SPCA Documentation

Release 0.3

Lisa Dang and Taylor James Bell

Oct 13, 2020

API Table of Contents:

1 Installation Instructions **3**

2 Update Log **5**
- 2.1 Version 0.3 5
- 2.2 Version 0.2 5

3 Package Usage **7**
- 3.1 SPCA package 8

4 Indices and tables **39**

5 Contributions **41**

6 Acknowledgements **43**

7 License & Attribution **45**

Python Module Index **47**

Index **49**

SPCA is an open-source, modular, and automated pipeline for Spitzer Phase Curve Analyses.

CHAPTER 1

Installation Instructions

To install SPCA, run the following in a terminal:

```
git clone https://github.com/lisadang27/SPCA.git
cd SPCA
pip install .
# IF you want to install george to run GP analyses, also then run
pip install .[GP]
```

Please note however that SPCA is in a state of alpha testing and is still under development. Frequent changes are expected over the upcoming few months as we finalize some aspects like PSF fitting with aperture photometry or nearby companion removal using PSF subtraction.

CHAPTER 2

Update Log

2.1 Version 0.3

As of version 0.3, we have made some important changes. These include:

- Added the ability to perform full-frame photometry
- Added the ability to fit unbinned photometry
- Added the ability to decorrelate using PSF-fitted centroids using the "_PSFX" mode suffix
- Added sigma-clipping along full time axis (not just within data cube). This comes at the cost of needing to load all frames into RAM at the same time which can be quite RAM intensive for some phase curves. Will consider allowing user to switch to using dask in the future to reduce RAM cost when performing photometry at the cost of slower run speeds.
- Changed BLISS knot density selection algorithm to be more like POET's algorithm
- Added a progress bar for when running photometry
- No updates on emcee freezing, but it now rarely ever occurs for me (Taylor). Perhaps the better initialization routine is helping here.

2.2 Version 0.2

As of version 0.2, we have made some important changes and some bug fixes. These include:

- Renamed decorrelation and fitting file to Decorrelation (.py and .ipynb)
- Full integration of PLD and PLDAper models
- Reduced the amount of static code that the user sees and placed this in separate files instead.
- Changed how photometry is saved. This will not be noticeable unless you want to use PLDAper models which cannot be run with the old photometry.

- Also changed the units in which the photometry is saved so that it is easy to compute the photon noise limit - a calculation that was previously done incorrectly.
- The emcee fitting routine sometimes freezing still seems to be an issue as of v0.2. This seems to be caused by the combination of a large dataset and either a large model or a poorly initialized model. Previous attempts at forcing a timeout have failed without directly editing emcee or multiprocessing code. If this occurs to you and you need to analyze your data soon, I'd recommend removing the few lines that use multiprocessing - contact us if you have a hard time doing this. We are looking into different samplers (such as PyMC3) to resolve this issue.

CHAPTER 3

Package Usage

0. Follow the installation instructions at the top of this page to ensure you have all of the necessary dependencies.
1. To use SPCA, you must first download your Spitzer data from the Spitzer Heritage Archive: https://sha.ipac.caltech.edu/applications/Spitzer/SHA/. Place the downloaded zip files in a directory with the same name as the planet (exluding spaces).

Most of the following commands have an .ipynb ending and .py ending option available, where the .py version is optimized for analyzing many data sets and the .ipynb file is optimized for viewing the analysis of a single data set. Each file has a portion at the top where you can set parameters which will determine the techniques.

2. Next, use the Make_Directory_Structure file to extract the data and setup the required directories.
3. Then use the Everything_Photometry file to perform a suite of different photometries on your data. The code will automatically select the best aperture photometry method which gives the lowest scatter after smoothing the raw flux by a boxcar filter of a width provided by the user.
4. Then use the QuickLook file to ensure that you have looked at the raw data (to ensure it looks reasonable) and to determine whether you want to remove the first AOR (in case it is a short AOR to be discarded). By looking at the raw data, you can gain some insight into how successful different decorrelation models might be.
5. Then decorrelate the data using the Decorrelation file. Most parameters here are explained with a nearby comment. One key parameter though is the "mode" which sets the decorrelation method used. Modes that contain "Poly#" use a 2-dimentionsal polynomial of order #, modes that contain "BLISS" use BiLinearly-Interpolated Subpixel Sensitivity mapping, modes that contain "GP" use a Gaussian Process using x and y centroid positions as covariates, and modes that contain "PLD#_x" use Pixel Level Decorrelation of order # (1 or 2) and use a pixel stamp size of $ by $ (3x3 or 5x5). PLD performed using aperture photometry (the recommended PLD technique) can be accessed using "PLDAper#_x". Each mode is then followed by an underscore and either "v1" or "v2" indicating the use of either a 1st or 2nd order sinusoidal model for the phase variations. If "PSFX" is present in the mode string for non-PLD models, the data are decorrelated using the PSF-fitted centroids rather than the flux-weighted mean centroids. If "ellipse" is present in the mode string, phase variations due to the elliptical shape of the planet are modelled. Any other text can be added to the mode keyword for your own convenience (e.g. "Poly2_v1_run2" or "PLDAper2_5x5_v1_testingFit").
6. Finally, some tables containing a selection of the fitted parameters from each model run can be made using the MakeTables file. These tables will also highlight the best decorrelation method for each analysis, determined

using delta-BIC (selecting your model is actually quite a challenging and complicated step, and user discretion is absolutely recommended).

3.1 SPCA package

3.1.1 Submodules

3.1.2 SPCA.Decorrelation_helper module

`SPCA.Decorrelation_helper.burnIn`(*p0*, *p0_labels*, *mode*, *astro_func*, *astro_labels*, *astro_inputs*, *signal_func*, *signal_inputs*, *lnprob_inputs*, *gparams*, *gpriorInds*, *priors*, *errs*, *time*, *flux*, *breaks*, *bestfitNbin*, *ncpu*, *savepath=None*, *showPlot=False*, *nIterScipy=10*)

`SPCA.Decorrelation_helper.downloadExoplanetArchive`()

Downloads the most recent masterfile of the best data on each target :param None:

Returns None

`SPCA.Decorrelation_helper.findPhotometry`(*rootpath*, *planet*, *channel*, *mode*, *pldIgnoreFrames=True*, *pldAddStack=False*)

Calculate the stellar brightness temperature based on phoenix spectrum. :param rootpath: Path to data :type rootpath: str :param planet: name of the planet :type planet: str :param channel: Spitzer channel used to obtain the observation (ch1 or ch2) :type channel: str :param mode: decorrelation mode chosen :type mode: str :param pldIgnoreFrames: True if you want PLD to also ignore the first AOR. Default is True. :type pldIgnoreFrames: bool, optional :param pldAddStack: True is you want to apply (and already have) custom Spitzer correction to your data when using PLD. Default is False. :type pldAddStack: bool, optional

Returns path to folder containing filename filename_full savepath path_params AOR_snip aors breaks ignoreFrames p0_obj['Tstar_err'] (float): uncertainty on stellar temperature

Return type foldername (str)

`SPCA.Decorrelation_helper.getTstarBright`(*rootpath*, *planet*, *channel*, *p0_obj*)

Calculate the stellar brightness temperature based on phoenix spectrum. :param rootpath: Path to data :type rootpath: str :param planet: name of the planet :type planet: str :param channel: Spitzer channel used to obtain the observation (ch1 or ch2) :type channel: str :param p0_obj: object with initial parameters :type p0_obj: obj

Returns

??? p0_obj['Tstar_err'] (float): uncertainty on stellar temperature

Return type tstar_b ()

`SPCA.Decorrelation_helper.get_photon_limit`(*path*, *mode*, *nFrames*, *ignoreFrames*)

`SPCA.Decorrelation_helper.get_photon_limit_oldData`(*rootpath*, *datapath*, *datapath_aper*, *planet*, *channel*, *mode*, *aors*, *nFrames*, *ignoreFrames*)

`SPCA.Decorrelation_helper.loadArchivalData`(*rootpath*, *planet*, *channel*)

Loads system parameters and respective uncertainties from the NASA Exoplanets Archives into p0_obj. :param rootpath: Path to data :type rootpath: str :param planet: name of the planet :type planet: str :param channel: Spitzer channel used to obtain the observation (ch1 or ch2) :type channel: str

Returns object with initial parameters

Return type p0_obj (obj)

SPCA.Decorrelation_helper.**loadCustomData**(*rootpath*, *planet*, *channel*, *rp*, *a*, *per*, *t0*, *inc*, *e*, *argp*, *Tstar*, *logg*, *feh*, *rp_err=inf*, *a_err=inf*, *t0_err=inf*, *per_err=inf*, *inc_err=inf*, *e_err=inf*, *argp_err=inf*, *Tstar_err=inf*)

Load custom parameters into p0_obj. :param rootpath: Path to data :type rootpath: str :param planet: name of the planet :type planet: str :param channel: Spitzer channel used to obtain the observation (ch1 or ch2) :type channel: str :param rp: radius of the planets in unit of stellar radius :type rp: float :param a: semi-major axis of the orbit of the planet in unit of stellar radius :type a: float :param per: period of the orbit in days :type per: float :param t0: time of transit in BMJD :type t0: float :param inc: inclination of the orbit in degrees (inc = 90 for edge-on orbit) :type inc: float :param e: eccentricity of the orbit :type e: float :param argp: argument of periastron in degrees :type argp: float :param Tstar: Stellar temperature in Kelvins :type Tstar: float :param logg: Surface gravity in log_10(cm/s^2) :type logg: float :param feh: Stellar metallicity ratio [Fe/H] :type feh: float :param rp_err: uncertainty on radius of the planets in unit of stellar radius :type rp_err: float,optional :param a_err: uncertainty on semi-major axis of the orbit of the planet in unit of stellar radius :type a_err: float,optional :param per_err: uncertainty on period of the orbit in days :type per_err: float,optional :param t0_err: uncertainty on time of transit in BMJD :type t0_err: float,optional :param inc_err: uncertainty on inclination of the orbit in degrees (inc = 90 for edge-on orbit) :type inc_err: float,optional :param e_err: uncertainty on eccentricity of the orbit :type e_err: float,optional :param argp_err: uncertainty on argument of periastron in degrees :type argp_err: float,optional :param Tstar_err: uncertainty on Stellar temperature in Kelvins :type Tstar_err: float,optional

Returns object with initial parameters

Return type p0_obj (obj)

SPCA.Decorrelation_helper.**print_MCMC_results**(*flux*, *flux_full*, *chain*, *lnprobchain*, *mode*, *channel*, *signal_func*, *signal_inputs*, *signal_inputs_full*, *p0_labels*, *p0_obj*, *astro_func*, *astro_inputs*, *astro_inputs_full*, *astro_labels*, *usebestfit*, *savepath*, *sigF_photon_ppm*, *nFrames*, *secondOrderOffset*, *compFactor=1*)

SPCA.Decorrelation_helper.**reload_old_fit**(*path_params*, *p0_obj*, *dparams*)

SPCA.Decorrelation_helper.**setup_gpriors**(*gparams*, *p0_obj*)

3.1.3 SPCA.Photometry_Aperture module

SPCA.Photometry_Aperture.**A_photometry**(*bg_err, cx=15, cx_med=15, cy=15, cy_med=15, r=[2.5], a=[5], b=[5], w_r=[5], h_r=[5], theta=[0], scale=1, shape='Circular', methods=['center', 'exact'], moveCentroids=[True], i=0, img_data=None*)

Performs aperture photometry, first by creating the aperture then summing the flux within the aperture.

Note that this will implicitly use the global variable image_stack (3D array) to allow for parallel computing with large (many GB) datasets.

Parameters

- **bg_err** (*1D array*) – Array of uncertainties on pixel value.
- **cx** (*int/array, optional*) – x-coordinate(s) of the center of the aperture. Default is 15.
- **cy** (*int/array, optional*) – y-coordinate(s) of the center of the aperture. Default is 15.
- **r** (*int/array, optional*) – If shape is 'Circular', the radii to try for aperture photometry in units of pixels. Default is just 2.5 pixels.

- **a** (*int/array, optional*) – If shape is 'Elliptical', the semi-major axes to try for elliptical aperture in units of pixels. Default is 5.
- **b** (*int/array, optional*) – If shape is 'Elliptical', the semi-minor axes to try for elliptical aperture in units of pixels. Default is 5.
- **w_r** (*int/array, optional*) – If shape is 'Rectangular', the full widths to try for rectangular aperture (x-axis). Default is 5.
- **h_r** (*int/array, optional*) – If shape is 'Rectangular', the full heights to try for rectangular aperture (y-axis). Default is 5.
- **theta** (*int/array, optional*) – If shape is 'Elliptical' or 'Rectangular', the rotation angles in radians of the semimajor axis from the positive x axis. The rotation angle increases counterclockwise. Default is 0.
- **scale** (*int, optional*) – If the image is oversampled, scaling factor for centroid and bounds, i.e, give centroid in terms of the pixel value of the initial image.
- **shape** (*string, optional*) – shape is the shape of the aperture. Possible aperture shapes are 'Circular', 'Elliptical', 'Rectangular'. Default is 'Circular'.
- **methods** (*iterable, optional*) – The methods used to determine the overlap of the aperture on the pixel grid. Possible methods are 'exact', 'subpixel', 'center'. Default is ['center', 'exact'].
- **i** (*int, optional*) – The current frame number being examined.
- **img_data** (*3D array*) – The image stack being analyzed if not using the global variable to allow for parallel computing.

Returns

results (2D array) Array of flux and flux errors, of shape (nMethods*nSizes, 2), where the nSizes loop is nested inside the nMethods loop which is itself nested inside the moveCentoids loop.

Return type tuple

class SPCA.Photometry_Aperture.**TestAperturehotometryMethods**(*methodName='runTest'*)

Bases: unittest.case.TestCase

test_FWM_centroiding()

test_circularAperture()

SPCA.Photometry_Aperture.**bin_all_data**(*highpassWidth, flux, binned_time, binned_time_std, binned_xo, binned_xo_std, binned_yo, binned_yo_std, binned_xw, binned_xw_std, binned_yw, binned_yw_std, binned_bg, binned_bg_std, binned_npp, binned_npp_std, bin_size*)

SPCA.Photometry_Aperture.**centroid_FWM**(*image_stack, highpassWidth=320, scale=1, bounds=(13, 18, 13, 18), defaultCentroid=['median', 'median'], defaultPSFW=['median', 'median']*)

Gets the centroid of the target by flux weighted mean and the PSF width of the target.

Parameters

- **image_stack** (*ndarray*) – Data cube of images (2D arrays of pixel values).
- **scale** (*int, optional*) – If the image is oversampled, scaling factor for centroid and bounds, i.e, give centroid in terms of the pixel value of the initial image.

- **bounds** (*tuple, optional*) – Bounds of box around the target to exclude background . Default is (14, 18, 14, 18).
- **defaultCentroid** (*list, optional*) – Default location for sigma clipped centroids. Default is median centroid position.
- **defaultPSFW** (*list, optional*) – Default width for sigma clipped PSF widths. Default is median of widths.

Returns

xo, yo, wx, wy (list, list, list, list). The updated lists of x-centroid, y-centroid, PSF width (x-axis), and PSF width (y-axis).

Return type tuple

SPCA.Photometry_Aperture.**compare_RMS**(*Run_list, fluxes, r, time, highpassWidth, basepath, planet, channel, ignoreFrames, addStack, save=True, onlyBest=False, showPlots=False, savePlots=True*)

SPCA.Photometry_Aperture.**func**(*arg*)

SPCA.Photometry_Aperture.**get_lightcurve**(*basepath, AOR_snip, channel, planet, save=True, onlyBest=True, highpassWidth=320, bin_data=True, bin_size=64, showPlots=False, savePlots=True, oversamp=False, scale=2, saveoversamp=True, reuse_oversamp=True, r=[2.4], edges=['Exact'], addStack=False, ignoreFrames=None, maskStars=None, moveCentroids=[True], ncpu=4, image_stack_input=None, bg=None, bg_err=None, time=None*)

Given a directory, looks for data (bcd.fits files), opens them and performs photometry.

Parameters

- **datapath** (*string*) – Directory where the spitzer data is stored.
- **savepath** (*string*) – Directory the outputs will be saved.
- **AORsnip** (*string*) – Common first characters of data directory eg. 'r579'
- **channel** (*string*) – Channel used for the observation eg. 'ch1' for channel 1
- **shape** (*string, optional*) – The aperture shape to try. Possible aperture shapes are 'Circular', 'Elliptical', 'Rectangular'. Default is 'Circular'.
- **edges** (*iterable, optional*) – The aperture edges to try. Options are 'hard', 'soft', and 'exact' which correspond to the 'center', 'subpixel', and 'exact' methods in astropy. Default is just ['hard'].
- **save** (*bool, optional*) – True if you want to save the outputs. Default is True.
- **save_full** (*string, optional*) – Filename of the full unbinned output data. Default is '/ch2_datacube_full_AORs579.dat'.
- **bin_data** (*bool, optional*) – True you want to get binned data. Default is True.
- **bin_size** (*int, optional*) – If bin_data is True, the size of the bins. Default is 64.
- **save_bin** (*string, optional*) – Filename of the full binned output data. Default is '/ch2_datacube_binned_AORs579.dat'.
- **plot** (*bool, optional*) – True if you want to plot the time resolved lightcurve. Default is True.

- **plot_name** (*string, optional*) – If plot and save is True, the filename of the plot to be saved as. Default is True.
- **oversamp** (*bool, optional*) – True if you want to oversample the image by a factor of 2. Default is False.
- **save_oversamp** (*bool, optional*) – True if you want to save oversampled images. Default is True.
- **reuse_oversamp** (*bool, optional*) – True if you want to reuse oversampled images that were previously saved. Default is False.
- **planet** (*string, optional*) – The name of the planet. Default is CoRoT-2b.
- **rs** (*iterable, optional*) – The radii to try for aperture photometry in units of pixels. Default is just 2.5 pixels.
- **ignoreFrames** (*list, optional*) – frame to remove first-frame systematic).
- **maskStars** (*list, optional*) – An array-like object where each element is an array-like object with the RA and DEC coordinates of a nearby star which should be masked out when computing background subtraction.
- **moveCentroids** (*iterable, optional*) – True if you want the centroid to be centered on the flux-weighted mean centroids (will default to median centroid when a NaN is returned), otherwise aperture will be centered on 15,15 (or 30,30 for 2x oversampled images). Default is [True].
- **rerun_photometry** (*bool, optional*) – Whether to overwrite old photometry if it exists. Default is False.
- **ncpu** (*int, optional*) – The number of aperture radii to try at the same time with multiprocessing. Default is 4.

Raises `Error` – If Photometry method is not supported/recognized by this pipeline.

`SPCA.Photometry_Aperture.`**noisepixparam**(*image_stack*, *bounds=(13, 18, 13, 18)*)

Compute the noise pixel parameter.

Parameters

- **image_stack** (*ndarray*) – FITS images stack.
- **npp** (*list, optional*) – Previously computed noise pixel parameters for other frames that will be appended to.

Returns The noise pixel parameter for each image in the stack.

Return type list

`SPCA.Photometry_Aperture.`**update**(*outputs*)

`SPCA.Photometry_Aperture.`**wrapMyFunc**(*arg*)

3.1.4 SPCA.Photometry_Common module

`SPCA.Photometry_Common.`**bgsubtract**(*bounds=(11, 19, 11, 19)*, *i=0*)

Measure the background level and subtracts the background from each frame.

Parameters **bounds** (*tuple, optional*) – Bounds of box around the target. Default is (11, 19, 11, 19).

Returns

bgsub_data (3D array) Data cube of background subtracted images. bg_flux (1D array) Updated array of background flux measurements for previous images. bg_err (1D array) Updated array of uncertainties on background measurements for previous images.

Return type tuple

`SPCA.Photometry_Common.`**`bin_array`**(*data*, *size*)

Median bin an array.

Parameters

- **`data`** (*`1D array`*) – Array of data to be binned.
- **`size`** (*`int`*) – Number of data points in each bin.

Returns

binned_data (1D array) Array of binned data. binned_data_std (1D array) Array of standard deviation for each entry in binned_data.

Return type tuple

`SPCA.Photometry_Common.`**`clip_data`**(*arr*, *highpassWidth*, *sigma1=5*, *sigma2=5*, *maxiters=3*)

`SPCA.Photometry_Common.`**`create_folder`**(*fullname*, *auto=False*, *overwrite=False*)

Create a folder unless it exists.

Parameters

- **`fullname`** (*`string`*) – Full path to the folder to be created.
- **`auto`** (*`bool, optional`*) – If the folder already exists, should the folder just be skipped (True) or should the user be asked whether they want to overwrite the folder or change the folder name (False, Default).
- **`overwrite`** (*`bool, optional`*) – Whether you want to overwrite the folder if it already exists

Returns The final name used for the folder.

Return type string

`SPCA.Photometry_Common.`**`get_fnames`**(*directory*, *AOR_snip*)

Find paths to all the fits files.

Parameters

- **`directory`** (*`string`*) – Path to the directory containing all the Spitzer data.
- **`AOR_snip`** (*`string`*) – Common first characters of data directory eg. 'r579'.
- **`ch`** (*`string`*) – Channel used for the observation eg. 'ch1' for channel 1.

Returns

fname, lens (list, list). List of paths to all bcd.fits files, number of files for each AOR (needed for adding correction stacks).

Return type tuple

`SPCA.Photometry_Common.`**`get_stacks`**(*calDir*, *dataDir*, *AOR_snip*)

Find paths to all the background subtraction correction stacks FITS files.

Parameters

- **`calDir`** (*`string`*) – Path to the directory containing the correction stacks.
- **`dataDir`** (*`string`*) – Path to the directory containing the Spitzer data to be corrected.

- **AOR_snip** (*string*) – Common first characters of data directory eg. 'r579'.
- **ch** (*string*) – Channel used for the observation eg. 'ch1' for channel 1.

Returns List of paths to the relevant correction stacks

Return type list

SPCA.Photometry_Common.**get_time**(*header*, *ignoreFrames*)

Gets the time stamp for each image.

Parameters

- **header** (*astropy.io.fits.header.Header*) – Header of fits file.
- **ignoreFrames** (*ndarray*) – Array of frames to ignore (consistently bad frames).

Returns Updated time stamp array.

Return type ndarray

SPCA.Photometry_Common.**highpassflist**(*signal*, *highpassWidth*)

SPCA.Photometry_Common.**oversampling**(*image_data*, *scale=2*)

First, substitutes all invalid/sigma-clipped pixels by interpolating the value, then linearly oversamples the image.

Parameters

- **image_data** (*ndarray*) – Data cube of images (2D arrays of pixel values).
- **scale** (*int, optional*) – Sampling factor, e.g. if scale = 2, there will be twice as many data points in the x and y axis. Default is 2. (Do not recommend larger than 2)

Returns Data cube of oversampled images (2D arrays of pixel values).

Return type ndarray

SPCA.Photometry_Common.**prepare_image**(*savepath*, *AOR_snip*, *fnames*, *lens*, *stacks=[]*, *ignoreFrames=[]*, *oversamp=False*, *scale=2*, *reuse_oversamp=True*, *saveoversamp=True*, *addStack=False*, *stackPath=''*, *maskStars=[]*, *i=0*)

SPCA.Photometry_Common.**prepare_images**(*basepath*, *planet*, *channel*, *AOR_snip*, *ignoreFrames=[]*, *oversamp=False*, *scale=2*, *reuse_oversamp=True*, *saveoversamp=True*, *addStack=False*, *maskStars=[]*, *ncpu=4*)

SPCA.Photometry_Common.**replace_clipped**(*arr*)

SPCA.Photometry_Common.**rolling_clip**(*arr*, *highpassWidth*, *sigma=5*, *maxiters=3*)

SPCA.Photometry_Common.**sigma_clipping**(*image_stack*, *bounds=(13, 18, 13, 18)*, *sigma=5*, *maxiters=3*)

Sigma clips bad pixels and mask entire frame if the sigma clipped pixel is too close to the target.

Parameters

- **image_stack** (*3D array*) – Data cube of images (2D arrays of pixel values).
- **bounds** (*tuple, optional*) – Bounds of box around the target. Default is (13, 18, 13, 18).
- **sigma** (*float, optional*) – How many sigma should something differ by to be clipped. Default is 5 which shouldn't trim any real data for Ndata=64*1000.
- **maxiters** (*int, optional*) – How many iterations of sigma clipping should be done.

Returns sigma_clipped_data - Data cube of sigma clipped images (2D arrays of pixel values).

Return type 3D array

3.1.5 SPCA.Photometry_Companion module

`SPCA.Photometry_Companion.`**`A_photometry`**(*image_data*, *bg_err*, *ape_sum=[]*, *ape_sum_err=[]*, *cx=15*, *cy=15*, *r=2.5*, *a=5*, *b=5*, *w_r=5*, *h_r=5*, *theta=0*, *shape='Circular'*, *method='center'*)

Performs aperture photometry, first by creating the aperture (Circular, Rectangular or Elliptical), then it sums up the flux that falls into the aperture.

Parameters

- **image_data** (*3D array*) – Data cube of images (2D arrays of pixel values).
- **bg_err** (*1D array*) – Array of uncertainties on pixel value.
- **ape_sum** (*1D array (optional)*) – Array of flux to append new flux values to. If 'None', the new values will be appended to an empty array
- **ape_sum_err** (*1D array (optional)*) – Array of flux uncertainty to append new flux uncertainty values to. If 'None', the new values will be appended to an empty array.
- **cx** (*int (optional)*) – x-coordinate of the center of the aperture. Default is 15.
- **cy** (*int (optional)*) – y-coordinate of the center of the aperture. Default is 15.
- **r** (*int (optional)*) – If phot_meth is 'Aperture' and ap_shape is 'Circular', c_radius is the radius for the circular aperture. Default is 2.5.
- **a** (*int (optional)*) – If phot_meth is 'Aperture' and ap_shape is 'Elliptical', e_semix is the semi-major axis for elliptical aperture (x-axis). Default is 5.
- **b** (*int (optional)*) – If phot_meth is 'Aperture' and ap_shape is 'Elliptical', e_semiy is the semi-major axis for elliptical aperture (y-axis). Default is 5.
- **w_r** (*int (optional)*) – If phot_meth is 'Aperture' and ap_shape is 'Rectangular', r_widthx is the full width for rectangular aperture (x-axis). Default is 5.
- **h_r** (*int (optional)*) – If phot_meth is 'Aperture' and ap_shape is 'Rectangular', r_widthy is the full height for rectangular aperture (y-axis). Default is 5.
- **theta** (*int (optional)*) – If phot_meth is 'Aperture' and ap_shape is 'Elliptical' or 'Rectangular', theta is the angle of the rotation angle in radians of the semimajor axis from the positive x axis. The rotation angle increases counterclockwise. Default is 0.
- **shape** (*string object (optional)*) – If phot_meth is 'Aperture', ap_shape is the shape of the aperture. Possible aperture shapes are 'Circular', 'Elliptical', 'Rectangular'. Default is 'Circular'.
- **method** (*string object (optional)*) – If phot_meth is 'Aperture', apemethod is the method used to determine the overlap of the aperture on the pixel grid. Possible methods are 'exact', 'subpixel', 'center'. Default is 'center'.

Returns

- **ape_sum** (*1D array*) – Array of flux with new flux appended.
- **ape_sum_err** (*1D array*) – Array of flux uncertainties with new flux uncertainties appended.

`SPCA.Photometry_Companion.`**`Gaussian2D`**(*position*, *amp*, *xo*, *yo*, *sigx*, *sigy*)

Parameters

- **position** (*3D array*) – Meshgrids of x and y indices of pixels. position[:,:,0] = x and position[:,:,1] = y.
- **amp** (*float*) – Amplitude of the 2D Gaussian.
- **xo** (*float*) – x value of the peak of the 2D Gaussian.
- **yo** (*float*) – y value of the peak of the 2D Gaussian.
- **sigx** (*float*) – Width of the 2D Gaussian along the x axis.
- **sigy** (*float*) – Width of the 2D Gaussian along the y axis.

Returns **PSF.ravel()** – z values of the 2D Gaussian raveled.

Return type 1D array

`SPCA.Photometry_Companion.`**`bgsubtract`**(*img_data*, *bg_flux=[]*, *bg_err=[]*, *bounds=(11, 19, 11, 19)*)

Measure the background level and subtracts the background from each frame.

Parameters

- **img_data** (*3D array*) – Data cube of images (2D arrays of pixel values).
- **bg_err** (*1D array (optional)*) – Array of uncertainties on background measurements for previous images. Default if none given is an empty list
- **bounds** (*tuple (optional)*) – Bounds of box around the target to exclude from the background level measurements. Default is (11, 19 ,11, 19).

Returns

- **bgsub_data** (*3D array*) – Data cube of sigma clipped images (2D arrays of pixel values).
- **bg_flux** (*1D array*) – Updated array of background flux measurements for previous images.
- **bg_err** (*1D array*) – Updated array of uncertainties on background measurements for previous images.

`SPCA.Photometry_Companion.`**`binning_data`**(*data*, *size*)

Median bin an array.

Parameters

- **data** (*1D array*) – Array of data to be binned.
- **size** (*int*) – Size of bins.

Returns

- **binned_data** (*1D array*) – Array of binned data.
- **binned_data** (*1D array*) – Array of standard deviation for each entry in binned_data.

`SPCA.Photometry_Companion.`**`centroid_FWM`**(*image_data*, *xo=[]*, *yo=[]*, *wx=[]*, *wy=[]*, *scale=1*, *bounds=(14, 18, 14, 18)*)

Gets the centroid of the target by flux weighted mean and the PSF width of the target.

Parameters

- **img_data** (*3D array*) – Data cube of images (2D arrays of pixel values).
- **xo** (*list (optional)*) – List of x-centroid obtained previously. Default if none given is an empty list.
- **yo** (*list (optional)*) – List of y-centroids obtained previously. Default if none given is an empty list.

- **wx** (*list (optional)*) – List of PSF width (x-axis) obtained previously. Default if none given is an empty list.
- **wy** (*list (optional)*) – List of PSF width (x-axis) obtained previously. Default if none given is an empty list.
- **scale** (*int (optional)*) – If the image is oversampled, scaling factor for centroid and bounds, i.e, give centroid in terms of the pixel value of the initial image.
- **bounds** (*tuple (optional)*) – Bounds of box around the target to exclude background . Default is (11, 19 ,11, 19).

Returns

- **xo** (*list*) – Updated list of x-centroid obtained previously.
- **yo** (*list*) – Updated list of y-centroids obtained previously.
- **wx** (*list*) – Updated list of PSF width (x-axis) obtained previously.
- **wy** (*list*) – Updated list of PSF width (x-axis) obtained previously.

SPCA.Photometry_Companion.**companion_subtraction**(*image_data*, *c_popt*)

SPCA.Photometry_Companion.**get_fnames**(*directory*, *AOR_snip*, *ch*)

Find paths to all the fits files.

Parameters

- **directory** (*string object*) – Path to the directory containing all the Spitzer data.
- **AOR_snip** (*string object*) – Common first characters of data directory eg. 'r579'
- **ch** (*string objects*) – Channel used for the observation eg. 'ch1' for channel 1

Returns

- **fname** (*list*) – List of paths to all bcd.fits files.
- **len(fnames)** (*int*) – Number of fits file found.

SPCA.Photometry_Companion.**get_lightcurve**(*datapath*, *savepath*, *AOR_snip*, *channel*, *subarray*, *save=True*, *save_full='/ch2_datacube_full_AORs579.dat'*, *bin_data=True*, *bin_size=128*, *save_bin='/ch2_datacube_binned128_AORs579.dat'*, *plot=True*, *plot_name='CoRoT-2b.pdf'*, *oversamp=False*, *savecomp=True*, ***kwargs*)

Given a directory, looks for data (bcd.fits files), opens them and performs photometry.

Parameters

- **datapath** (*string object*) – Directory where the spitzer data is stored.
- **savepath** (*string object*) – Directory the outputs will be saved.
- **AORsnip** (*string objects*) – Common first characters of data directory eg. 'r579'
- **channel** (*string objects*) – Channel used for the observation eg. 'ch1' for channel 1
- **subarray** (*bool*) – True if observation were taken in subarray mode. False if observation were taken in full-array mode.
- **save** (*bool (optional)*) – True if you want to save the outputs. Default is True.

- **save_full** (*string object (optional)*) – Filename of the full unbinned output data. Default is '/ch2_datacube_full_AORs579.dat'.
- **bin_data** (*bool (optional)*) – True you want to get binned data. Default is True.
- **bin_size** (*int (optional)*) – If bin_data is True, the size of the bins. Default is 64.
- **save_bin** (*string object (optional)*) – Filename of the full binned output data. Default is '/ch2_datacube_binned_AORs579.dat'.
- **plot** (*bool (optional)*) – True if you want to plot the time resolved lightcurve. Default is True.
- **plot_name** (*string object (optional)*) – If plot and save is True, the filename of the plot to be saved as. Default is True.
- **oversamp** (*bool (optional)*) – True if you want to oversample you image. Default is True.
- **savecomp** (*boold (optional)*) – True is you want to save companion subtracted image. Default is True.
- ****kwargs** (*dictionary*) – Argument passed onto other functions.

Raises *Error :* – If Photometry method is not supported/recognized by this pipeline.

SPCA.Photometry_Companion.**get_time**(*hdu_list*, *time*)

Gets the time stamp for each image.

Parameters

- **hdu_list** – content of fits file.
- **time** (*1D array*) – Time array to append new time stamps to.

Returns **time** – Time array with new values appended.

Return type 1D array

SPCA.Photometry_Companion.**image_template**(*position*, *amp*, *xo*, *yo*, *sigx*, *sigy*, *amp2*, *xo2*, *yo2*, *sigx2*, *sigy2*)

Parameters

- **position** (*3D array*) – Meshgrids of x and y indices of pixels. position[:,:,0] = x and position[:,:,1] = y.
- **amp** (*float*) – Amplitude of the 2D Gaussian.
- **xo** (*float*) – x value of the peak of the target 2D Gaussian.
- **yo** (*float*) – y value of the peak of the target 2D Gaussian.
- **sigx** (*float*) – Width of the 2D Gaussian target along the x axis.
- **sigy** (*float*) – Width of the 2D Gaussian target along the y axis.
- **amp2** (*float*) – Amplitude of the 2D Gaussian.
- **xo2** (*float*) – x value of the peak of the companion 2D Gaussian.
- **yo2** (*float*) – y value of the peak of the companion 2D Gaussian.
- **sigx2** (*float*) – Width of the 2D Gaussian companion along the x axis.
- **sigy2** (*float*) – Width of the 2D Gaussian companion along the y axis.

Returns **PSF.ravel()** – z values of the 2D Gaussian raveled.

Return type 1D array

`SPCA.Photometry_Companion.`**`imagefit`**(*image_data, tossed, popt, pcov, scale=1, tbounds=(9, 20, 9, 20), pinit=(18000, 15, 15, 2, 2, 2000, 12, 13, 1, 1), ub=(25000, 15.3, 15.3, 5, 5, 5000, 12.4, 13.6, 2, 2), lb=(3000, 14.6, 14.6, 0.5, 0.5, 200, 11.7, 12.7, 0, 0)*)

`SPCA.Photometry_Companion.`**`oversampling`**(*image_data, a=2*)

First, substitutes all invalid/sigmaclipped pixel by interpolating the value. Then oversamples the image.

Parameters

- **`image_data`** (*`3D array`*) – Data cube of images (2D arrays of pixel values).
- **`a`** (*`int (optional)`*) – Sampling factor, e.g. if a = 2, there will be twice as much data points in the x and y axis. Default is 2. (Do not recommend larger than 2)

Returns **image_over** – Data cube of oversampled images (2D arrays of pixel values).

Return type 3D array

`SPCA.Photometry_Companion.`**`sigma_clipping`**(*image_data, filenb=0, fname=['not provided'], tossed=0, badframetable=[], bounds=(11, 19, 11, 19), **kwarg*)

Sigma clips bad pixels and mask entire frame if the sigma clipped pixel is too close to the target.

Parameters

- **`image_data`** (*`3D array`*) – Data cube of images (2D arrays of pixel values).
- **`filenb`** (*`int (optional)`*) – Index of current file in the 'fname' list (list of names of files) to keep track of the files that were tossed out. Default is 0.
- **`fname`** (*`list (optional)`*) – list (list of names of files) to keep track of the files that were tossed out.
- **`tossed`** (*`int (optional)`*) – Total number of image tossed out. Default is 0 if none provided.
- **`badframetable`** (*`list (optional)`*) – List of file names and frame number of images tossed out from 'fname'.
- **`bounds`** (*`tuple (optional)`*) – Bounds of box around the target. Default is (11, 19 ,11, 19).

Returns

- **sig_clipped_data** (*3D array*) – Data cube of sigma clipped images (2D arrays of pixel values).
- **tossed** (*int*) – Updated total number of image tossed out.
- **badframetable** (*list*) – Updated list of file names and frame number of images tossed out from 'fname'.

3.1.6 SPCA.Photometry_PLD module

`SPCA.Photometry_PLD.`**`bin_array2D`**(*data, size*)

Median bin PLD stamps.

Parameters

- **`data`** (*`2D array`*) – Array of PLD stamps to be binned.

- **size** (*int*) – Size of bins.

Returns

binned_data (2D array) Array of binned PLD stamps. binned_data_std (2D array) Array of standard deviation for each entry in binned_data.

Return type tuple

`SPCA.Photometry_PLD.`**`get_lightcurve`**(*basepath, AOR_snip, channel, planet, stamp_sizes=[3, 5], save=True, highpassWidth=320, bin_data=True, bin_size=64, showPlots=False, savePlots=True, addStack=False, ignoreFrames=None, maskStars=None, ncpu=4, image_stack_input=None, bg=None, bg_err=None, time=None*)

Given a directory, looks for data (bcd.fits files), opens them and performs PLD "photometry".

Parameters

- **datapath** (*string*) – Directory where the spitzer data is stored.
- **savepath** (*string*) – Directory the outputs will be saved.
- **AORsnip** (*string*) – Common first characters of data directory eg. 'r579'
- **channel** (*string*) – Channel used for the observation eg. 'ch1' for channel 1
- **save** (*bool, optional*) – True if you want to save the outputs. Default is True.
- **save_full** (*string, optional*) – Filename of the full unbinned output data. Default is '/ch2_datacube_full_AORs579.dat'.
- **bin_data** (*bool, optional*) – True you want to get binned data. Default is True.
- **bin_size** (*int, optional*) – If bin_data is True, the size of the bins. Default is 64.
- **save_bin** (*string, optional*) – Filename of the full binned output data. Default is '/ch2_datacube_binned_AORs579.dat'.
- **plot** (*bool, optional*) – True if you want to plot the time resolved lightcurve. Default is True.
- **plot_name** (*string, optional*) – If plot and save is True, the filename of the plot to be saved as. Default is True.
- **planet** (*string, optional*) – The name of the planet. Default is CoRoT-2b.
- **stamp_size** (*int, optional*) – The size of PLD stamp to use (returns stamps that are stamp_size x stamp_size). Only 3 and 5 are currently supported.
- **addStack** (*bool, optional*) – Whether or not to add a background subtraction correction stack. Default is False.
- **stackPath** (*string, optional*) – Path to the background subtraction correction stack.
- **ignoreFrames** (*list, optional*) – frame to remove first-frame systematic).
- **maskStars** (*list, optional*) – An array-like object where each element is an array-like object with the RA and DEC coordinates of a nearby star which should be masked out when computing background subtraction.
- **rerun_photometry** (*bool, optional*) – Whether to overwrite old photometry if it exists. Default is False.

Raises `Error` – If Photometry method is not supported/recognized by this pipeline.

`SPCA.Photometry_PLD.get_pixel_values`(*image*, *cx=15*, *cy=15*, *nbx=3*, *nby=3*)

Median bin PLD stamps.

Parameters

- **image** (*2D array*) – Image to be cut into PLD stamps.
- **P** (*ndarray*) – Previously made PLD stamps to append new stamp to.
- **cx** (*int, optional*) – x-coordinate of the center of the PLD stamp. Default is 15.
- **cy** (*int, optional*) – y-coordinate of the center of the PLD stamp. Default is 15.
- **nbx** (*int, optional*) – Number of pixels to use along the x-axis for the PLD stamp.
- **nby** (*int, optional*) – Number of pixels to use along the y-axis for the PLD stamp.

Returns Updated array of binned PLD stamps including the new stamp.

Return type ndarray

3.1.7 SPCA.Photometry_PSF module

`SPCA.Photometry_PSF.fit_2DGaussian`(*image_stack, scale=1, bounds=(13, 18, 13, 18), defaultCentroid=['median', 'median'], defaultPSFW=['median', 'median'], ncpu=4*)

Gets the centroid of the target by flux weighted mean and the PSF width of the target.

Parameters

- **image_data** (*ndarray*) – Data cube of images (2D arrays of pixel values).
- **scale** (*int, optional*) – If the image is oversampled, scaling factor for centroid and bounds, i.e, give centroid in terms of the pixel value of the initial image.
- **bounds** (*tuple, optional*) – Bounds of box around the target to exclude background . Default is (14, 18, 14, 18).
- **defaultCentroid** (*list, optional*) – Default location for sigma clipped centroids. Default is median centroid position.
- **defaultPSFW** (*list, optional*) – Default width for sigma clipped PSF widths. Default is median of widths.

Returns

xo, yo, wx, wy (list, list, list, list). The updated lists of x-centroid, y-centroid, PSF width (x-axis), and PSF width (y-axis).

Return type tuple

`SPCA.Photometry_PSF.fitgaussian`(*bounds*, *scale*, *i*)

Returns (height, x, y, width_x, width_y) the gaussian parameters of a 2D distribution found by a fit

`SPCA.Photometry_PSF.func`(*arg*)

`SPCA.Photometry_PSF.gaussian`(*height*, *center_x*, *center_y*, *width_x*, *width_y*, *x*, *y*)

Returns a gaussian function with the given parameters

`SPCA.Photometry_PSF.get_lightcurve`(*basepath*, *AOR_snip*, *channel*, *planet*, *save=True*, *highpassWidth=320*, *bin_data=True*, *bin_size=64*, *showPlots=False*, *savePlots=True*, *oversamp=False*, *scale=2*, *saveoversamp=True*, *reuse_oversamp=True*, *addStack=False*, *ignoreFrames=None*, *maskStars=None*, *ncpu=4*, *image_stack_input=None*, *bg=None*, *bg_err=None*, *time=None*)

Given a directory, looks for data (bcd.fits files), opens them and performs PSF photometry.

Parameters

- **AORsnip** (*string*) – Common first characters of data directory eg. 'r579'
- **channel** (*string*) – Channel used for the observation eg. 'ch1' for channel 1
- **planet** (*string, optional*) – The name of the planet.
- **save** (*bool, optional*) – True if you want to save the outputs. Default is True.
- **bin_data** (*bool, optional*) – True you want to get binned data. Default is True.
- **bin_size** (*int, optional*) – If bin_data is True, the size of the bins. Default is 64.
- **oversamp** (*bool, optional*) – True if you want to oversample the image by a factor of 2. Default is False.
- **save_oversamp** (*bool, optional*) – True if you want to save oversampled images. Default is True.
- **reuse_oversamp** (*bool, optional*) – True if you want to reuse oversampled images that were previously saved. Default is False.
- **ignoreFrames** (*list, optional*) – frame to remove first-frame systematic).
- **maskStars** (*list, optional*) – An array-like object where each element is an array-like object with the RA and DEC coordinates of a nearby star which should be masked out when computing background subtraction.
- **ncpu** (*int, optional*) – The number of aperture radii to try at the same time with multiprocessing. Default is 4.

Raises `Error` – If Photometry method is not supported/recognized by this pipeline.

`SPCA.Photometry_PSF.moments`(*starbox*)

Returns (height, x, y, width_x, width_y) the gaussian parameters of a 2D distribution by calculating its moments

`SPCA.Photometry_PSF.update`(*outputs*)

`SPCA.Photometry_PSF.wrapMyFunc`(*arg*)

3.1.8 SPCA.astro_models module

`SPCA.astro_models.area`(*time*, *t_sec*, *per*, *rp*, *inc*, *r2*, *r2off*)

Model the variations in projected area of a bi-axial ellipsoid over time, without assuming elongated axis is the sub-stellar axis.

Parameters

- **time** (*ndarray*) – Array of times at which to calculate the model.
- **t_sec** (*float*) – Time of secondary eclipse.
- **per** (*float*) – Orbital period.

- **rp** (*float*) – Planet radius (in units of stellar radii).
- **inc** (*float*) – Orbital inclination (in degrees).
- **r2** (*float*) – Planet radius along sub-stellar axis (in units of stellar radii).
- **r2off** (*float*) – Angle to the elongated axis with respect to the sub-stellar axis (in degrees).

Returns Modelled projected area of the ellipsoid over time.

Return type ndarray

`SPCA.astro_models.`**`area_noOffset`**(*time*, *t_sec*, *per*, *rp*, *inc*, *r2*)

Model the variations in projected area of a bi-axial ellipsoid over time, assuming elongated axis is the sub-stellar axis.

Parameters

- **time** (*ndarray*) – Array of times at which to calculate the model.
- **t_sec** (*float*) – Time of secondary eclipse.
- **per** (*float*) – Orbital period.
- **rp** (*float*) – Planet radius (in units of stellar radii).
- **inc** (*float*) – Orbital inclination (in degrees).
- **r2** (*float*) – Planet radius along sub-stellar axis (in units of stellar radii).

Returns Modelled projected area of the ellipsoid over time.

Return type ndarray

`SPCA.astro_models.`**`check_phase`**(*checkPhasePhis*, *A*, *B*, *C=0*, *D=0*)

Check if the phasecurve ever dips below zero, implying non-physical negative flux coming from the planet.

Parameters

- **phis** (*ndarray*) – Array of phases in radians at which to calculate the model, e.g. phis=np.linspace(-np.pi,np.pi,1000).
- **A** (*float*) – Amplitude of the first-order cosine term.
- **B** (*float*) – Amplitude of the first-order sine term.
- **C** (*float, optional*) – Amplitude of the second-order cosine term. Default=0.
- **D** (*float, optional*) – Amplitude of the second-order sine term. Default=0.

Returns True if lightcurve implies non-physical negative flux coming from the planet, False otherwise.

Return type bool

`SPCA.astro_models.`**`eclipse`**(*time*, *t0*, *per*, *rp*, *a*, *inc*, *ecc*, *w*, *fp*, *t_sec*)

Get a model secondary eclipse lightcurve.

Parameters

- **time** (*ndarray*) – Array of times at which to calculate the model.
- **t0** (*float*) – Time of inferior conjunction.
- **per** (*float*) – Orbital period.
- **rp** (*float*) – Planet radius (in units of stellar radii).

- **a** (*float*) – Semi-major axis (in units of stellar radii).
- **inc** (*float*) – Orbital inclination (in degrees).
- **ecc** (*float*) – Orbital eccentricity.
- **w** (*float*) – Longitude of periastron (in degrees).
- **fp** (*float*) – Planet-to-star flux ratio.
- **t_sec** (*float*) – Time of secondary eclipse.

Returns flux. The model eclipse light curve.

Return type ndarray

`SPCA.astro_models.`**`fplanet_model`**(*time*, *anom*, *t0*, *per*, *rp*, *a*, *inc*, *ecc*, *w*, *u1*, *u2*, *fp*, *t_sec*, *A*, *B*, *C=0.0*, *D=0.0*, *r2=None*, *r2off=None*)

Model observed flux coming from the planet over time.

Parameters

- **time** (*ndarray*) – Array of times at which to calculate the model.
- **anom** (*ndarray*) – The true anomaly over time.
- **t0** (*float*) – Time of inferior conjunction.
- **per** (*float*) – Orbital period.
- **rp** (*float*) – Planet radius (in units of stellar radii).
- **a** (*float*) – Semi-major axis (in units of stellar radii).
- **inc** (*float*) – Orbital inclination (in degrees).
- **ecc** (*float*) – Orbital eccentricity.
- **w** (*float*) – Longitude of periastron (in degrees).
- **u1** (*float*) – Limb darkening coefficient 1.
- **u2** (*float*) – Limb darkening coefficient 2.
- **fp** (*float*) – Planet-to-star flux ratio.
- **t_sec** (*float*) – Time of secondary eclipse.
- **A** (*float*) – Amplitude of the first-order cosine term.
- **B** (*float*) – Amplitude of the first-order sine term.
- **C** (*float, optional*) – Amplitude of the second-order cosine term. Default=0.
- **D** (*float, optional*) – Amplitude of the second-order sine term. Default=0.
- **r2** (*float, optional*) – Planet radius along sub-stellar axis (in units of stellar radii). Default=None.
- **r2off** (*float, optional*) – Angle to the elongated axis with respect to the sub-stellar axis (in degrees). Default=None.

Returns Observed flux coming from planet over time.

Return type ndarray

`SPCA.astro_models.`**`ideal_lightcurve`**(*time*, *t0*, *per*, *rp*, *a*, *inc*, *ecosw*, *esinw*, *q1*, *q2*, *fp*, *A*, *B*, *C=0*, *D=0*, *r2=None*, *r2off=None*)

Model observed flux coming from the star+planet system over time.

Parameters

- **time** (*ndarray*) – Array of times at which to calculate the model.
- **t0** (*float*) – Time of inferior conjunction.
- **per** (*float*) – Orbital period.
- **rp** (*float*) – Planet radius (in units of stellar radii).
- **a** (*float*) – Semi-major axis (in units of stellar radii).
- **inc** (*float*) – Orbital inclination (in degrees).
- **ecosw** (*float*) – Eccentricity multiplied by the cosine of the longitude of periastron (value between -1 and 1).
- **esinw** (*float*) – Eccentricity multiplied by the sine of the longitude of periastron (value between -1 and 1).
- **q1** (*float*) – Limb darkening coefficient 1, parametrized to range between 0 and 1.
- **q2** (*float*) – Limb darkening coefficient 2, parametrized to range between 0 and 1.
- **fp** (*float*) – Planet-to-star flux ratio.
- **A** (*float*) – Amplitude of the first-order cosine term.
- **B** (*float*) – Amplitude of the first-order sine term.
- **C** (*float, optional*) – Amplitude of the second-order cosine term. Default=0.
- **D** (*float, optional*) – Amplitude of the second-order sine term. Default=0.
- **r2** (*float, optional*) – Planet radius along sub-stellar axis (in units of stellar radii). Default=None.
- **r2off** (*float, optional*) – Angle to the elongated axis with respect to the sub-stellar axis (in degrees). Default=None.

Returns Observed flux coming from star+planet system over time.

Return type ndarray

`SPCA.astro_models.`**`phase_variation`**(*time*, *t_sec*, *per*, *anom*, *ecc*, *w*, *A*, *B*, *C=0*, *D=0*)

Model first- or second-order sinusoidal phase variations.

Parameters

- **time** (*ndarray*) – Array of times at which to calculate the model.
- **t_sec** (*float*) – Time of secondary eclipse.
- **per** (*float*) – Orbital period.
- **anom** (*ndarray*) – The true anomaly over time.
- **ecc** (*float*) – Orbital eccentricity.
- **w** (*float*) – Longitude of periastron (in degrees).
- **A** (*float*) – Amplitude of the first-order cosine term.
- **B** (*float*) – Amplitude of the first-order sine term.
- **C** (*float, optional*) – Amplitude of the second-order cosine term. Default=0.
- **D** (*float, optional*) – Amplitude of the second-order sine term. Default=0.

Returns Modelled phase variations.

Return type ndarray

`SPCA.astro_models.transit_model`(*time*, *t0*, *per*, *rp*, *a*, *inc*, *ecc*, *w*, *u1*, *u2*)

Get a model transit lightcurve.

Parameters

- **time** (*ndarray*) – Array of times at which to calculate the model.
- **t0** (*float*) – Time of inferior conjunction.
- **per** (*float*) – Orbital period.
- **rp** (*float*) – Planet radius (in units of stellar radii).
- **a** (*float*) – Semi-major axis (in units of stellar radii).
- **inc** (*float*) – Orbital inclination (in degrees).
- **ecc** (*float*) – Orbital eccentricity.
- **w** (*float*) – Longitude of periastron (in degrees).
- **u1** (*float*) – Limb darkening coefficient 1.
- **u2** (*float*) – Limb darkening coefficient 2.

Returns

flux, t_secondary, anom (ndarray, float, ndarray). Model transit lightcurve, time of secondary eclipse,

Return type tuple

3.1.9 SPCA.bliss module

`SPCA.bliss.compute_sensMap`(*flux*, *astroModel*, *knot_nrst_lin*, *nBinX*, *nBinY*, *knotNdata*)

Compute the average sensitivity of each knot.

Parameters

- **flux** (*ndarray*) – The flux measurements for each data point.
- **astroModel** (*ndarray*) – The astrophysical model for each data point.
- **knot_nrst_lin** (*ndarray*) – Index of the knot assoicated with each data point.
- **nBinX** (*int*) – The number of knots you want along the x-axis.
- **nBinY** (*int*) – The number of knots you want along the y-axis.
- **knotNdata** (*ndarray*) – The number of data assoicated with each knot.

Returns sensMap (The average flux/astroModel (aka ~ sensitivity) value for each knot).

Return type ndarray

`SPCA.bliss.knot_assiation`(*knots_pos*, *cents*, *nBin*, *nData*)

Find the two knots adjacted to each knot to later be used with linear interpolation.

Parameters

- **knots_pos** (*array*) – The detector coordinate for the x or y center of each knot.
- **cents** (*array*) – The x or y centroids for each data point.
- **nBinX** (*int*) – The number of knots you want along the axis.

- **nData** (*int*) – The number of data points.

Returns

low_bnd (array; the index of the knot to the left of or below each centroid), up_bnd (array; the index of the knot to the right of or above each centroid).

Return type tuple

`SPCA.bliss.precompute`(*flux*, *xdata*, *ydata*, *nBinX=8*, *nBinY=8*, *astroModel=None*, *savepath=None*, *plot=False*, *showPlot=False*)

Precompute all of the knot associations, etc. that are needed to run BLISS in a fitting routine.

Parameters

- **flux** (*ndarray*) – The flux measurements for each data point.
- **xdata** (*ndarray*) – The x-centroid for each data point.
- **ydata** (*ndarray*) – The y-centroid for each data point.
- **nBinX** (*int, optional*) – The number of knots you want along the x-axis.
- **nBinY** (*int, optional*) – The number of knots you want along the y-axis.
- **astroModel** (*ndarray, optional*) – The astrophysical model for each data point.
- **savepath** (*string, optional*) – The full path to where you would like to save plots that can be used to debug BLISS.
- **plot** (*boolean, optional*) – Whether or not you want to make plots that can be used to debug BLISS (default is False).

Returns signal_input (All of the inputs needed to feed into the signal_bliss function)

Return type tuple

3.1.10 SPCA.detec_models module

`SPCA.detec_models.detec_model_GP`(*input_data*, *astroModel*, *gpAmp*, *gpLx*, *gpLy*, *sigF*)

Model the detector systematics with a Gaussian process based on the centroid.

Parameters

- **input_data** (*tuple*) – (flux, xdata, ydata, time, returnGp, astroModel) with dtypes (ndarray, ndarray, ndarray, ndarray, bool, ndarray). Formatted this way to allow for scipy.optimize.minimize optimization.
- **gpAmp** (*float*) – The natural logarithm of the GP covariance amplitude.
- **gpLx** (*float*) – The natural logarithm of the GP covariance lengthscale in x.
- **gpLy** (*float*) – The natural logarithm of the GP covariance lengthscale in y.
- **sigF** (*float*) – The white noise in units of F_star.

Returns The flux variations due to the detector systematics.

Return type ndarray

SPCA.detec_models.**detec_model_PLD**(*input_data*, *astroModel*, *p1_1*, *p2_1*, *p3_1*, *p4_1*, *p5_1*, *p6_1*, *p7_1*, *p8_1*, *p9_1*, *p10_1=0*, *p11_1=0*, *p12_1=0*, *p13_1=0*, *p14_1=0*, *p15_1=0*, *p16_1=0*, *p17_1=0*, *p18_1=0*, *p19_1=0*, *p20_1=0*, *p21_1=0*, *p22_1=0*, *p23_1=0*, *p24_1=0*, *p25_1=0*, *p1_2=0*, *p2_2=0*, *p3_2=0*, *p4_2=0*, *p5_2=0*, *p6_2=0*, *p7_2=0*, *p8_2=0*, *p9_2=0*, *p10_2=0*, *p11_2=0*, *p12_2=0*, *p13_2=0*, *p14_2=0*, *p15_2=0*, *p16_2=0*, *p17_2=0*, *p18_2=0*, *p19_2=0*, *p20_2=0*, *p21_2=0*, *p22_2=0*, *p23_2=0*, *p24_2=0*, *p25_2=0*)

Model the detector systematics with a PLD model.

Parameters

- **input_data** (*tuple*) – (Pgroup, mode) with dtypes (ndarray, string).
- **p0_0** (*float*) – The constant offset term for PLD decorrelation.
- **p1_1--p25_1** (*float*) – The 1st order PLD coefficients for 3x3 or 5x5 PLD stamps.
- **p1_2--p25_2** (*float*) – The 2nd order PLD coefficients for 3x3 or 5x5 PLD stamps.

Returns The flux variations due to the detector systematics.

Return type ndarray

SPCA.detec_models.**detec_model_PSFW**(*input_data*, *astroModel*, *d1=1*, *d2=0*, *d3=0*)

Model the detector systematics with a simple linear model based on the PSF width.

Parameters

- **detec_inputs** (*tuple*) – (x, y, mode) with dtypes (ndarray, ndarray, string). Formatted this way to allow for easy minimization with scipy.optimize.minimize.
- **d1** (*float*) – The constant offset term. #FIX - I don't think this should be here.
- **d2** (*float*) – The slope in sensitivity with the PSF width in the x direction.
- **d3** (*float*) – The slope in sensitivity with the PSF width in the y direction.

Returns The flux variations due to the detector systematics.

Return type ndarray

SPCA.detec_models.**detec_model_bliss**(*signal_input*, *astroModel*)

Model the detector systematics with a BLISS model based on the centroid.

Parameters

- **signal_input** (*tuple*) – (flux, nBinX, nBinY, knotNdata, low_bnd_x, up_bnd_x, low_bnd_y, up_bnd_y, LL_dist, LR_dist, UL_dist, UR_dist, delta_xo, delta_yo, knot_nrst_x, knot_nrst_y, knot_nrst_lin, BLS, NNI) with dtypes (????). # FIX dtypes!
- **astroModel** (*ndarray*) – The modelled astrophysical flux variations.

Returns The flux variations due to the detector systematics.

Return type ndarray

SPCA.detec_models.**detec_model_poly**(*detec_inputs*, *astroModel*, *c1*, *c2*, *c3*, *c4*, *c5*, *c6*, *c7=0*, *c8=0*, *c9=0*, *c10=0*, *c11=0*, *c12=0*, *c13=0*, *c14=0*, *c15=0*, *c16=0*, *c17=0*, *c18=0*, *c19=0*, *c20=0*, *c21=0*, *d1=1*, *d2=0*, *d3=0*, *s0=0*, *s0break=0*, *s1=0*, *s1break=0*, *s2=0*, *s2break=0*, *s3=0*, *s3break=0*, *s4=0*, *s4break=0*, *m1=0*)

Model the detector systematics with a 2D polynomial model based on the centroid.

Parameters

- **detec_inputs** (*tuple*) – (x, y, mode) with dtypes (ndarray, ndarray, string). Formatted this way to allow for easy minimization with scipy.optimize.minimize.
- **c1--c21** (*float*) – The polynomial model amplitudes.

Returns The flux variations due to the detector systematics.

Return type ndarray

SPCA.detec_models.**hside**(*time*, *astroModel*, *s0=0*, *s0break=0*, *s1=0*, *s1break=0*, *s2=0*, *s2break=0*, *s3=0*, *s3break=0*, *s4=0*, *s4break=0*)

Model the detector systematics with a heaviside step function at up to five AOR breaks.

Parameters

- **time** (*ndarray*) – The time.
- **s#** (*float*) – The amplitude of the heaviside step function.
- **s#break** (*float*) – The time of the aor break

Returns The flux variations due to the detector systematics.

Return type ndarray

SPCA.detec_models.**signal**(*p0*, *p0_labels*, *astrofunc*, *astro_labels*, *astro_input*, *detecfuncs*, *detec_labels*, *detec_inputs*)

SPCA.detec_models.**tslope**(*time*, *astroModel*, *m1=0*)

Model the detector systematics with a simple slope in time.

Parameters

- **time** (*ndarray*) – The time.
- **m1** (*float*) – The slope in sensitivity over time with respect to time[0].

Returns The flux variations due to the detector systematics.

Return type ndarray

3.1.11 SPCA.frameDiagnosticsBackend module

SPCA.frameDiagnosticsBackend.**run_diagnostics**(*planet*, *channel*, *AOR_snip*, *basepath*, *addStack*, *highpassWidth=320*, *nsigma=3*, *ncpu=4*, *showPlot=False*, *savePlot=True*)

Run frame diagnostics and choose which frames within a datacube are consistently bad and should be discarded.

Parameters

- **planet** (*string*) – The name of the planet.
- **channel** (*string*) – The channel being analyzed.
- **AOR_snip** (*string*) – AOR snippet used to figure out what folders contain data.
- **basepath** (*string*) – The full path to the folder containing folders for each planet.
- **addStack** (*bool*) – Whether or not to add a background subtraction correction stack (will be automatically selected if present).
- **nsigma** (*float*) – The number of sigma a frame's median value must be off from the median frame in order to be added to ignore_frames.

Returns The frames whose photometry is typically nsigma above/below the median frame and should be removed from photometry.

Return type list

3.1.12 SPCA.freeze module

`SPCA.freeze.`**`get_fitted_params`**(*function*, *dparams*)

`SPCA.freeze.`**`get_lambdaparams`**(*function*)

`SPCA.freeze.`**`load_past_params`**(*path*)

Load the fitted parameters from a previous run.

Parameters **`path`** (*`string`*) – Path to the file containing past mcmc result (must be a table saved as .npy file).

Returns p0 (the previously fitted values)

Return type ndarray

`SPCA.freeze.`**`make_lambdafunc`**(*function*, *p0_labels*, *dparams=[]*, *obj=[]*, *debug=False*)

Create a lambda function called dynamic_funk that will fix the parameters listed in dparams with the values in obj.

Note: The module where the original function is needs to be loaded in this file.

Parameters

- **`function`** (*`string`*) – Name of the original function.
- **`dparams`** (*`list, optional`*) – List of all input parameters the user does not wish to fit (default is None.)
- **`obj`** (*`string, optional`*) – Object containing all initial and fixed parameter values (default is None.)
- **`debug`** (*`bool, optional`*) – If true, will print mystr so the user can read the command because executing it (default is False).

Returns dynamic_funk (the lambda function with fixed parameters.)

Return type function

3.1.13 SPCA.helpers module

`SPCA.helpers.`**`BIC`**(*logL*, *Npar*, *Ndat*)

Compute the Bayesian Information Criterion.

Parameters

- **`logL`** (*`float`*) – The lnlikelihood.
- **`Npar`** (*`int`*) – The number of fitted parameters.
- **`Ndat`** (*`int`*) – The number of data fitted.

Returns The Bayesian Information Criterion.

Return type float

`SPCA.helpers.`**`binValues`**(*values*, *binAxisValues*, *nbin*, *assumeWhiteNoise=False*)

Bin values and compute their binned noise.

Parameters

- **values** (*ndarray*) – An array of values to bin.
- **binAxisValues** (*ndarray*) – Values of the axis along which binning will occur.
- **nbin** (*int*) – The number of bins desired.
- **assumeWhiteNoise** (*bool, optional*) – Divide binned noise by sqrt(nbinned) (True) or not (False, default).

Returns

binned (ndarray; the binned values), binnedErr (ndarray; the binned errors)

Return type tuple

SPCA.helpers.**binnedNoise**(*x*, *y*, *nbin*)

Compute the binned noise (not assuming white noise)

Parameters

- **x** (*ndarray*) – The values along the binning axis.
- **y** (*ndarray*) – The values which should be binned.
- **nbin** (*int*) – The number of bins desired.

Returns The binned noise (not assuming white noise).

Return type ndarray

SPCA.helpers.**chi2**(*data*, *fit*, *err*)

Compute the chi-squared statistic.

Parameters

- **data** (*ndarray*) – The real y values.
- **fit** (*ndarray*) – The fitted y values.
- **err** (*ndarray or float*) – The y error(s).

Returns The chi-squared statistic.

Return type float

SPCA.helpers.**evidence**(*logL*, *Npar*, *Ndat*)

Compute the Bayesian evidence.

Parameters

- **logL** (*float*) – The lnlikelihood.
- **Npar** (*int*) – The number of fitted parameters.
- **Ndat** (*int*) – The number of data fitted.

Returns The Bayesian evidence.

Return type float

SPCA.helpers.**expand_dparams**(*dparams*, *mode*)

Add any implicit dparams given the mode (e.g. GP parameters if using a Polynomial model).

Parameters

- **dparams** (*ndarray*) – A list of strings specifying which parameters shouldn't be fit.
- **mode** (*string*) – The string specifying the detector and astrophysical model to use.

Returns The updated dparams array.

Return type ndarray

`SPCA.helpers.`**`getIngressDuration`**(*p0_mcmc*, *p0_labels*, *p0_obj*, *intTime*)

Compute the transit/eclipse ingress duration in units of datapoints.

Warning - this assumes a circular orbit!

Parameters

- **p0_mcmc** (*ndarray*) – The array containing the fitted values.
- **p0_labels** (*ndarray*) – The array containing all of the names of the fittable parameters.
- **p0_obj** (*object*) – The object containing the default values for non-fitted variables.
- **intTime** (*float*) – The integration time of each measurement.

Returns The transit/eclipse ingress duration in units of datapoints.

Return type float

`SPCA.helpers.`**`getOccultationDuration`**(*p0_mcmc*, *p0_labels*, *p0_obj*, *intTime*)

Compute the full transit/eclipse duration in units of datapoints.

Warning - this assumes a circular orbit!

Parameters

- **p0_mcmc** (*ndarray*) – The array containing the fitted values.
- **p0_labels** (*ndarray*) – The array containing all of the names of the fittable parameters.
- **p0_obj** (*object*) – The object containing the default values for non-fitted variables.
- **intTime** (*float*) – The integration time of each measurement.

Returns The full transit/eclipse duration in units of datapoints

Return type float

`SPCA.helpers.`**`get_data`**(*foldername*, *filename*, *mode*, *foldername_aper=''*, *foldername_psf=''*, *cut=0*)

Retrieve binned data.

Parameters

- **path** (*string*) – Full path to the data file output by photometry routine.
- **mode** (*string*) – The string specifying the detector and astrophysical model to use.
- **path_aper** (*string, optional*) – Full path to the data file output by aperture photometry routine.
- **cut** (*int, optional*) – Number of data points to remove from the start of the arrays.

Returns

flux (ndarray; Flux extracted for each frame), time (ndarray; Time stamp for each frame), xdata (ndarray; X-coordinate of the centroid for each frame), ydata (ndarray; Y-coordinate of the centroid for each frame), psfwx (ndarray; X-width of the target's PSF for each frame), psfwy (ndarray; Y-width of the target's PSF for each frame).

Return type tuple

`SPCA.helpers.`**`get_full_data`**(*foldername*, *filename*, *mode*, *foldername_aper=''*, *foldername_psf=''*, *cut=0*, *nFrames=64*, *ignore=array([]*, *dtype=float64)*)

Retrieve unbinned data.

Parameters

- **path** (*string*) – Full path to the unbinned data file output by photometry routine.
- **mode** (*string*) – The string specifying the detector and astrophysical model to use.
- **path_aper** (*string, optional*) – Full path to the data file output by aperture photometry routine.
- **cut** (*int, optional*) – Number of data points to remove from the start of the arrays.
- **nFrames** (*int, optional*) – The number of frames that were binned together in the binned data.
- **ignore** (*ndarray, optional*) – Array specifying which frames were found to be bad and should be ignored.

Returns

flux (ndarray; Flux extracted for each frame), time (ndarray; Time stamp for each frame), xdata (ndarray; X-coordinate of the centroid for each frame), ydata (ndarray; Y-coordinate of the centroid for each frame), psfwx (ndarray; X-width of the target's PSF for each frame), psfwy (ndarray; Y-width of the target's PSF for each frame).

Return type tuple

SPCA.helpers.**get_p0**(*dparams*, *obj*)

Initialize the p0 variable to the defaults.

Parameters

- **dparams** (*ndarray*) – A list of strings specifying which parameters shouldn't be fit.
- **obj** (*object*) – An object containing the default values for all fittable parameters. #FIX: change this to dict later

Returns

p0 (ndarray; the initialized values), fit_params (ndarray; the names of the fitted variables), fancy_labels (ndarray; the nicely formatted names of the fitted variables)

Return type tuple

SPCA.helpers.**lnlike**(*p0*, *flux*, *mode*, *signal_func*, *signal_inputs*)

Evaluate the ln-likelihood at the position p0.

Note: We assumine that we are always fitting for the photometric scatter (sigF).

Parameters

- **p0** (*ndarray*) – The array containing the n-D position to evaluate the log-likelihood at.
- **p0_labels** (*ndarray*) – An array containing the names of the fitted parameters.
- **signalfunc** (*function*) – The super function to model the astrophysical and detector functions.
- **signal_input** (*list*) – The collection of other assorted variables required for signalfunc beyond just p0.

Returns The ln-likelihood evaluated at the position p0.

Return type float

SPCA.helpers.**lnprior_custom**(*p0*, *gpriorInds*, *priors*, *errs*, *upriorInds*, *uparams_limits*, *gammaInd*)

SPCA.helpers.**lnprior_gamma**(*p0*, *priorInd*, *shape*, *rate*)

SPCA.helpers.**lnprior_gaussian**(*p0*, *priorInds*, *priors*, *errs*)

SPCA.helpers.**lnprior_uniform**(*p0*, *priorInds*, *limits*)

SPCA.helpers.**lnprob**(*p0*, *flux*, *mode*, *p0_labels*, *signal_func*, *signal_inputs*, *gpriorInds*, *priors*, *errs*, *upriorInds*, *uparams_limits*, *gammaInd*, *positivity_func=None*, *positivity_labels=None*)

Evaluate the ln-probability of the signal function at the position p0, including priors.

Parameters

- **p0** (*ndarray*) – The array containing the n-D position to evaluate the log-likelihood at.
- **p0_labels** (*ndarray*) – An array containing the names of the fitted parameters.
- **signalfunc** (*function*) – The super function to model the astrophysical and detector functions.
- **lnpriorfunc** (*function*) – The function to evaluate the default ln-prior.
- **signal_input** (*list*) – The collection of other assorted variables required for signalfunc beyond just p0.
- **checkPhasePhis** (*ndarray*) – The phase angles to use when checking that the phase-curve is always positive.
- **lnpriorcustom** (*function, optional*) – An additional function to evaluate the a user specified ln-prior function (default is None).

Returns The ln-probability evaluated at the position p0.

Return type float

SPCA.helpers.**loglikelihood**(*data*, *fit*, *err*)

Compute the lnlikelihood.

Parameters

- **data** (*ndarray*) – The real y values.
- **fit** (*ndarray*) – The fitted y values.
- **err** (*ndarray or float*) – The y error(s).

Returns The lnlikelihood.

Return type float

SPCA.helpers.**signal_params**()

3.1.14 SPCA.make_plots module

SPCA.make_plots.**look_for_residual_correlations**(*time*, *flux*, *xdata*, *ydata*, *psfxw*, *psfyw*, *residuals*, *p0_mcmc*, *p0_labels*, *p0_obj*, *mode*, *savepath=None*, *showPlot=False*)

SPCA.make_plots.**plot_centroids**(*xdata0*, *ydata0*, *xdata*, *ydata*, *savepath=''*, *showPlot=False*)

Makes a multi-panel plot from photometry outputs.

Parameters

- **xdata0** (*1D array*) – Initial modelled the fluxes for each time stamps. Discarded points not removed.

- **ydata0** (*1D array*) – Initial modelled astrophysical flux variation for each time stamps. Discarded points not removed.
- **xdata** (*1D array*) – Initial modelled the fluxes for each time stamps. Discarded points removed.
- **ydata** (*1D array*) – Initial modelled astrophysical flux variation for each time stamps. Discarded points removed.
- **savepath** (*string*) – Path to directory where the plot will be saved

Returns None

SPCA.make_plots.**plot_knots**(*xdata*, *ydata*, *tmask_good_knotNdata*, *knots_x*, *knots_y*, *knots_x_mesh*, *knots_y_mesh*, *knotNdata*, *savepath=None*, *showPlot=False*)

Plot the Bliss map

SPCA.make_plots.**plot_model**(*time*, *flux*, *astro*, *detec*, *breaks*, *savepath=None*, *plotName='Initial_Guess.pdf'*, *plotTrueAnomaly=False*, *nbin=None*, *showPlot=False*, *fontsize=24*, *plot_peritime=False*)

SPCA.make_plots.**plot_photometry**(*time0*, *flux0*, *xdata0*, *ydata0*, *psfxw0*, *psfyw0*, *time*, *flux*, *xdata*, *ydata*, *psfxw*, *psfyw*, *breaks=[]*, *savepath=None*, *peritime=''*, *showPlot=False*)

Makes a multi-panel plot from photometry outputs.

Parameters

- **time0** (*1D array*) – Array of time stamps. Discarded points not removed.
- **flux0** (*1D array*) – Array of flux values for each time stamps. Discarded points not removed.
- **xdata0** (*1D array*) – Initial modelled the fluxes for each time stamps. Discarded points not removed.
- **ydata0** (*1D array*) – Initial modelled astrophysical flux variation for each time stamps. Discarded points not removed.
- **psfxw0** (*1D array*) – Point-Spread-Function (PSF) width along the x-direction. Discarded points not removed.
- **psfyw0** (*1D array*) – Point-Spread-Function (PSF) width along the x-direction. Discarded points not removed.
- **time** (*1D array*) – Array of time stamps. Discarded points removed.
- **flux** (*1D array*) – Array of flux values for each time stamps. Discarded points removed.
- **xdata** (*1D array*) – Initial modelled the fluxes for each time stamps. Discarded points removed.
- **ydata** (*1D array*) – Initial modelled astrophysical flux variation for each time stamps. Discarded points removed.
- **psfxw** (*1D array*) – Point-Spread-Function (PSF) width along the x-direction. Discarded points removed.
- **psfyw** (*1D array*) – Point-Spread-Function (PSF) width along the x-direction. Discarded points removed.
- **break** (*1D array*) – Time of the breaks from one AOR to another.
- **savepath** (*string*) – Path to directory where the plot will be saved
- **pertime** (*float*) – Time of periapsis

Returns None

SPCA.make_plots.**plot_rednoise**(*residuals*, *minbins*, *ingrDuration*, *occDuration*, *intTime*, *mode*, *savepath=None*, *showPlot=True*, *showtxt=True*, *savetxt=False*, *fontsize=10*)

SPCA.make_plots.**triangle_colors**(*all_data*, *firstEcl_data*, *transit_data*, *secondEcl_data*, *fname=None*, *showPlot=False*)

Make a triangle plot like figure to help look for any residual correlations in the data.

Parameters

- **all_data** (*list*) – A list of the all of the xdata, ydata, psfxw, psfyw, flux, residuals.
- **firstEcl_data** (*list*) – A list of the xdata, ydata, psfxw, psfyw, flux, residuals during the first eclipse.
- **transit_data** (*list*) – A list of the xdata, ydata, psfxw, psfyw, flux, residuals during the transit.
- **secondEcl_data** (*list*) – A list of the xdata, ydata, psfxw, psfyw, flux, residuals during the second eclipse.
- **fname** (*string, optional*) – The savepath for the plot (or None if you want to return the figure instead).

Returns None

SPCA.make_plots.**walk_style**(*chain*, *labels*, *interv=10*, *fname=None*, *showPlot=False*)

Make a plot showing the evolution of the walkers throughout the emcee sampling.

Parameters

- **chain** (*ndarray*) – The ndarray accessed by calling sampler.chain when using emcee
- **labels** (*ndarray*) – The fancy labels for each dimension
- **interv** (*int*) – Take every 'interv' element to thin out the plot
- **name** (*string, optional*) – The savepath for the plot (or None if you want to return the figure instead).
- **showPlot** (*bool, optional*) – Whether or not you want to show the plotted figure.

Returns None

3.1.15 SPCA.make_plots_custom module

SPCA.make_plots_custom.**plot_bestfit**(*x*, *flux*, *astro*, *detec*, *mode*, *breaks*, *savepath=None*, *showplot=True*, *peritime=-inf*, *nbin=None*, *fontsize=10*)

SPCA.make_plots_custom.**plot_init_guess**(*time*, *data*, *astro*, *detec_full*, *savepath*)

Makes a multi-panel plots for the initial light curve guesses. params: ——-

time [1D array] array of time stamps

data [1D array] array of flux values for each time stamps

astro [1D array] initial modelled astrophysical flux variation for each time stamps

detec_full [1D array] initial modelled flux variation due to the detector for each time stamps

savepath [str] path to directory where the plot will be saved

none

`SPCA.make_plots_custom.`**`plot_photometry`**(*time0*, *flux0*, *xdata0*, *ydata0*, *psfxw0*, *psfyw0*, *time*, *flux*, *xdata*, *ydata*, *psfxw*, *psfyw*, *breaks*, *savepath*, *peritime*)

Makes a multi-panel plot from photometry outputs. params: ——-

time0 [1D array] array of time stamps. Discarded points not removed.

flux0 [1D array] array of flux values for each time stamps. Discarded points not removed.

xdata0 [1D array] initial modelled the fluxes for each time stamps. Discarded points not removed.

ydata0: 1D array initial modelled astrophysical flux variation for each time stamps. Discarded points not removed.

psfxw0: 1D array Point-Spread-Function (PSF) width along the x-direction. Discarded points not removed.

psfyw0: 1D array Point-Spread-Function (PSF) width along the x-direction. Discarded points not removed.

time [1D array] array of time stamps. Discarded points removed.

flux [1D array] array of flux values for each time stamps. Discarded points removed.

xdata [1D array] initial modelled the fluxes for each time stamps. Discarded points removed.

ydata [1D array] initial modelled astrophysical flux variation for each time stamps. Discarded points removed.

psfxw [1D array] Point-Spread-Function (PSF) width along the x-direction. Discarded points removed.

psfyw [1D array] Point-Spread-Function (PSF) width along the x-direction. Discarded points removed.

break [1D array] time of the breaks from one AOR to another.

savepath [str] path to directory where the plot will be saved

none

`SPCA.make_plots_custom.`**`plot_psf_dependence`**(*time*, *flux*, *detec_guess*, *astro_guess*, *psfxw*, *psfyw*, *breaks*, *savepath*, *peritime*)

3.1.16 SPCA.photometryBackend module

3.1.17 Module contents

CHAPTER 4

Indices and tables

- genindex
- modindex

CHAPTER 5

Contributions

Lisa Dang contributed the initial code and idea for SPCA, and wrote much of the photometry and fitting routines, as well as the polynomial and PLD decorrelation methods.

Both authors worked extensively to debug the code and simplify the user experience.

Taylor James Bell further generalized and streamlined SPCA, allowing it to be run quickly and easily for any given planet with minimal effort. Taylor also contributed most of the documentation, and the GP decorrelation method.

CHAPTER 6

Acknowledgements

We thank Joel Schwartz for his aid in writing out BLISS decorrelation method. We also thank Dylan Keating for alpha testing SPCA. We thank Antoine Darveau-Bernier for writing code to parse through the Exoplanet Archive data and select the best constrained value for each parameter in the database (code can be found here).

CHAPTER 7

License & Attribution

Copyright © 2018-2020 Lisa Dang & Taylor James Bell.

SPCA is free software made available under the GPL3 License. For details see the LICENSE.

If you make use of SPCA in your work, please cite the Dang et al. (2018) paper that was the first to use this pipeline (arXiv, ADS, BibTeX).

Python Module Index

S

SPCA, 37
SPCA.astro_models, 22
SPCA.bliss, 26
SPCA.Decorrelation_helper, 8
SPCA.detec_models, 27
SPCA.frameDiagnosticsBackend, 29
SPCA.freeze, 30
SPCA.helpers, 30
SPCA.make_plots, 34
SPCA.make_plots_custom, 36
SPCA.Photometry_Aperture, 9
SPCA.Photometry_Common, 12
SPCA.Photometry_Companion, 15
SPCA.Photometry_PLD, 19
SPCA.Photometry_PSF, 21

Index

A

A_photometry() (*in module SPCA.Photometry_Aperture*), 9
A_photometry() (*in module SPCA.Photometry_Companion*), 15
area() (*in module SPCA.astro_models*), 22
area_noOffset() (*in module SPCA.astro_models*), 23

B

bgsubtract() (*in module SPCA.Photometry_Common*), 12
bgsubtract() (*in module SPCA.Photometry_Companion*), 16
BIC() (*in module SPCA.helpers*), 30
bin_all_data() (*in module SPCA.Photometry_Aperture*), 10
bin_array() (*in module SPCA.Photometry_Common*), 13
bin_array2D() (*in module SPCA.Photometry_PLD*), 19
binnedNoise() (*in module SPCA.helpers*), 31
binning_data() (*in module SPCA.Photometry_Companion*), 16
binValues() (*in module SPCA.helpers*), 30
burnIn() (*in module SPCA.Decorrelation_helper*), 8

C

centroid_FWM() (*in module SPCA.Photometry_Aperture*), 10
centroid_FWM() (*in module SPCA.Photometry_Companion*), 16
check_phase() (*in module SPCA.astro_models*), 23
chi2() (*in module SPCA.helpers*), 31
clip_data() (*in module SPCA.Photometry_Common*), 13
companion_subtraction() (*in module SPCA.Photometry_Companion*), 17
compare_RMS() (*in module SPCA.Photometry_Aperture*), 11
compute_sensMap() (*in module SPCA.bliss*), 26
create_folder() (*in module SPCA.Photometry_Common*), 13

D

detec_model_bliss() (*in module SPCA.detec_models*), 28
detec_model_GP() (*in module SPCA.detec_models*), 27
detec_model_PLD() (*in module SPCA.detec_models*), 27
detec_model_poly() (*in module SPCA.detec_models*), 28
detec_model_PSFW() (*in module SPCA.detec_models*), 28
downloadExoplanetArchive() (*in module SPCA.Decorrelation_helper*), 8

E

eclipse() (*in module SPCA.astro_models*), 23
evidence() (*in module SPCA.helpers*), 31
expand_dparams() (*in module SPCA.helpers*), 31

F

findPhotometry() (*in module SPCA.Decorrelation_helper*), 8
fit_2DGaussian() (*in module SPCA.Photometry_PSF*), 21
fitgaussian() (*in module SPCA.Photometry_PSF*), 21
fplanet_model() (*in module SPCA.astro_models*), 24
func() (*in module SPCA.Photometry_Aperture*), 11
func() (*in module SPCA.Photometry_PSF*), 21

G

gaussian() (*in module SPCA.Photometry_PSF*), 21
Gaussian2D() (*in module SPCA.Photometry_Companion*), 15

get_data() (*in module SPCA.helpers*), 32
get_fitted_params() (*in module SPCA.freeze*), 30
get_fnames() (*in module SPCA.Photometry_Common*), 13
get_fnames() (*in module SPCA.Photometry_Companion*), 17
get_full_data() (*in module SPCA.helpers*), 32
get_lambdaparams() (*in module SPCA.freeze*), 30
get_lightcurve() (*in module SPCA.Photometry_Aperture*), 11
get_lightcurve() (*in module SPCA.Photometry_Companion*), 17
get_lightcurve() (*in module SPCA.Photometry_PLD*), 20
get_lightcurve() (*in module SPCA.Photometry_PSF*), 21
get_p0() (*in module SPCA.helpers*), 33
get_photon_limit() (*in module SPCA.Decorrelation_helper*), 8
get_photon_limit_oldData() (*in module SPCA.Decorrelation_helper*), 8
get_pixel_values() (*in module SPCA.Photometry_PLD*), 20
get_stacks() (*in module SPCA.Photometry_Common*), 13
get_time() (*in module SPCA.Photometry_Common*), 14
get_time() (*in module SPCA.Photometry_Companion*), 18
getIngressDuration() (*in module SPCA.helpers*), 32
getOccultationDuration() (*in module SPCA.helpers*), 32
getTstarBright() (*in module SPCA.Decorrelation_helper*), 8

H

highpassflist() (*in module SPCA.Photometry_Common*), 14
hside() (*in module SPCA.detec_models*), 29

I

ideal_lightcurve() (*in module SPCA.astro_models*), 24
image_template() (*in module SPCA.Photometry_Companion*), 18
imagefit() (*in module SPCA.Photometry_Companion*), 19

K

knot_assiation() (*in module SPCA.bliss*), 26

L

lnlike() (*in module SPCA.helpers*), 33
lnprior_custom() (*in module SPCA.helpers*), 33
lnprior_gamma() (*in module SPCA.helpers*), 33
lnprior_gaussian() (*in module SPCA.helpers*), 33
lnprior_uniform() (*in module SPCA.helpers*), 34
lnprob() (*in module SPCA.helpers*), 34
load_past_params() (*in module SPCA.freeze*), 30
loadArchivalData() (*in module SPCA.Decorrelation_helper*), 8
loadCustomData() (*in module SPCA.Decorrelation_helper*), 8
loglikelihood() (*in module SPCA.helpers*), 34
look_for_residual_correlations() (*in module SPCA.make_plots*), 34

M

make_lambdafunc() (*in module SPCA.freeze*), 30
moments() (*in module SPCA.Photometry_PSF*), 22

N

noisepixparam() (*in module SPCA.Photometry_Aperture*), 12

O

oversampling() (*in module SPCA.Photometry_Common*), 14
oversampling() (*in module SPCA.Photometry_Companion*), 19

P

phase_variation() (*in module SPCA.astro_models*), 25
plot_bestfit() (*in module SPCA.make_plots_custom*), 36
plot_centroids() (*in module SPCA.make_plots*), 34
plot_init_guess() (*in module SPCA.make_plots_custom*), 36
plot_knots() (*in module SPCA.make_plots*), 35
plot_model() (*in module SPCA.make_plots*), 35
plot_photometry() (*in module SPCA.make_plots*), 35
plot_photometry() (*in module SPCA.make_plots_custom*), 36
plot_psf_dependence() (*in module SPCA.make_plots_custom*), 37
plot_rednoise() (*in module SPCA.make_plots*), 36
precompute() (*in module SPCA.bliss*), 27
prepare_image() (*in module SPCA.Photometry_Common*), 14
prepare_images() (*in module SPCA.Photometry_Common*), 14
print_MCMC_results() (*in module SPCA.Decorrelation_helper*), 9

R

reload_old_fit() (*in module SPCA.Decorrelation_helper*), 9
replace_clipped() (*in module SPCA.Photometry_Common*), 14
rolling_clip() (*in module SPCA.Photometry_Common*), 14
run_diagnostics() (*in module SPCA.frameDiagnosticsBackend*), 29

S

setup_gpriors() (*in module SPCA.Decorrelation_helper*), 9
sigma_clipping() (*in module SPCA.Photometry_Common*), 14
sigma_clipping() (*in module SPCA.Photometry_Companion*), 19
signal() (*in module SPCA.detec_models*), 29
signal_params() (*in module SPCA.helpers*), 34
SPCA (*module*), 37
SPCA.astro_models (*module*), 22
SPCA.bliss (*module*), 26
SPCA.Decorrelation_helper (*module*), 8
SPCA.detec_models (*module*), 27
SPCA.frameDiagnosticsBackend (*module*), 29
SPCA.freeze (*module*), 30
SPCA.helpers (*module*), 30
SPCA.make_plots (*module*), 34
SPCA.make_plots_custom (*module*), 36
SPCA.Photometry_Aperture (*module*), 9
SPCA.Photometry_Common (*module*), 12
SPCA.Photometry_Companion (*module*), 15
SPCA.Photometry_PLD (*module*), 19
SPCA.Photometry_PSF (*module*), 21

T

test_circularAperture() (*SPCA.Photometry_Aperture.TestAperturehotometryMethods method*), 10
test_FWM_centroiding() (*SPCA.Photometry_Aperture.TestAperturehotometryMethods method*), 10
TestAperturehotometryMethods (*class in SPCA.Photometry_Aperture*), 10
transit_model() (*in module SPCA.astro_models*), 26
triangle_colors() (*in module SPCA.make_plots*), 36
tslope() (*in module SPCA.detec_models*), 29

U

update() (*in module SPCA.Photometry_Aperture*), 12
update() (*in module SPCA.Photometry_PSF*), 22

W

walk_style() (*in module SPCA.make_plots*), 36
wrapMyFunc() (*in module SPCA.Photometry_Aperture*), 12
wrapMyFunc() (*in module SPCA.Photometry_PSF*), 22