Palomar+Keck Observing Quick Reference

Z. Vanderbosch et al.

December 1, 2023

Contents

1 Introduction & Motivation

One shortfall I have noticed in our current CHIMERA data reduction stream is that users have very little control over which comparison star, or stars, they use to create the final divided light curve. To test different comparison stars, one would have to run the ULTRACAM pipeline multiple times (e.g. `setaper` and `reduce`), setting the first reference star aperture to be a different object each time. This is a very cumbersome and time-consuming process, but I actually developed software to deal with exactly this problem for the McDonald 2.1m data reduction pipeline while a Ph.D. student at UT Austin. This software is a Python package called `phot2lc` (https://phot2lc.readthedocs.io/en/latest/), and I have now added in the capability for `phot2lc` to handle data products from both the ULTRACAM and HiPERCAM pipelines. Below is a description of both our current CHIMERA pipeline, and how `phot2lc` can be used in place of certain steps.

2 The Current Pipeline

Our current pipeline stream can be broken down into the following steps:

1. `chimera_preproc.py`: Performs image calibration steps (bias subtraction and flat-field normalization), runs checks on per-exposure timestamps, and creates properly formatted `ucm` files that can be loaded by the ULTRACAM pipeline.

2. `setaper`: Part of the ULTRACAM pipeline suite, this program will load and display one of your science images, typically the first, and allow you to choose the positions of target and comparison star apertures. You can choose as many comparison stars as you want, but ultimately only the first one will be used later on to create the final divided light curve.

3. `reduce`: Also part of the ULTRACAM pipeline suite, this program performs aperture photometry on all images for all apertures chosen in the `setaper` step. It outputs a file typically named `output.log` containing the background-subtracted photometric counts in each aperture, and additional info such as count uncertainties and sky background counts.

4. `log2lcurve.py`: This program loads in the output from `reduce`, calculates a divided light curve using just the first comparison star chosen in `setaper`, and calculates barycentric corrections to timestamps. It outputs the final data product `lc.dat`, a file containing columns with time, normalized flux, and normalized flux uncertainties.

In practice, running all of these commands from the command line on the `schoty` computer would look like the following:

```
source ~jvanroestel/.bashrc_lcurve
conda activate lcurve_env
chimera_preproc.py --dir=CHIMERA_RED/20211202 -c -p -u
setaper ucm_files/r_Chimera_00000.ucm
reduce source=u reduce.red output.log ucm.list
log2lcurve.py --ra=1.234 --dec=5.678 output.log
```

3 Replacing log2lcurve with phot2lc

`phot2lc` (full documentation here) is a light curve extraction program, meaning it does not perform aperture photometry, but rather loads in photometry and timestamp data generated by other programs, and provides users with a set of tools to perform optimal light curve extraction. Optimal light curve extraction in this sense means attempting to find the combination of comparison stars and aperture size that minimizes light curve scatter. It can do this process automatically, testing all possible combinations, but also provides additional tools such as manual outlier removal, sigma clipping, polynomial detrending, and manual comparison star selection/exclusion.

With respect to our current pipeline stream, `phot2lc` is only suited to replace the final step, `log2lcurve.py`, which loads in the output from the ULTRACAM pipeline and generates the final divided light curve. Once `phot2lc` is installed and configured, its use is very straightforward. After sourcing the `bashrc` file from `/home/zvanderbosch` and activating the `zachpy` conda environment, you simply type `phot2lc` on the command line from the directory containing the output from the ULTRACAM pipeline, followed by the option `-o` to indicate your object's name, e.g.:

```
source ~zvanderbosch/.bashrc
conda activate zachpy
phot2lc -o ZTFJ0011+3344
```

[WARNING: Once you have sourced the `.bashrc` file in `/home/zvanderbosch/`, the commands you are familiar with for running the ULTRACAM pipeline, such as `setaper` and `reduce`, are now HiPERCAM commands! You will have to source `~jvanroestel/.bashrc_lcurve` if you wish to use the ULTRACAM pipeline again after using `phot2lc`.]

The object name you provide must match up with an object contained in the stars.dat file. This is a local file, currently stored on `schoty` at `/scr/zvanderbosch/data/`, that contains the RA and Dec coordinates needed to do barycentric corrections for your object. If you don't already have your object entered into the stars.dat file, you can enter it manually following the format guidelines here, or you can use the convenience function `addstar` provided with the `phot2lc` package. With this function, you simply provide your object name and RA and Dec coordinates that can be parsed by Astropy, such as the following examples:

```
addstar ZTFJ0011+3344 '00 11 22.2' '+33 44 55.5'
addstar ZTFJ0011+3344 00:11:22.2 +33:44:55.5
addstar ZTFJ0011+3344 2.8425 33.74875
```

You only have to enter an object into the `stars.dat` file once, so in subsequent reductions of data for your object you just need to run the `phot2lc` command.

3.1 Checking and Changing phot2lc Configuration

If your star is definitely included in the stars.dat file but you are seeing errors when running `phot2lc`, you should check to make sure `phot2lc` is properly configured for the photometry pipeline and telescope/instrument you used. To do this, simply type the following command:

```
photconfig
```

This will open the `config.dat` file located in the install directory of the `phot2lc` package, and lead you through a series of prompts to keep or change any of the configuration values. If you are using the ULTRACAM pipeline with data from CHIMERA, you should have the following settings:

```
author            = sys-user
image_list_name   = ucm.list
pixloc_name       = aperture.ape
photbase_name     = output
stardat_location  = /scr/zvanderbosch/data/stars.dat
default_telescope = p200
default_source    = ucm
default_image     = None
default_object    = None
```

[WARNING: With `author = sys-user`, `phot2lc` will automatically use your username on schoty when adding metadata into the final reduced light curve file, saying who performed the light curve extraction. You can change this parameter to your full name if desired, but this could be problematic since the `config.dat` file is globally used by all `phot2lc` users, and other users may forget to change the value back to `sys-user` or their own full name. If you wish to use a different name, it would be better to use the `phot2lc` command line option `-a` or `--author`, which overrides the value in the `config.dat` file.]

If you are using the HiPERCAM pipeline with data from CHIMERA, only the following two parameters need to be changed:

```
image_list_name   = hcm.lis
default_source    = hcm
```

4 Replacing ULTRACAM with HiPERCAM

The HiPERCAM pipeline has been installed into the `zachpy` environment. To run it, first source the necessary `bashrc` file:

```
source ~zvanderbosch/.bashrc
conda activate zachpy
```

Then move into the directory containing the raw CHIMERA data for a given night and use the following commands:

```
chimera_preproc.py --dir=./ -c -p -u
cd ../<date>_reduced/<object_dir>/
setaper hcm_files/Chimera_00000.hcm aperture.ape 10 20 30 0.5 1024.5 0.5 1024.5 True p 1 99
reduce source=hf hcm.lis False reduce.red output.log 0 True False
```

Here, `<date>` and `<object_dir>` need to be replaced with the date and object directory of interest. For the `setaper` and `reduce` commands, all of the required input parameters are shown being passed on the command line. You can run `setaper` and `reduce` without providing any of the parameter values, but you will then be prompted to check or change each parameter one by one. The values shown above will work for all standard reductions of CHIMERA data. For the `setaper` command, it's worth noting that all of the pixel-value parameters (e.g. xlo = 0.5 and xhi = 1024.5) are for the UNBINNED values, so they do not need to be changed for different binnings of data.

The `chimera_preproc.py` script is also a different version built specifically for HiPERCAM data. This version of `chimera_preproc` lives in the directory `/home/zvanderbosch/software/chimera/` along with some associated python scripts. When you run the `chimera_preproc` script, it will copy a default `reduce.red` file from the directory `/scr/zvanderbosch/software/hipercam/` into each reduced object directory. The parameters in this file were chosen to match as closely as possible the default parameters we have been using for the ULTRACAM pipeline, though this was sometimes difficult to achieve given the change in parameter names between the two pipelines.

4.1 Comparing Photometric Pipeline Performance

In this section, I have done a brief comparison of the final light curve quality from the two pipelines ULTRACAM and HiPERCAM, to ensure that the HiPERCAM pipeline is performing as well as expected.

4.1.1 HCAM versus UCAM with Optimal Extraction

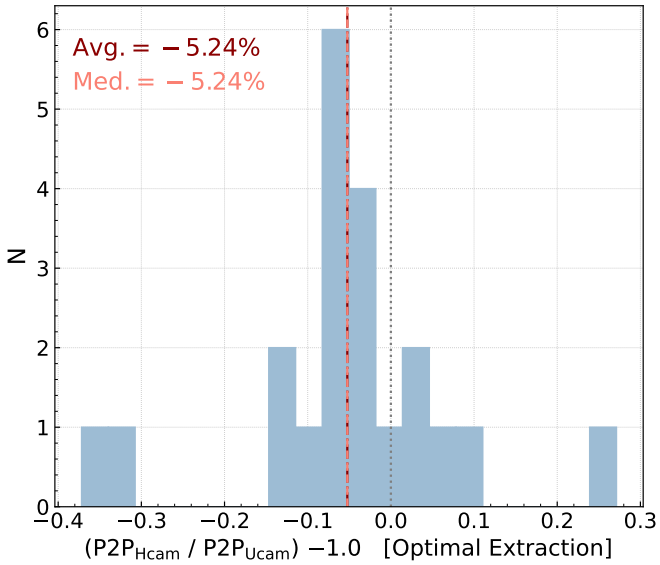

Figure 1: Comparison of divided light curve scatter between ULTRACAM and HiPERCAM pipelines using optimal extraction method.

In this experiment, I performed aperture photometry for 21 sources observed on 15 separate nights between 2022 March 23 and 2023 April 26. For each source, I ran both the ULTRACAM and HiPERCAM pipelines on the same calibrated science images, both using the default setup parameters in their respective `reduce.red` files with optimal extraction. I then used `phot2lc` to extract the light curves, being careful to use the exact same comparison stars in both cases to generate the divided light curves. I then made an estimate of the scatter of each light curve by calculating the average point-2-point scatter (P2P), and used the ratio $\mathrm{P2P_{Hcam}/P2P_{Ucam}}$ to estimate the relative difference in light curve quality between the two pipelines. Figure 1 shows a histogram of these ratios. 16 out of 21 objects exhibit reduced scatter in their HiPERCAM light curves, and on average the HiPERCAM pipeline reduced light curve scatter by around 5% compared to the ULTRACAM light curves.

4.1.2 HCAM Optimal versus Normal Extraction

In this experiment, I performed aperture photometry for the exact same sources as above, this time using just the HiPERCAM pipeline but in one case using normal extraction (unweighted pixels), and in the other case using optimal extraction (pixels weighted by a Moffat profile), and keeping all other input parameters in the `reduce.red` file the same. I again used `phot2lc` to extract the divided light curves, using the same comparison stars for each target.

I used the P2P metric to quantify the scatter of each light curve, and used the ratio $\mathrm{P2P_{Optimal}/P2P_{Normal}}$ to estimate the relative difference in light curve quality between the two extraction methods. Fig-

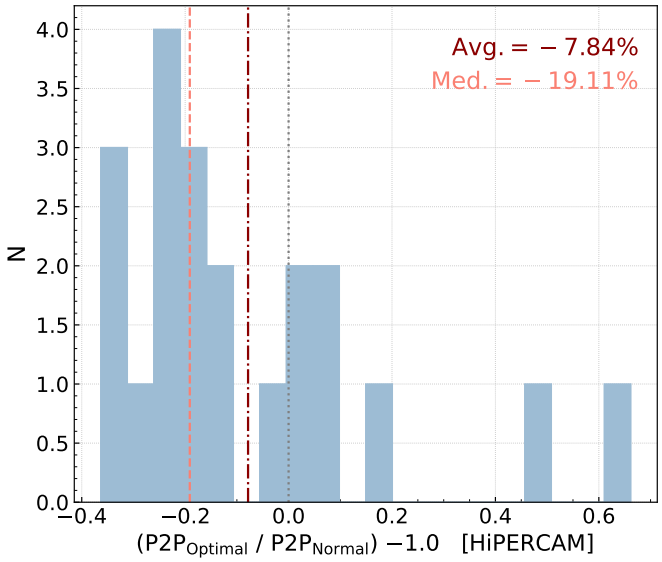

Figure 2: Comparison of divided light curve scatter between ULTRACAM and HiPERCAM pipelines using optimal extraction method.

ure 2 shows a histogram of these ratios. 14 out of 21 sources exhibit reduced scatter when using the optimal extraction method. The average and median reduction of scatter when using the optimal extraction method are about 8% and 19%, respectively.

To provide a more complete picture of the comparison between the different photometric extraction methods, Figure 3 shows the object-by-object comparison of extracted light curves, with the left column overplotting the final divided light curves, the middle column showing the point-by-point difference in flux between the light curves, and the right column comparing the periodograms of the two light curves. This makes it clear that the light curves do not only differ in terms of photometric scatter, but can also show slightly differing long-term trends, and in some cases also show differences in the appearance of periodic variability.

The most notable example of a difference in periodic variability is the first object at the top of Figure 3. The periodogram of the normal-extraction light curve shows no signs of variability, while the optimal-extraction light curve shows a fairly significant peak of around 175 cycles per day, which could easily be interpreted as an astrophysical signal within the observed object. The difference in photometric scatter between the two methods is negligible for this object, so it's unclear why the signal shows up with one method and not the other. There does not seem to be any obvious correlation between the variability and factors such as pixel position or FWHM, and the signal remains when using different comparison stars. So, unless you know what period to expect for your source beforehand, you might want to test different photometry methods if you detect any new periodic signals in your source.

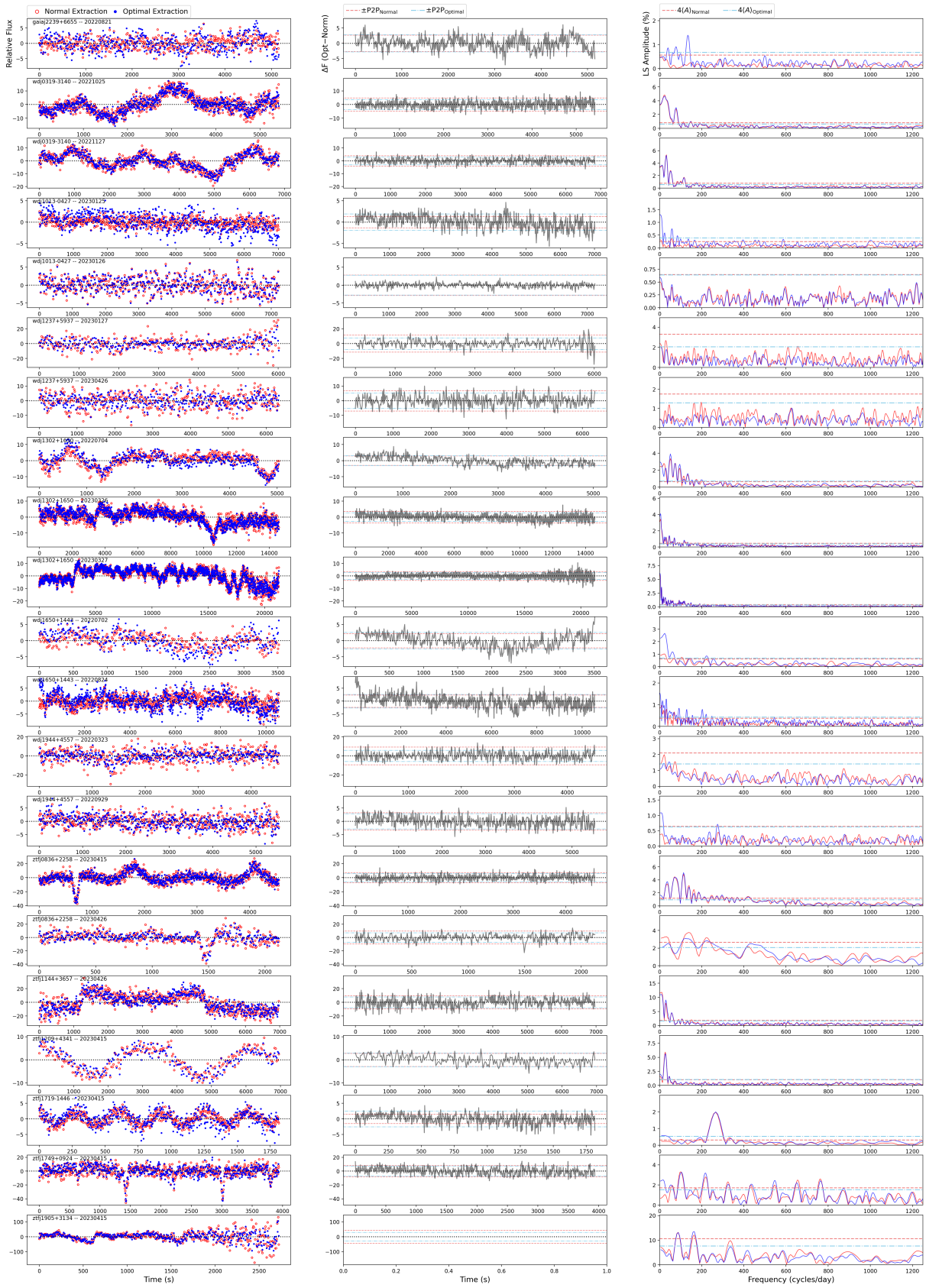

Figure 3: Object by object comparisons of the extracted light curves. (Left) Light curves overplotted. (Middle) Point-by-point flux difference. (Right) Periodograms overplotted.