works. Start by opening and running this file. The comments in the tutorial.py file serve as a tutorial for how to use observation_helper. Further information and detail can be gained from reading the API documentation.

CHAPTER

TWO

# THE OBSERVATION_HELPER API REFERENCE

## 2.1 The "observation_helper" Module

**class** observation_helper.observation_helper.**AIJFile**(*filename*, *filter_type*, *target_star_dict*, *ref_star_dict*, *point_format=None*)

This is a class of objects which represent the output text files from AstroImageJ. Each object must be created with properties encoded to tell the program what filter the data were taken in, which stars belong to which aperture, etc.

**Parameters**

- **filename** (*str*) – name of output text file
- **filter_type** (*str*) – filter in which data were taken
- **target_star_dict** (*dict{str:int}*) – dictionary containing the name of the target star in each target aperture (designated with a 'T' in the label by AstroImageJ) as keys, and the integer aperture number (i.e., 1 for 'T1', 5 for 'T5') as the corresponding value
- **ref_star_dict** (*dict{str:(int, float)}*) – dictionary, similar to above, containing name of each reference star as key, and a tuple as value. Tuple should be of the format (aperture number, magnitude).
- **point_format** (*str*) – matplotlib color/style for plotting anything from this file (e.g., "b." or "k-"). Default is "b."

**clipFile**(*ref_star*, *sigma*, ***kwargs*)

Clips data points from file if the magnitude of ref_star is more that sigma standard deviations away from its mean magnitude. Actually creates a new .txt file in the same directory with the same name, with _clipped appended to the end and uses this new file for all following calculations, etc.

**Parameters**

- **ref_star** (*str*) – name of the reference star whose magnitude to calculate and use to determine which data points should be clipped. MUST match name in ref_star_dict.
- **sigma** (*int or float*) – limit on number of standard deviations away a value can be without being eliminated

Kwargs: :param use_ref_stars: a list of reference star names (see `ref_star_dict`) to use for calculation of magnitude, if use of only some reference stars is desired :type use_ref_stars: list

**getColumn**(*label*)

Returns all values from the column in the AstroImageJ file with the label label.

**Parameters** **label** (*str*) – label of the column whose values are returned

**Returns** column in file with label `label`

**Return type** ndarray

**plotFile**(*xLabel*, *yLabel*, *showPlot=None*)

Creates a plot of column `yLabel` vs. column `xLabel`, where `xLabel` and `yLabel` are the labels of each column in the AstroImageJ output file.

**Parameters**

- **xLabel** (*str*) – label of column in AstroImageJ output file whose data will be considered the x-values of plot
- **yLabel** (*str*) – label of column in AstroImageJ output file whose data will be considered the y-values of plot
- **showPlot** (*bool*) – determines whether to print plot to console (default is True)

**Returns** column in file with label `xLabel`, column in file with label `yLabel`

**Return type** ndarray, ndarray

**unclipFile**()

Un-clips the data file. In reality, goes back to using old, untouched file for calculations, etc.

**class** observation_helper.observation_helper.**OtherDataFile**(*filename*, *filter_type*, *column_positions*, *point_format=None*, *has_header=None*)

Class of objects which represent data files not from AstroImageJ. These could be, for instance, .txt files of data from ASAS-SN or CSS, etc. Text files must contain at least columns for HJD, magnitude, and error in magnitude.

**Parameters**

- **filename** (*str*) – name of the file which contains the data
- **filter_type** (*str*) – filter in which data were taken
- **column_positions** – list of zero-indexed columns corresponding to HJD, magnitude, and error in magnitude, in that order. (e.g. [0, 2, 3] would mean that HJD data is in first column, magnitude is in the third column, and error in magnitude is in 4th column)
- **point_format** (*str*) – matplotlib color/style for plotting this file (e.g., "b." or "k-"). Default is "b."
- **has_header** (*int*) – number of rows to ignore in beginning of file (in case of headers, column labels, etc.). Default is 0

**class** observation_helper.observation_helper.**Target**(*name*, *coords*, *given_period*, *phase0_hjd=None*)

Class of objects which represent target stars. Each has properties like a name, coordinates, a period, etc.

**Parameters**

- **name** (*str*) – name of the target
- **coords** (*tuple (float, float)*) – tuple containing the J2000 RA and Dec of the target (RA, Dec), in decimal format (NOT sexagesimal)
- **given_period** (*float*) – period of target to use as default, initially (can be changed– see *calcPeriod()*, *addPeriod()*, and *setPeriod()*)
- **phase0_hjd** (*float*) – when phasing, this will be the HJD with the initial phase, or phase=0 (default is 0)

**addData** (*file*, *comparison_star*)
Adds an AIJFile object or an OtherDataFile object to the target object.

**Parameters**

- **file** (*AIJFile or OtherDataFile*) – File to be added to the target
- **comparison_star** (*str*) – name of the comparison star to use to calculate characteristic error

**addPeriod** (*label*, *period*)
Adds a period to the list of periods for Target (does NOT set this period as the default period. See *setPeriod()*)

**Parameters**

- **label** (*str*) – label of period to be added
- **period** (*float*) – value of period to be added

**calcPeriod** (*filter_type*, *period_range=None*, *graphP=None*, *printP=None*, ***kwargs*)
Calculates a period for the target based upon data added in filter_type.

**Parameters**

- **filter_type** (*str*) – filter in which data to be used for calculation of period was taken
- **period_range** (*tuple*) – renge, in days, in which to search for periods
- **graphP** (*bool*) – determines whether or not to print periodogram to the console. Default is False.
- **printP** (*bool*) – determines whether or not to print the value of the best period to the console. Default is False.

**Returns** calcP – calculated period

**Return type** float

**getColorFit** (*filter1*, *filter2*, *forceN1=None*, *forceN2=None*, ***kwargs*)
Fits a Fourier series each to data in filter1 and filter2, then uses these fits to find color index (filter1 - filter2) over phase for a Target.

**Parameters**

- **filter1** (*str*) – filter in which data to be fitted was taken (e.g. V in V - R)
- **filter2** (*str*) – filter in which data to be fitted was taken (e.g. R in V - R)
- **forceN1** (*int*) – highest N out to which to calculate fit for filter 1 (see definition of Fourier series). By default, uses unit-lag auto-correlation to determine best N for fit, but this parameter overrides the process
- **forceN2** (*int*) – highest N out to which to calculate fit for filter 2 (see definition of Fourier series). By default, uses unit-lag auto-correlation to determine best N for fit, but this parameter overrides the process

**Returns** xPhase, color– lists of the phases and color index at each of the phases, respectively

**Return type** list, list

**getColorTemperature** (*filter1*, *filter2*, *tempModelFit*, *forceN1=None*, *forceN2=None*, ***kwargs*)
Uses *getColorFit()* and a quadratic model relating temperature to color index to calculate the color temperature of the Target.

**Parameters**

- **filter1** (*str*) – filter in which data to be fitted was taken (e.g. V in V - R)
- **filter2** (*str*) – filter in which data to be fitted was taken (e.g. R in V - R)
- **tempModelFit** (*tuple of length 3*) – (a, b, c) where color temperature = a + b(filter1 - filter2) + c(filter1 - filter2)^2
- **forceN1** (*int*) – highest N out to which to calculate fit for filter 1 (see definition of Fourier series). By default, uses unit-lag auto-correlation to determine best N for fit, but this parameter overrides the process
- **forceN2** (*int*) – highest N out to which to calculate fit for filter 2 (see definition of Fourier series). By default, uses unit-lag auto-correlation to determine best N for fit, but this parameter overrides the process

**Returns** xPhase, temp – lists of the phases and the calculated temperature at each of those phases, respectively

**Return type** list, list

**getFluxData**(*filter_type*, *return_1D_lists=None*, ***kwargs*)

Calculates the relative flux of the target star

**Parameters**

- **filter_type** (*str*) – Filter in which data to be returned was taken
- **return_1D_lists** (*bool*) – determines whether to return lists of equal length with the data from all files combined into the same list, or whether to return lists of lists, where each returned list contains lists of values which correspond to a particular file (default is False)

**Returns** HJD, flux, dFlux, pltColor

**Return type** list, list, list, list

**getFourierFit**(*filter_type*, *lineFmt=None*, *showPlot=None*, *forceN=None*, ***kwargs*)

Fits a Fourier series to data added to target in filter_type

**Parameters**

- **filter_type** (*str*) – filter in which data to be fitted was taken
- **lineFmt** (*str*) – matplotlib parameter for style and color of line indicating Fourier Fit. Default is "k-"
- **showPlot** (*bool*) – determines whether or not to show the graph of the resulting fit.
- **forceN** (*int*) – highest N out to which to calculate fit (see definition of Fourier series). By default, uses unit-lag auto-correlation to determine best N for fit, but this parameter overries that process.

**Returns** popt, pcov (fitted parameters and their covariance– see curve_fit in SciPy package)

**Return type** array, 2d array

**getMagData**(*filter_type*, *return_1D_lists=None*, ***kwargs*)

Calculates magnitude of Target object for all data points added and returns HJD, magnitude, error in magnitude, and plot color

**Parameters**

- **filter_type** (*str*) – Filter in which data to be returned was taken

- **return_1D_lists** (*bool*) – determines whether to return lists of equal length with the data from all files combined into the same list, or whether to return lists of lists, where each returned list contains lists of values which correspond to a particular file (default is False)

**Returns** HJD, mag, dMag, pltColor

**Return type** list, list, list, list

**makeLightCurve**(*filter_type*, *plotHJD=None*, *plotFlux=None*, *runShowPlot=None*, *bar_position=None*, *set_title=None*, ***kwargs*)

Creates a light curve from target data added in filter_type

**Parameters**

- **filter_type** (*str*) – filter in which data to be plotted was taken
- **plotHJD** (*bool*) – indicates whether x-axis should be HJD or phase. Default is phase (False)
- **plotFlux** (*bool*) – indicates whether y-axis should be magnitude or flux. Default is magnitude (False)
- **runShowPlot** (*bool*) – indicates whether to run show() command at end of sequence– if False, can add/edit plot (i.e., plot additional items on light curve, like a fit line, etc.). Default is True
- **bar_position** (*float*) – determines the y-axis position of the characteristic error bars. Calculates automatically by default, but to set manually, this argument can be used.
- **set_title** (*str*) – title to be used for light curve. Default is the target name.

**plotFiles**(*xLabel*, *yLabel*, ***kwargs*)

Plots column with label `xLabel` vs. column with label `yLabel` for files added to Target.

**Parameters**

- **xLabel** (*str*) – label of column in AstroImageJ output files whose data will be considered the x-values of plot
- **yLabel** (*str*) – label of column in AstroImageJ output files whose data will be considered the y-values of plot

**setPeriod**(*period_label*)

Sets the default period to the period with label `period_label`.

**Parameters** **period_label** (*str*) – label of period to be set as default

**timeOnTarget**(*time_threshold*)

Prints a summary of time spent observing Target based upon files added.

[note that this WILL NOT work for things like ASAS-SN, since images are taken every few days, unless threshold is set to be larger than the time difference between images being taken]

**Parameters** **time_threshold** (*float*) – value (in days) of how long of a break between images being taken to not include it in the calculation of time on target.

## 2.2 kwargs

**use_files** (list containing AIJFile and/or OtherDataFile objects): list containing which files to use for calculations, plots, etc., if not all files should be included.

**use_ref_stars** (list): a list of reference star names (see `ref_star_dict`) to use for calculation of magnitude, if use of only some reference stars is desired

# PYTHON MODULE INDEX

## O

# INDEX