KSP Photometry Manual

Yuan Qi Ni

April 23, 2018

Abstract

This document describes the current KMTNet Supernova Project (KSP) photometry routines contained in the SuperNova Analysis Package (`SNAP`; `https://github.com/niyuanqi/SNAP`) as well as the standard workflow to extract light curves using them. The current package has routines for standard tasks including image manipulation (cropping, stacking, subtracting), photometry (PSF photometry, aperture photometry, estimating detection limits), and calibration (querying reference stars, differential photometry), as well as routines for transforming light curves between the KMTNet BVI filter system and standard Johnson/SDSS filters. There are also many supernova-specific programs for analysing their light curves and fitting them with models. Included below are detailed descriptions of how individual routines work, tests them on KMTNet data to establish behavioral benchmarks, and instructions on how you can setup or modify these routines/workflows for your purposes.

Contents

1 Introduction **2**

2 How to Setup and Operate **3**
- 2.1 The standard workflow 4
- 2.2 The configuration file 5
- 2.3 The `/Analysis/` suite 6

3 Implementation **7**
- 3.1 Image subtraction 7
  - 3.1.1 Astroscrappy masking 7
  - 3.1.2 HOTPANTS subtraction 8
- 3.2 Photometry 10
  - 3.2.1 PSF fitting 10
  - 3.2.2 Photometry 11
  - 3.2.3 Limiting magnitude 13
- 3.3 Filter corrections 13
  - 3.3.1 $B - V$ color correction 14
  - 3.3.2 S-corrections 15

4 Benchmarks **17**
- 4.1 PSF and Kron aperture extraction 17
- 4.2 Magnitude and noise measurement 19
- 4.3 Image subtraction 21

5 Future Suggestions **22**

References **23**

1 Introduction

The routines described in this document were written with the purpose of analyzing fits images obtained by the KSP with the end goal of creating light curves calibrated to standard filters. This is done by processing the KSP images (e.g., cropping or subtraction) and then performing photometry using the core code `MagCalc.py`. There are user-input options that allow `MagCalc.py` to handle a number of photometry situation, including fixed-position flux extraction for faint sources near the detection limit (or "forced photometry"), near-fixed position source extraction, saturated star PSF photometry, simple aperture photometry for sources that may not have an easily extracted PSF, and photometry of differenced images. There are also standard workflows that can be copied to make the entire process almost automatic, with slight modifications to the configuration file by the user based on situation. Below is a table summarizing the code that is found in the `SNAP` main directory. Many of the codes can be run from command line (as in `python -m SNAP.MagCalc`), and the `-h` flag will give you a basic idea of how to use them.

Core photometry routines	description
`MagCalc.py`	Calculates photometry, calibration, and detection limit.
`Photometry.py`	Contains PSF and aperture photometry functions.
`PSFlib.py`	Library of fittable PSF functions.
`Astrometry.py`	Deals with time and coordinate calculations.
`Catalog.py`	Interprets reference star catalogs (including AAVSO).
`Vizier.py`	Queries online database for reference star catalog.
Core plot/diagnostic routines	description
`MagPlot.py`	Used in MagCalc.py to plot calibration statistics.
`ColorCorr.py`	Used in MagCalc.py to plot filter color dependencies.
Image manipulation routines	description
`CropIm.py`	Crops a fits image into a smaller one (preserves WCS).
`DiffIm.py`	Performs image subtraction using HOTPANTS.
`BinIm.py`	Stacks images using SWARP.
`StampIm.py`	Used to make a series of images into a "stamp collage".
Quality of life files	description
`/Analysis/`	Contains routines for interpreting light curves and analysis.
`/Examples/`	Contains example routines that are often used.
`/cockpit-lc/`	Suite of routines for generating light curves from images.
`/cockpit-sn1a/`	Suite of routines for analysing light curves.
Other photometry routines	description
`ClickMag.py`	Runs MagCalc.py on sources by clicking the image.
`AutoSEx.py`	Runs SExtractor on an image to produce a source catalog.
`MatchPhot.py`	Calibrates SExtractor source catalog with AAVSO stars.

In Section 2, we instruct you on how to setup and use the programs in `SNAP` to make light curves as well as where to look if you want to modify the process. In Section 3, we detail how each routine works and how it is implemented. In Section 4, we use the code on a sample dataset to show the behavior over a wide operating range conforms with expectations, establishing some performance benchmarks including correct measurements of known magnitudes and proper relationship between intensity and noise under PSFs. In Section 5, we give some concluding remarks about where the software may be improved. `SNAP` is under active development, and sometimes codes wont work out of the box because updates are being made to them or they're just very old. If you want to do something in `SNAP` and it's not working, you should just ask Yuan Qi Ni (chris.ni@mail.utoronto.ca) and he will help you get it working.

2 How to Setup and Operate

Making light curves using `SNAP` is a fairly standardized procedure, usually beginning with copying and unpacking a lot of data from the `sn1987a` server. Then, depending on what you need, you can run automatic routines that either crop or subtract images. Finally, there are automatic routines that make light curves. The process of using `SNAP`, therefore, typically consists of setting up a standard file structure and workflow for your object (see Section 2.1), editting the configuration file accordingly (`ObjData.py`; see Section 2.2), and running the codes. For any automatic tasks, we *strongly* suggest testing the code on a few images before fully committing.

2.1 The standard workflow

Normally, we make light curves for each KSP object as follows (see Figure 1 for a diagram):

1. **Setup:** Make a special working directory for your object, and copy the directory `/cockpit-lc/` from the `SNAP` main directory into the working directory. Nominally, this directory contains all the code you need to run to make a light curve (see table below). Edit the setup section of the configuration file `ObjData.py` (see Section 2.2), then run `DataSetup.py`. This will create new directories with raw and reference images as shown in Figure 1.

`/cockpit-lc/` main code	description
`ObjData.py`	This is usually the only thing you need to edit.
`DataSetup.py`	Automatically syncs images from `sn1987a` → `/raw/*.fits`, and corresponding reference images → `/ref/*.fits`.
`CropFits.py`	Auto-crops files: `/raw/*.fits` → `/crop/*.crop.fits`.
`Diffgen.py`	Auto-subtracts files: `/raw/*.fits` → `/diff/*.diff.fits`.
`LCgen.py`	Automatic light curve → `/cockpit-lc/*.txt`.
`LCdgen.py`	Automatic differenced image photometry light curve.
`/cockpit-lc/` other code	**description**
`MakeReg.py`	Makes `*.reg` file for plotting a finder circle in DS9.
`LCmgen.py`	Automatic photometry for multiple objects simultaneously.
`LCbgen.py`	Auto-bins files `/raw/*.fits` → `/bin/*.bin.fits`.
`ContextManager.py`	The glue that holds the standard workflow together.

2. **Image Processing:** Fill out the image processing section of `ObjData.py` according to Section 2.2. Check the reference images in `/ref/` to see if you need to perform image subtraction. If you don't need image subtraction, I suggest running `CropFits.py` to crop your images to a smaller size so that your photometry can run much ($\sim 10\times$) faster. If you do need to peform image subtraction, then you need to run `Diffgen.py`. These codes will create new directories with processed images as shown in Figure 1.

3. **Light Curves:** Here, we use the `MagCalc.py` program to measure magnitudes (or detection limits) from each image. Read Section 3.2.2 to figure out what kind of photometry you want to do and and try `python -m SNAP.MagCalc -h` to figure out what flags you need to use. Fill out the photometry section of `ObjData.py` accordingly, and then run `LCgen.py` if you are using raw/cropped images or `LCdgen.py` if you are using differenced images.

4. **Final Products:** The final output are `*.txt` files for each filter (BVI), containing a row for each image, and columns for time, position, flux, signal-to-noise ratio (S/N), magnitude, error, and detection limit. There are codes in the `/Analysis/` directory for loading them and correcting the BVI filters to standard AAVSO (BVi) ones (see Section 2.3, below).

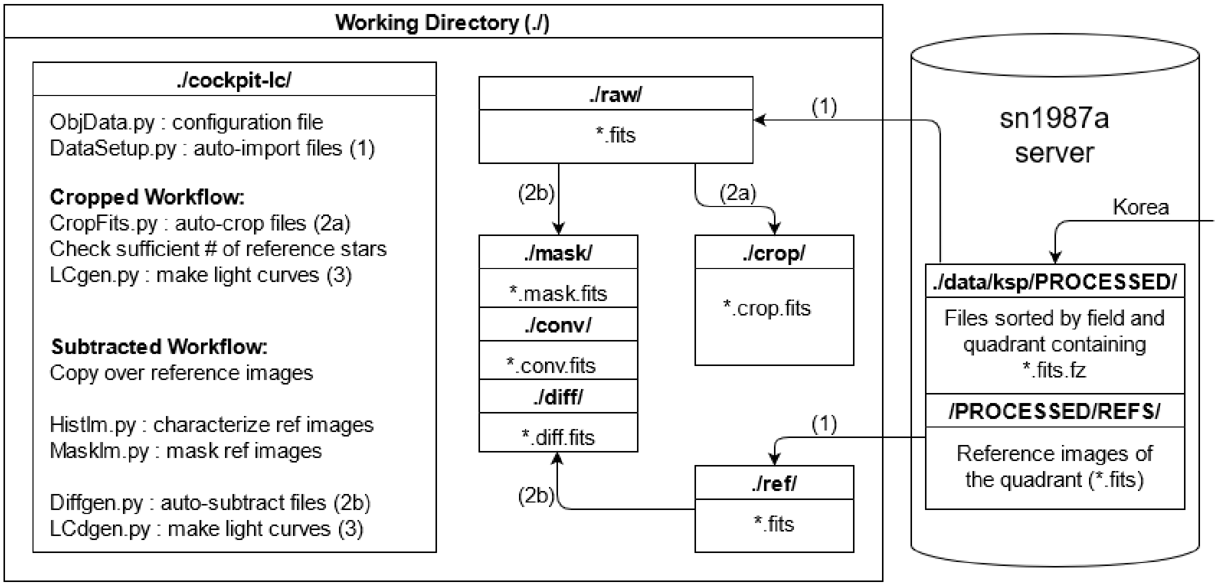


Figure 1: Standard directory structure for making light curves using SNAP. Most of what you need to do should be done in /cockpit-lc/, which you can copy from the SNAP main directory. The main procedure for running codes (in parentheses) is described in Section 2.1.

2.2 The configuration file

Aside from copying standard routines from SNAP, almost all of what you need to edit in order to make light curves for your object is contained in the configuration file `ObjData.py` (everything else is standard). The configuration file is divided into 3 main sections as follows:

1. **Setup section:** Most importantly, you need to fill out the prefix for the `*.fits` files of your object, the year when the object was first detected.

2. **Image processing section:** Here, you need to fill out the RA, DEC position where the source is located and the radius where reference stars should be taken. If you need to perform image subtraction, then you must uncomment the image subtraction variables and fill in the data for your reference images, following the special instructions in Section 4.3.

3. **Photometry section:** These are parameters that control the `MagCalc.py` photometry. Try `python -m SNAP.MagCalc -h` to learn about them. Sometimes, you will also need to modify the `LC*gen.py` codes in the "`Photometry Sequence`" section to do what you want to do. For example, the default instructions in `LCgen.py` perform forced photometry (`psftype = 1`) and if the S/N > `SNRnoise`, then `psftype = 2` will be used to see if higher S/N can be obtained. It's a good idea to note down what photometry sequence and flags you decided to use in every run, and you should fill out the `*.txt` suffix differently each time so that you can tell the final light curve files apart. `nrefs` is the number of reference stars for each of the BVI filters, and we suggest doing some test runs to check how many and which reference stars you are using for your photometry to fill this out. If a reference star is unacceptable, you should definitely be commenting it out of your catalog files.

2.3 The /Analysis/ suite

The `/Analysis/` directory contains codes that can be used to do many things, including loading light curves and correcting BVI filter photometry to standard AAVSO (BVi) filters. Many of these codes are not in a form that can be used without significant modification. Typically, they also cannot be run from command line, so you need to do some digging to understand what functions you want to use and how they work. Some of them have explanations that can be accessed via a python `help(<function>)` call. Below, we provide a table summarizing some important functions that you will probably want to actually use, as well as some explanation for how to use codes from the `/Analysis/` suite to perform filter corrections.

function	description
`LCRoutines.LCload`	Loads multi-band light curves made by `LC*gen.py` (see above).
`LCRoutines.LCcrop`	Crops a time segment from light curves.
`LCRoutines.LClimcut`	Filters out bad images based on a limiting magnitude cutoff.
`LCRoutines.LCcolors`	Interpolates and subtracts adjacent bands.
`LCRoutines.LCcolors`	Interpolates and subtracts adjacent bands.
`LCRoutines.LCbin`	Bins light curve into time intervals.
`SEDAnalysis.SEDinterp`	Interpolates multi-band light curves in time and wavelength either linearly or using a Gaussian process interpolator.
`FilterCorrect.BVcorrectMag`	Applies Park et al. (2017) [1] color correction, converting KMTNet B-band magnitudes to standard AAVSO B band.
`FilterCorrect.BVcorrectLim`	Applies color correction to B band limiting magnitudes.
`SpecAnalysis.Scorr`	Evaluates S-correction between filters based on spectrum.
`SpecAnalysis.Scorr_Vega`	Evaluates S-correction between filters for Vega spectrum.
`FilterCorrect.SBcorrectMag`	Applies S-correction from `SpecAnalysis.Scorr` to B band.
`FilterCorrect.SIcorrectMag`	Applies S-correction from `SpecAnalysis.Scorr` to I band.

Here we will explain how you can simply implement filter correction using `SNAP` routines. The details of filter correction are provided in Section 3.3. First, determine for your case whether simple color corrections from Park et al. (2017) [1] are sufficient, or if you require S-corrections as described in [2] (also see Section 3.3). For both cases, you will need to evaluate the the mean $B - V$ color of the reference stars which you used to perform calibration (`mBVr`) as well as the error of the mean (`mBVrerr`). For the former case, color correction only needs to be applied to the B band, and `FilterCorrect.BVcorrectMag` can be used to apply it, as shown below. You can choose whether you want the interpolation for the color calculation to be linear or to use a Gaussian process depending on how sparse your data is. Limiting magnitudes need to be color-corrected also, and this can be done using `FilterCorrect.BVcorrectLim`. Note that if you change the set of reference stars you are using, then you need to re-measure `mBVr`.

```
ts,mags,errs = BVcorrectMag(ts,mags,errs, 0,1, mBVr,mBVrerr, interp='GP')
ts,lims = BVcorrectLim(ts,lims, 0, mBVr)
```

If your case requires S-corrections, then you need a series of spectra for your source. These can be real observations or spectral templates that you fitted to your object, e.g., using `snpy` for Type Ia SNe (`https://csp.obs.carnegiescience.edu/data/snpy`). Then, you need to read Section 3.3.2 for instructions on how to calculate S-corrections from the spectra. Typically, both B and I bands will require S-corrections, and you can do it using the functions `FilterCorrect.SBcorrectMag` and `FilterCorrect.SIcorrectMag`, respectively, as shown below. You can choose whether you want the S-corrections to be interpolated linearly or with a Gaussian process using `Sinterp`. (Note, you must provide errors for the S-corrections if `Sinterp='GP'`). If you provide the `tdiv` parameter (usually the time of first spectrum) then Park et al. (2017) color corrections will be applied before `tdiv` and S-corrections after, with `interp` specifying the interpolation to be used for the color calculation before `tdiv`.

```
ts,mags,errs = SBcorrectMag(ts,mags,errs, tcorr,scorr, tdiv, interp,
                            0,1 SBVega, mBVr,mBVrerr, Sinterp, scorr_err)
ts,mags,errs = SIcorrectMag(ts,mags,errs, tcorr,scorr, tdiv, interp,
                            0,1 SIVega, Sinterp, scorr_err)
```

3 Implementation

In this section, we describe how our photometry and image subtraction works to help you decide what flags to provide and what variables to set when you run the `SNAP` routines.

3.1 Image subtraction

Image subtraction in `Diffgen.py` is currently done in two steps: (1) cosmic ray and artefact masking using Astroscrappy is done for both the science and template images, and (2) image subtraction is done using HOTPANTS. For each science image, `Diffgen.py` measures PSF and noise characteristics in order to perform Astroscrappy masking and image subtraction optimally. These steps also have to be performed on the template images individually before running `Diffgen.py` using routines in `/Examples/`, and this is described below.

3.1.1 Astroscrappy masking

Astroscrappy (`https://zenodo.org/record/1482019`) masks cosmic rays and artefacts based on the Laplacian edge detection method [3]. The code detects cosmic rays and artefacts by identifying edges that are sharper than point sources ($\sim$ seeing), more significant than background noise, and not a saturated star. The following table lists the input parameters for Astroscrappy. Figure 2 shows how our masking performs on a sample image (E489-1.Q3.B.161121_0725.C). If you want to test Astroscrappy on individual images for fun, you can check out `/Examples/MaskIm.py`. You should also use `MaskIm.py` to mask your template images before running `Diffgen.py`. (For this, keep the template masks in the same folder (`/ref/`) as your templates).

parameter	value	description
`sigclip`	4.0	Laplacian-to-noise limit for cosmic ray detection.
`readnoise`	σ	Background noise level measured from each image.
`satlevel`	`satpix`	This is set in `ObjData.py`. (Usually 40000 for single images.)
`psffwhm`	$2 \times$ seeing	Measured from image with `MagCalc.py` (`diagnosis = True`).
`psfsize`	$5 \times$ seeing	Size of the PSF kernel (note seeing = PSF half-width).

3.1.2 HOTPANTS subtraction

HOTPANTS [4] subtracts images by photometrically matching two images by convolving one of them with a kernel. The matched images are scaled and then subtracted. Our HOTPANTS implementation works by finding the PSFs for both science and template images and then matching them by blurring the sharper of the two images. HOTPANTS uses an analytic kernel composed of a number of Gaussian distributions and we find that the subtraction works better if three of their widths are $\sim 0.5 \times$ seeing, $\sim 1.0 \times$ seeing, and $\sim 2.0 \times$ seeing. The consequence of using an analytic kernel to perform photometric matching is that saturated sources wont convolve in the right way because of saturation artifacts and hence will leave significant residuals, and this is also why sharp artefacts need to be masked out prior to subtraction. The subtraction works better if raw images are divided into discrete regions in which the photometric matching/subtraction is done separately, though this results in some artifacts along the boundaries after subtraction. The table below summarizes how we set our HOTPANTS parameters. Note, before subtraction, you must characterize the saturation, noise level, and fwhm of the template images individually,

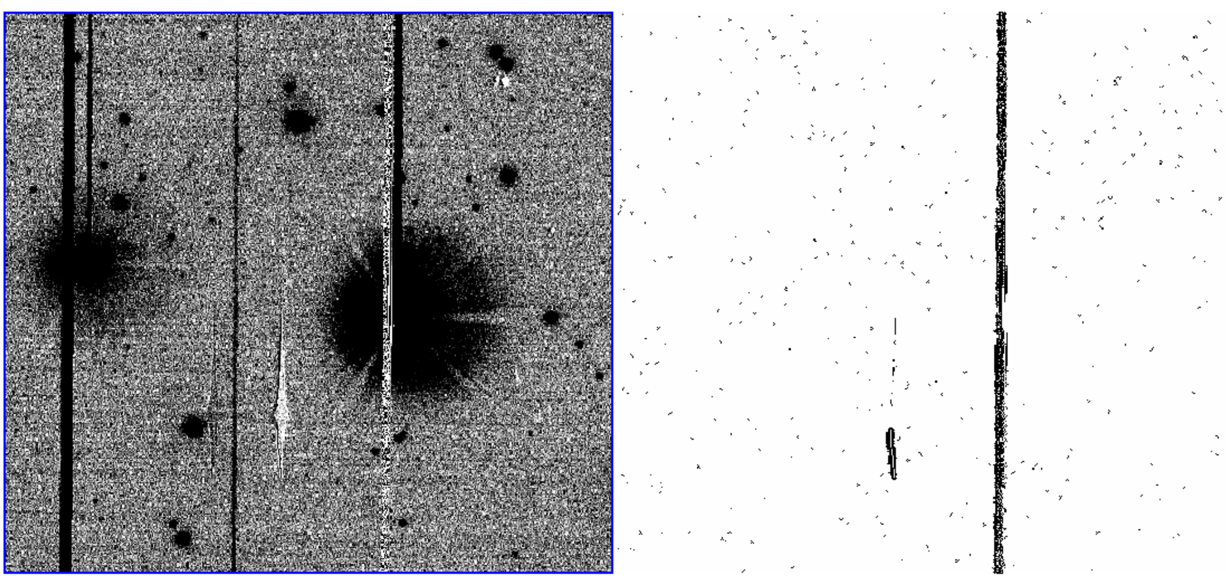


Figure 2: Astroscrappy applied to a science image (left; E489-1.Q3.B.161121_0725.C) to produce the mask (right). Two vertical negative artefacts in the left science image were masked (right; black pixels) as well as many individual pixels consistent with cosmic rays, while saturated stars are not masked (right; white pixels). Subsequent image subtraction will ignore the masked pixels.

setting `reflims` and `ref_fwhm` in `ObjData.py` (see table below).

parameter	value	description
`ng`	3 (6, 4, 2)	Number and degrees of Gaussians to compose kernel.
	(0.5, 1, 2) $\times$ seeing	Gaussian half-widths. Note seeing = larger fwhm of science and template (measure this with `/Examples/HistIm.py`).
`r`	2.5 $\times$ seeing	Half-width of the convolution kernel.
`rss`	6 $\times$ seeing	Half-width of substamp to extract around stars.
`iu`	`satpix`	This is set in `ObjData.py`. (Usually 40000 for single images.)
`il`	`Median` - $10\times\sigma$	From science image (`MagCalc.py`; `diagnosis = True`).
`tu`	`reflims[*][0]`	Usually this is 40000, check using `/Examples/HistIm.py`.
`tl`	`reflims[*][1]`	You need to run `/Examples/MaskIm.py` on the template image to measure `Median` - $10\times\sigma$ (lower limit of valid counts).
`c`	`i/t`	`i` if science image is sharper, `t` otherwise.
`nsx/nsy`	size/(60 $\times$ seeing)	Number of subimages to divide subtraction among.

The result of HOTPANTS subtraction is a differenced science image where stable unsaturated sources present in the template image have been subtracted almost completely and a convolved science image whose PSF matches that of the differenced image. Thus, in photometry, the reference stars are measured from the convolved image, while the source is measured from the differenced image. Figure 3 shows an example of a differenced and a convolved image obtained by applying HOTPANTS to a science image (E489-1.Q3.B.161121_0725.C), where in this case,

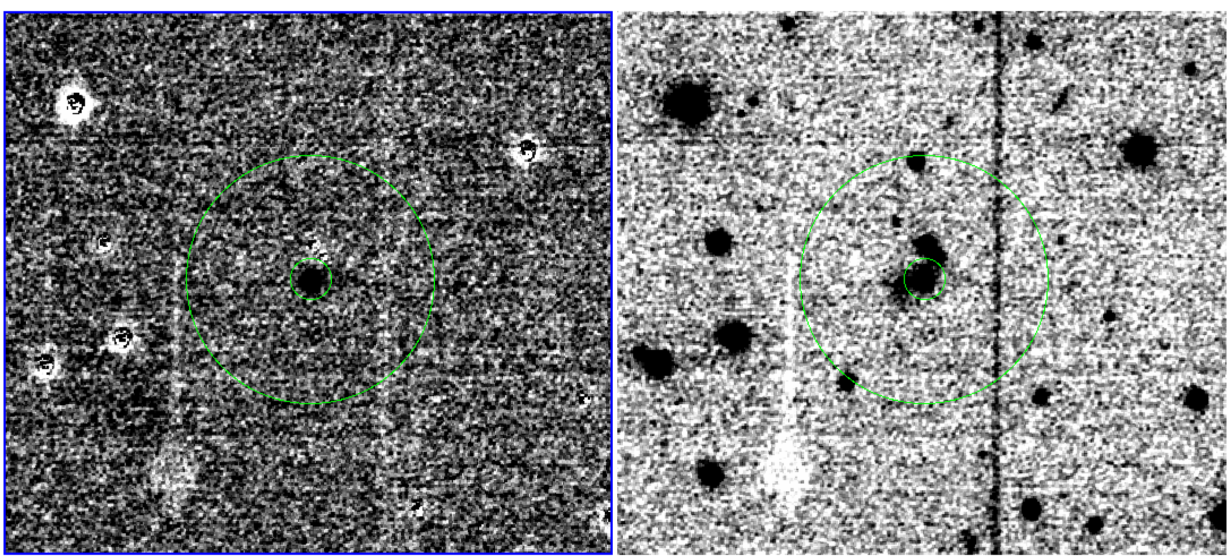


Figure 3: HOTPANTS applied to a science image (E489-1.Q3.B.161121_0725.C), producing a differenced (left) and convolved (right) image. The source in the green circle in the differenced image is a Type Ia SN (KSP-E489-2016ad) on top of a host galaxy. The host galaxy and nearby stars are visible in the convolved image, but they are removed in the differenced image.

the science image was sharper than the template so the science image was blurred. Note that cosmic ray masking was already done before subtraction (Figure 2 is the mask image).

3.2 Photometry

All SNAP photometry is done through the magnitude function in MagCalc.py. It obtains PSF of an image by fitting a Moffat function to a set of reference stars within some radius of the source, and performs photometry in a number of ways (see subsections below). The reference stars can be automatically obtained from AAVSO or read in from a catalog file. An explanation of all of the input flags is provided when you run python -m SNAP.MagCalc -h, but several common use cases can be accessed with flag combinations in the following table.

description	parameters
Fit PSF to source at fixed RA, DEC position.	Set psftype = 1
Fit PSF and position for source near some RA, DEC position.	Set psftype = 2
Fit Moffat function separately for the source.	Set psftype = 3
Fit Sersic profile with index <n> to the source.	Set psftype = s<n>
Perform PSF photometry with <any> psftype.	aperture = None psftype = <any>
Perform aperture photometry with fixed position and aperture radius <r>. (Note, <r> = 0 defines <r> using PSF Kron radius).	aperture = <r> psftype = 1
Perform aperture photometry with fitted central position and aperture radius <r>.	aperture = <r> psftype = 2
Perform aperture photometry with separately fitted Moffat function to define the Kron aperture radius of the source.	aperture = 0 psftype = 3
Fit the background sky simultaneously with a plane (default).	fitsky = 1
Ignore saturated pixels > satpix (default 40000) in PSF fitting.	Set satpix
Automatically query AAVSO reference stars and save to catname (make sure it doesn't exist, or it will simply use the existing one).	Set cat = "AAVSO", also set catname
Perform photometry on HOTPANTS differenced (<diff>) and convolved (<conv>) images.	Set image = <diff>, catimage = <conv>
Measure limiting magnitude at S/N = <n>.	Set limsnr = <n>
Output only image diagnostic info (PSF, noise, median).	diagnosis = True
No plotting and minimal printing.	Set verbosity = 0
Print everything, and no plotting.	Set verbosity = 1
Plot source fitting and statistics of reference stars.	Set verbosity = 2
Plot fitting for reference stars and source.	Set verbosity = 3
Plot reference star and source positions, and fitting for everything.	Set verbosity = 4

3.2.1 PSF fitting

Point spread function (PSF), or image kernel, is the characteristic distribution measured by a detector for a source of photons that is best approximated by a dimensionless point. An ideal image with perfect resolution would render all point sources as dirac delta distributions. However, even in nearly ideal scenarios (e.g., outer space), diffraction limits the resolution of images (diffraction PSF is described by an Airy disk). KMTNet image resolution is limited by atmospheric turbulence, and PSFs in this regime are best approximated by Moffat functions [5].

`MagCalc.py` approximates the PSF using an elliptical moffat distribution characterized by 4 parameters: x, y semi major axes (a_x, a_y), sharpness β, and orientation angle θ. The distribution is described by by equation 1 below (such functions are found in `PSFlib.py`).

$$f(x,y) = \frac{\beta - 1}{\pi a_x a_y}(1 + (\frac{x\cos\theta + y\sin\theta}{a_x})^2 + (\frac{y\cos\theta - x\sin\theta}{a_y})^2)^{-\beta} \tag{1}$$

Thus, any particular point source is approximated by an analytic Moffat function with 7 parameters: central position (x,y) and amplitude, in addition to 4 parameters describing the shape. Fitting is performed in an aperture of size 3M around the source, where M is a given upper bound on the size of full-width half maxima in an image (usually $M = 5$). The fit is usually quite good for most stable reference stars in KMTNet images with χ^2/dof scoring between 0.5 and 2.0, consistently. `MagCalc.py` extracts the PSF of an image (described by 4 shape parameters of the Moffat function) by fitting the 7-parameter Moffat function to reference stars and taking the weighted average of the shape parameters using fit parameter errors. The Scipy `leastsq` function is used for fitting, which performs nonlinear optimization using gradient descent. The fit parameter errors are obtained from the diagonal of the covariance matrix.

`MagCalc.py` can also perform planar background subtraction before fitting point sources. Here planar background is approximated by the 3-parameter function in Equation 2. The planar fit is performed in an annulus around each source which contains background sky. The annulus inner and outer radii will be chosen to be one of 4M-5M, 5M-6M, 6M-7M, 7M-10M, 10M-12M to minimize light pollution from the source itself and any background sources. The annulus whose mean intensity is smallest is chosen. In addition, pixels that are more than 10-σ above or below the planar fit are removed before refining the planar fit again in order to further reduce light from background sources. Standard deviation of the residuals of planar fit to annulus is taken as noise in background, which includes atmospheric turbulence and read noise. The average of the background noise among reference stars can be output using `diagnostic = True` and is the value used in calculating limiting magnitudes (see below).

$$s(x,y) = ax + by + c \tag{2}$$

3.2.2 Photometry

Photometry is done either by integrating an analytic PSF (PSF photometry) or by summing pixel values numerically within an aperture (aperture photometry). For PSF photometry, the global PSF shape (extracted from reference stars, see above) is fitted to the source object and reference stars before the best-fit PSF for each source is integrated analytically to obtain the intensity (see `Photometry.py` for the details). Thus, the accuracy of the measured intensity relies completely on the quality of the PSF fit. The error in measured intensity is estimated by adding contributions from Poisson noise and background noise in Equation 3. Here, size is taken to be the pixel size of the elliptical aperture which contains 90% of source light (Kron aperture) which is analytically known for the Moffat function (see `Photometry.py`).

$$\delta I_{obj} = \sqrt{I_{obj} + (Noise * Size) * *2} \tag{3}$$

The magnitude of the source is calibrated using the reference star catalog magnitudes. The calibrated flux (in Janskies; Jy) of the source is obtained using each i^{th} reference star following Equation 4, where F_0 is the flux zero point of the filter used to take the image in Jy. Equation 5 estimates the flux error. Notice that δI_{obj} is not a random error among calibrations with respect to different reference stars, but shared amongst all and is not included in $\delta F_{ref,i}$ (δI_{obj} will be included after the reference stars are averaged out, as described below).

$$F_{obj,i} = F_0 \frac{I_{obj}}{I_{ref,i}} 10^{-M_{ref,i}/2.512} \tag{4}$$

$$\delta F_{ref,i} = F_{obj,i} \sqrt{(\frac{\delta I_{ref,i}}{I_{ref,i}})^2 + (\frac{\ln(10)}{2.512} \delta M_{ref,i})^2} \tag{5}$$

The final calibrated flux of the source is obtained by taking a weighted mean of the calibrated fluxes with respect to each reference star to get the measured flux of the source F_{obj}. The error of the weighted mean δF_{ref} is a random error term due to the variance among reference stars. The total error in F_{obj} is the sum in quadrature of this random error term due to reference stars and the noise under the source (F_{src}). Here, F_{src} can be calculated by Equation 6 since signal to noise ratio is an invariant quantity. We can then calculate total error in measured flux by Equation 7. F_{obj} and its error in Jy are easily converted to apparent magnitudes.

$$\delta F_{src} = F_{obj} \frac{\delta I_{obj}}{I_{obj}} \tag{6}$$

$$\delta F_{obj} = \sqrt{(\delta F_{src})^2 + (\delta F_{ref})^2} \tag{7}$$

For non-point sources and sometimes for very faint sources, aperture photometry can be a better choice than PSF photometry. For the construction of very early light curves, aperture photometry can provide more robust measurements. It is also more robustly applicable in general, since no matter what happens to a source, it is always possible to manually integrate pixels. However, that means it's measurements are not always the most reliable. When point sources are contaminated by cosmic rays, image subtraction artifacts, or any contamination that can't be eliminated by planar background subtraction, aperture photometry will add all light indiscriminately. PSF photometry, on the other hand, is more reliable against disturbances in PSF shape, since shape is constrained to be a moffat function.

Aperture photometry is set by the choice of aperture radius, and the same circular aperture is usually applied globally across the image to the source and to all reference stars. After planar background subtraction, intensity is measured by direct summation of pixels in aperture and calibrated in the same way as described in Section 3.2.1. The aperture may be manually selected by the user, or may be set by image PSF. The latter extracts PSFs from standard reference stars as in Section 3.2.1, and uses the analytic PSF shape to determine a Kron aperture. This is an elliptical aperture containing 90% of source light, and is known analytically for the moffat function (see `Photometry.py`). The average of semi-major axes for this Kron aperture defines the Kron radius, which is then applied to integrate intensities of the source and reference stars.

3.2.3 Limiting magnitude

The limiting magnitude is the magnitude of the dimmest star detectable at some image location. Here, detectable means above the signal to noise (S/N) threshold required for detection at a given confidence, typically S/N $= 2$ or 3. We find limiting magnitude by evaluating magnitudes of stars synthesized using the measured image PSF, where we are allowed to vary the height alone. We generate 10 sources of various brightnesses and we evaluate their intensities analytically. We use the measured noise at the image location (see PSF fitting subsection above) to evaluate what the S/N would be for each synthetic source. Taking the two synthetic sources that are closest to the detection threshold (above and below), we refine the brighness binning and evaluate again. If all sources are above or below the requires signal to noise ratio, we relax the upper or lower bound of the binning by an order of magnitude. In this way, we are able to converge to the given S/N within a few iterations, providing a synthetic source whose intensity defines the detection limit. Its magnitude is calibrated as if it were a real source (see photometry subsection above).

3.3 Filter corrections

Sources in KMTNet images are calibrated to their analogous filters in AAVSO, Johnson BV and Sloan i. The calibration given in Equation 4 can be written as follows in magnitudes, where $m_{B,src}$ is the standard magnitude of the source, $n_{B,src}$ is the instrumental magnitude of the source, $m_{B,ref}$ are standard magnitudes of individual reference stars, I are intensities, and Z is the B-band photometric zero point obtained by averaging over all reference stars.

$$m_{B,src} - m_{B,ref} \simeq -2.5 \log \left(I_{B,src}/I_{B,ref} \right) \tag{8}$$

$$m_{B,src} \simeq -2.5 \log I_{B,src} + \left(m_{B,ref} + 2.5 \log I_{B,ref} \right) \tag{9}$$

$$m_{B,src} - n_{B,src} \simeq \left(m_{B,ref} + 2.5 \log I_{B,ref} \right) \tag{10}$$

$$m_{B,src} - n_{B,src} = \langle m_{B,ref} + 2.5 \log I_{B,ref} \rangle_{ref} = Z \tag{11}$$

Thus, calibration is the result of applying the following transformation equations.

$$m_{B,src} = n_{B,src} + Z_B \tag{12}$$

$$m_{V,src} = n_{V,src} + Z_V \tag{13}$$

$$m_{i,src} = n_{i,src} + Z_i \tag{14}$$

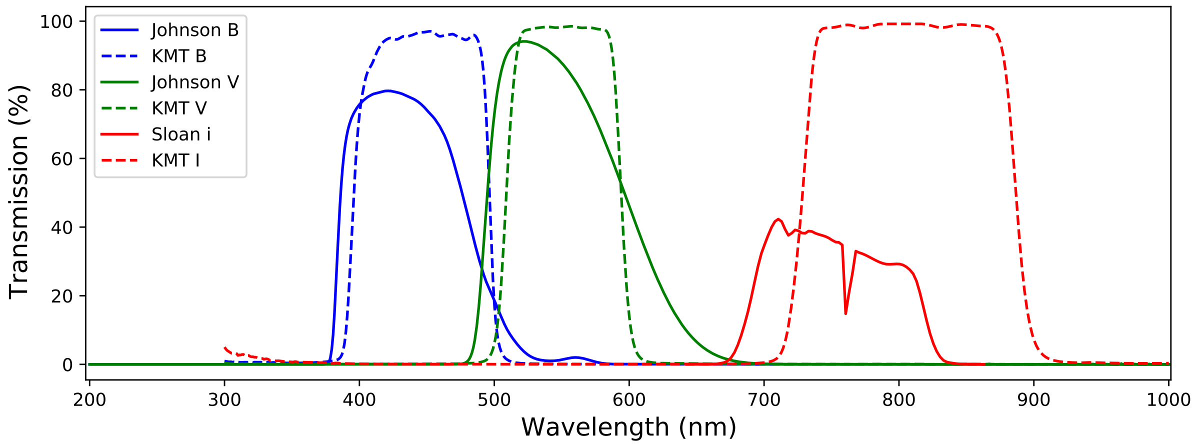


Figure 4: Comparison of KMTNet BVI filter transmission functions to standard AAVSO filters (Johnson BV and Sloan i). The most significant filter difference is actually in the B-band because stellar SEDs tend to be steeper there. This causes our calibrated B-band magnitudes of them to be systematically different from the AAVSO value, dependent on $B-V$ color (Park et al. 2017 [1]). For SNe near-/post-peak, when broad spectral features develop in the B-band, the magnitude offset depends on the SN spectrum (Stritzinger et al. 2002 [2]).

However, as shown in Figure 4, the KMTNet filter functions are not exact matches for the AAVSO filters. (Below, we refer to the KMTNet filters as $B'V'i'$). The discrepancy is most significant for the B band since most stars have steep spectral energy distributions (SEDs) in that band, while the Vi-band SEDs are relatively flat. Park et al. (2017) [1] showed that the difference between KMTNet instrumental B'-band magnitudes and AAVSO B-band magnitudes for standard reference stars is not a constant, but systematically dependent on $B-V$ color with slope $c = 0.27$. We describe how to correct for this systematic in two cases below.

3.3.1 $B-V$ color correction

In the case where the source spectrum is expected to be similar to standard stars, we can apply the following transformations with $B-V$ color correction from Park et al. (2017).

$$m_{B,src} = n_{B',src} + c\,(m_{B,src} - m_{V,src}) + Z_B \tag{15}$$

$$m_{V,src} = n_{V',src} + Z_V \tag{16}$$

$$m_{i,src} = n_{i',src} + Z_i \tag{17}$$

Below, we derive the method by which we can convert our old calibrated magnitudes (which we will call m'_B) to the standard color-corrected magnitudes (m_B). First, we derive the new equation for calibrating source magnitude with respect to reference stars as follows.

$$(m_{B,src} - m_{B,ref}) - (n_{B',src} - n_{B',ref}) \simeq c\,(m_{B,src} - m_{V,src}) - c\,(m_{B,ref} - m_{V,ref}) \tag{18}$$

$$m_{B,src} - c\,(m_{B,src} - m_{V,src}) \simeq m_{B,ref} - 2.5\log\frac{I_{B',src}}{I_{B',ref}} - c\,(m_{B,ref} - m_{V,ref}) \tag{19}$$

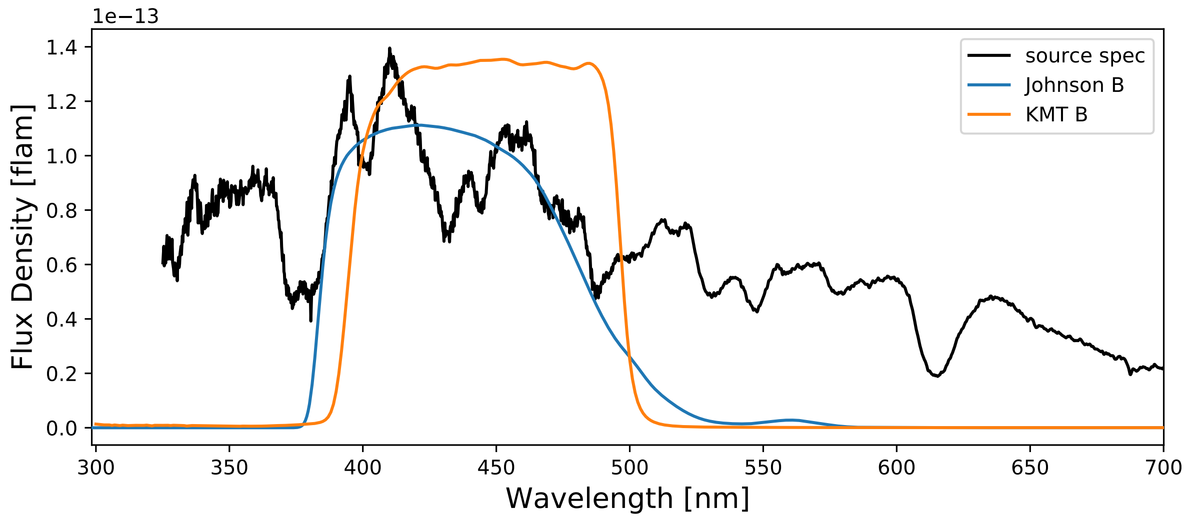


Figure 5: Comparison of the KMTNet B and Johnson B filter transmission functions to a Type Ia SN (KSP-SN-2018ku) spectrum near-peak. Note the broad spectral features that can significantly affect the measured magnitudes in the B band (similar feature can also affect the I band).

The new calibration equation is obtained by averaging over all reference stars.

$$m_{B,src} - c\,(m_{B,src} - m_{V,src}) = \langle m_{B,ref} - 2.5\log\frac{I_{B',src}}{I_{B',ref}}\rangle_{ref} - c\,\langle m_{B,ref} - m_{V,ref}\rangle_{ref} \tag{20}$$

$$m_{B,src} - c\,(m_{B,src} - m_{V',src}) = m_{B',src} - c\,\langle m_{B,ref} - m_{V',ref}\rangle_{ref} \tag{21}$$

In the last step, we noticed that the first angle-bracketed term is how we originally calibrated our source magnitudes and $m_{V'} = m_V$ is unchanged. Thus, Equation 22 corrects B band.

$$m_{B,src} = \frac{1}{1-c}(m_{B',src} - c\,m_{V',src} - c\,\langle m_{B,ref} - m_{V,ref}\rangle_{ref}) \tag{22}$$

3.3.2 S-corrections

SNe near- and post-peak can develop broad spectral features as shown in Figure 5, which can cause significant differences in photometry when the filter functions are sufficiently different. Thus, it would be inaccurate to apply the Park et al. (2017) color correction, which was derived from standard stars with no such spectral features. Moreover, both the B' and i' (KMTNet B and I) bands are likely to be affected give the relatively large differences between them and the standard B and i filters, whereas the color correction only affected the B band. For SNe near-/post-peak, we need to perform Spectroscopic- (S-)corrections as described in Stritzinger et al. (2002) [2]. Below are new transformation equations, where we adopt an S-correction term

($S_{B,src}$) for the source, and a color-correction term for reference stars.

$$m_{B,src} = n_{B',src} - 2.5\log\frac{I_{B,src}}{I_{B',src}} + Z_{SB} \tag{23}$$

$$m_{B,src} = n_{B',src} + S_{B,src} + Z_{SB} \tag{24}$$

$$m_{B,ref} = n_{B',ref} + c\,(m_{B,ref} - m_{V,ref}) + Z_B \tag{25}$$

We can derive the new zero point using Vega, for which $m_{B,0} = m_{V,0} = 0$.

$$0 = n_{B',Vega} - 2.5\log\frac{I_{B,Vega}}{I_{B',Vega}} + Z_{SB} \tag{26}$$

$$0 = n_{B',Vega} + Z_B \tag{27}$$

$$Z_{SB} = Z_B + 2.5\log\frac{I_{B,Vega}}{I_{B',Vega}} \tag{28}$$

$$Z_{SB} = Z_B - S_{B,Vega} \tag{29}$$

Then the transformation equations can be re-written with the same zero point.

$$m_{B,src} = n_{B',src} + S_{B,src} - S_{B,Vega} + Z_B \tag{30}$$

$$m_{B,ref} = n_{B',ref} + c\,(m_{B,ref} - m_{V,ref}) + Z_B \tag{31}$$

The magnitude of the source can be calibrated with respect to reference stars as follows.

$$m_{B,src} = \langle m_{B,ref} - 2.5\log\frac{I_{B',src}}{I_{B',ref}}\rangle_{ref} + S_{B,src} - S_{B,Vega} - c\,\langle m_{B,ref} - m_{V,ref}\rangle_{ref} \tag{32}$$

Again, we recognize the first angle-bracketed term as how we originally calibrated our source magnitudes ($m_{B',src}$), and we obtain Equation 33 which corrects the B band.

$$m_{B,src} = m_{B',src} + S_{B,src} - S_{B,Vega} - c\,\langle m_{B,ref} - m_{V,ref}\rangle_{ref} \tag{33}$$

We can obtain $S_{B,src}$ and $S_{B,Vega}$ by performing synthetic photometry (e.g., using `synphot`; `https://ascl.net/1811.001`) on the source and Vega's spectrum using the two filters and the same zero point to obtain $m_{B',src}$ and $m_{B,src}$, $m_{B',Vega}$ and $m_{B,Vega}$. The S-corrections are simply the difference of the magnitudes obtained using the two filters.

$$S_{B,Vega} = m_{B,Vega} - m_{B',Vega} \tag{34}$$

$$S_{B,src} = m_{B,src} - m_{B',src} \tag{35}$$

We can S-correct the i-band magnitudes with a similar equation (Equation 36), except that the i-band requires no color correction term as shown in Park et al. (2017) [1].

$$m_{i,src} = m_{i',src} + S_{i,src} - S_{i,Vega} \tag{36}$$

S-corrections were calculated for a Type Ia SN, KSP-SN-2018ku, using a series of spectra obtained near- and post-peak, shown in Figure 6. As seen in the figure, the S-correction is most significant most near-/post-peak, but goes to zero for early pre-peak phases.

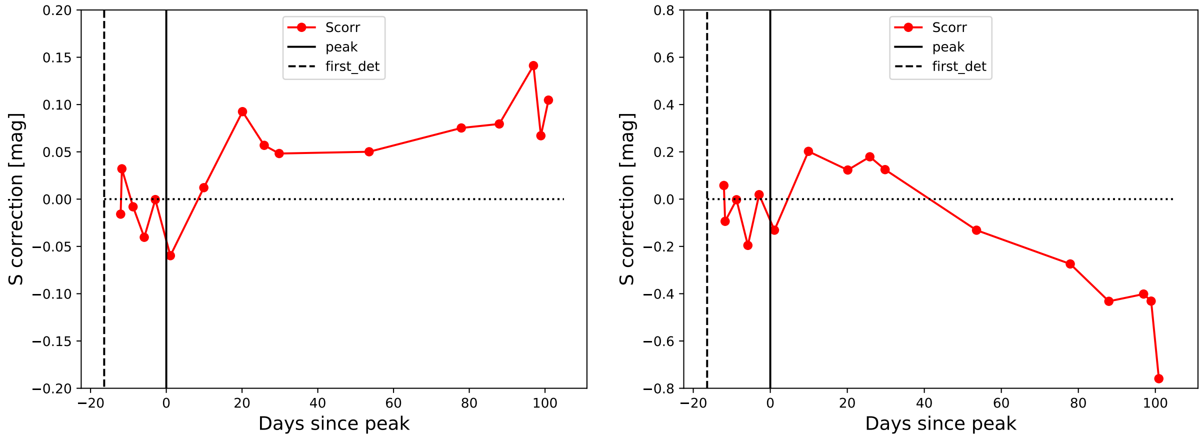


Figure 6: Calculated S-corrections for a Type Ia SN (KSP-SN-2018ku) in KMTNet B (left) and I filters. Note that both S-corrections increase significantly post-peak due to broad spectral features that develop over the SN evolution. For early epochs pre-peak, the S-correction goes to zero in both bands, consistent with the spectrum being dominated by continuum emission.

4 Benchmarks

We use the N5128-1.Q1.SN dataset below to establish the following benchmarks.

1. The extracted PSF is a good approximations to bright point sources in the image. The Kron aperture radius nearly optimizes S/N in aperture photometry.

2. The measured magnitudes of reference stars matches those given in the catalog, and those measured using PSF and aperture photometry match each other. Measured noise and intensities also have the proper scaling relation in each method.

3. Image subtraction removes stable non-saturated stars. For a new source, measured magnitudes pre-/post-subtraction match if there is no underlying source (e.g., host galaxy).

4.1 PSF and Kron aperture extraction

To test PSF extraction, we tested the PSF fitting with `psftype = 3` and `fitsky = 1` on 1409 unsaturated AAVSO reference stars with magnitudes from 14.0 to 17.5 mags in a typical image with limiting magnitude 21.9 mag at S/N = 2. Figure 7 shows the extracted PSF for a typical reference star in that range. To measure the fit quality, we apply the χ^2/dof parameter, where $\chi^2/dof \sim 1$ is a good fit, $\chi^2/dof >> 1$ means underfit, and $\chi^2/dof << 1$ means overfit. Figure 8 is a histogram of χ^2/dof for all 1409 reference stars. As seen in the figure, the median χ^2/dof is ~ 1. Hence a 7-parameter moffat function appears to fit reference stars quite well.

Kron aperture photometry tries to solve an optimization problem for the radius that can extract the maximum signal to noise from a source. Selecting too small of an aperture will omit valuable source light at the PSF wings and hence decrease S/N. Selecting too large an aperture

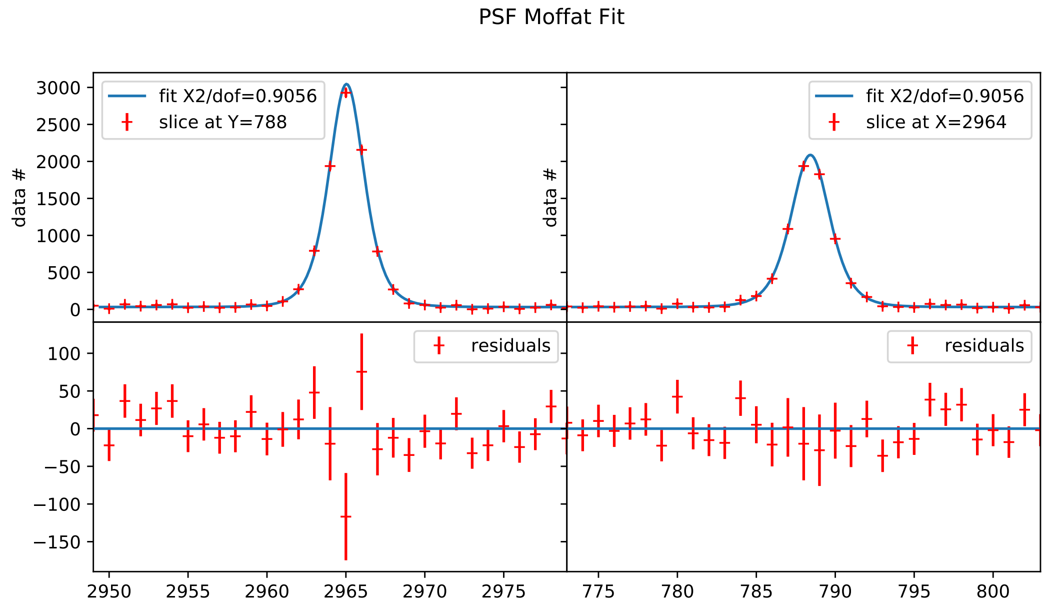


Figure 7: (Top) PSF of a 17.2 mag reference star in B band fitted using `MagCalc.py` with `psftype = 3` and `fitsky = 1`). The fit χ^2/dof is ~ 1, indicating an excellent fit. (Bottom) The residuals of the fit are within ~ 1 errorbar of zero, confirming the goodness of the fit. Note that plots like these can be readily generated when using `MagCalc.py` by choosing `verbosity > 2`.

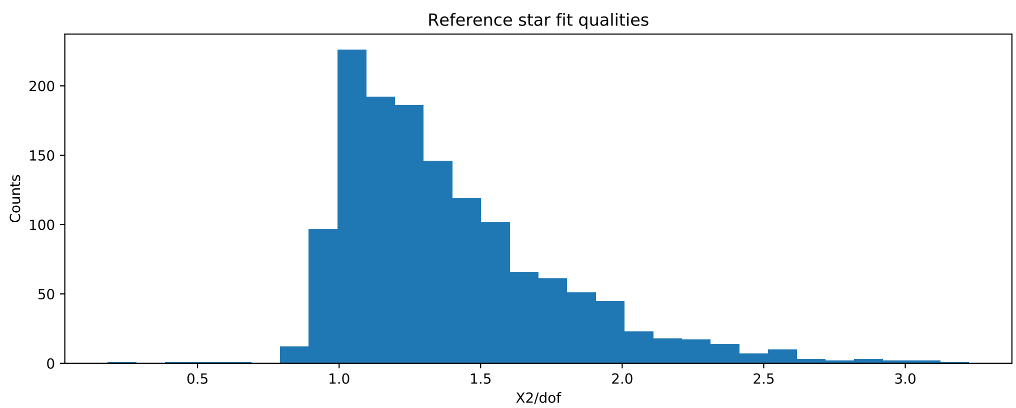


Figure 8: Histogram of χ^2/dof for 1409 reference stars from 14.0 to 17.5 mags in an image with limiting magnitude 21.9 mag at S/N = 2. All of them are clustered around $\chi^2/dof = 1$. Note that `MagCalc.py` automatically produces this plot if you choose `verbosity > 2`.

will incorporate a lot of pixels in which signal is weaker than noise and hence also decrease S/N. The optimal aperture incorporates source light up to where the S/N tradeoff of adding an extra pixel is no longer justifiable. We show that the Kron aperture [6], containing 90% of source light for a point source, is often a good choice by performing aperture photometry on a typical reference star (same as the one shown in Figure 7) using a range of aperture radii between 1–9

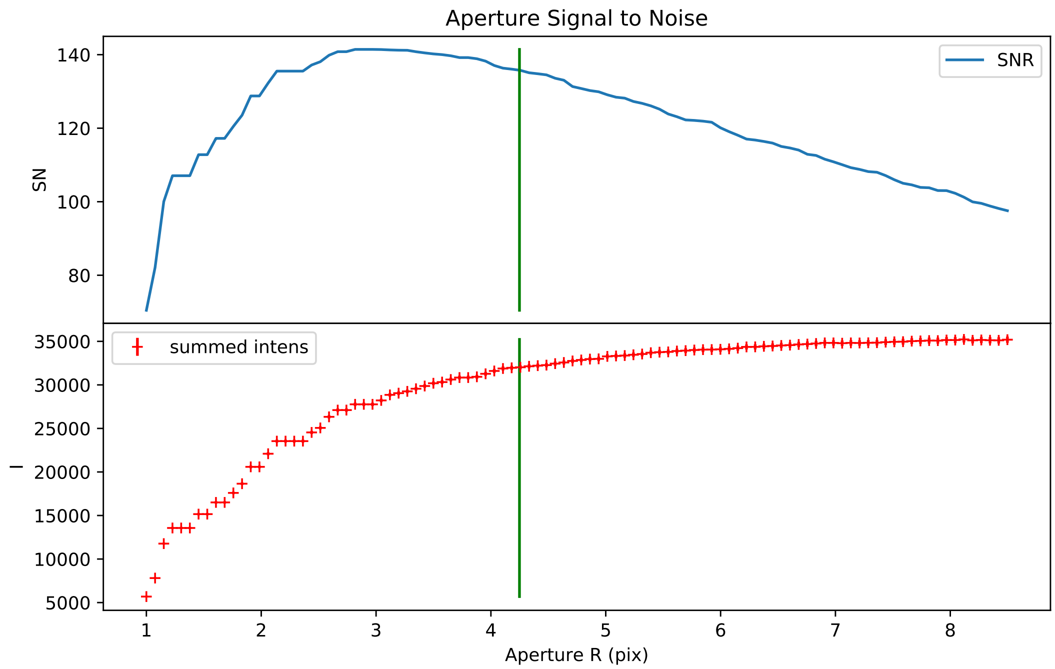


Figure 9: (Top) S/N extracted for a 17.2 mag reference star in B band using aperture photometry for a range of aperture radii. (Bottom) Intensity extracted by aperture photometry for the same aperture radii. The green vertical line marks the Kron aperture radius, which captures 90% of source light and nearly optimizes S/N. Note that when performing aperture photometry with `MagCalc.py`, plots like these will be generated if `verbosity > 2` is selected.

pixels. Figure 9 shows the measured intensity as well as S/N as a function of aperture radius. The vertical green line shows the Kron aperture radius. Apparently, the Kron radius does incorporate $\sim 90\%$ of total intensity and almost optimizes the S/N of the source.

4.2 Magnitude and noise measurement

In order to test the magnitude measurements, we tested `MagCalc.py` on 1409 unsaturated AAVSO reference stars between 14.0 and 17.5 mags (the same ones as in Figure 8). Figure 10 compares the instrumental magnitudes (i.e., log intensity) of the reference stars measured by `MagCalc.py` using aperture (left) and PSF (right) photometry with their known magnitudes listed in the AAVSO catalog (x-axis). The blue solid lines are the best-fit photometric solutions. Fit bias is low, but variance is quite high, i.e., errorbars seem underestimated compared to the variance. (Note, the variance is reflected in the δF_{ref} error term during calibration, see Section 3.2.2 above). Figure 11 compares the noise measured for each reference star to their instrumental magnitude ($\sim$ log intensity), showing the "noise-intensity" relation for both aperture (left) and PSF (right) photometry. The expected noise-intensity relation is different in the high intensity regime, where noise is dominated by Poisson noise, and the low intensity regime, where noise is background

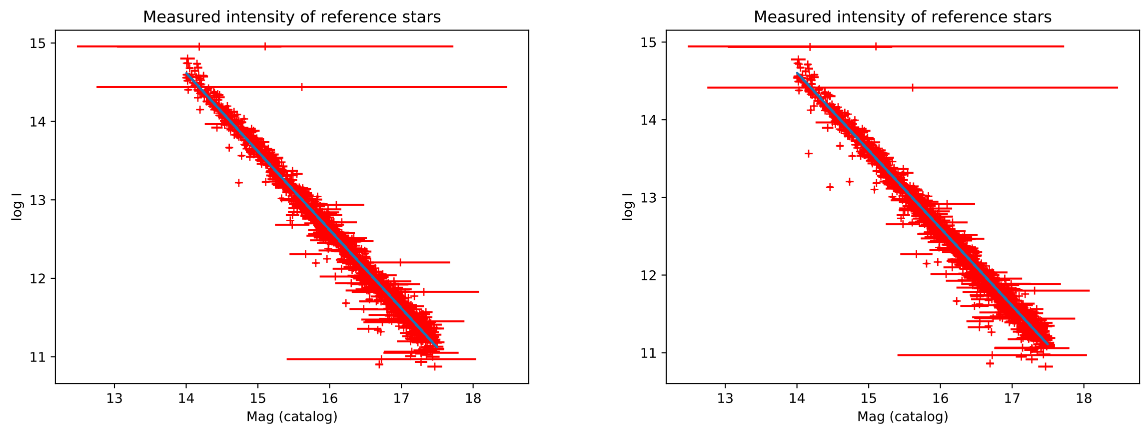


Figure 10: Comparison of calculated instrumental magnitudes (y-axes) to catalog magnitudes (x-axes) for 1409 reference stars from 14.0 to 17.5 mags measured using aperture photometry (left) and PSF photometry (right) in an image with limiting magnitude 21.9 mag at S/N = 2. There is a clear linear relation (blue solid lines) that can be extracted between instrumental and catalog (or calibrated) magnitudes, representing the photometric solution. Note that these plots are generated when running `MagCalc.py` photometry with `verbosity > 2`.

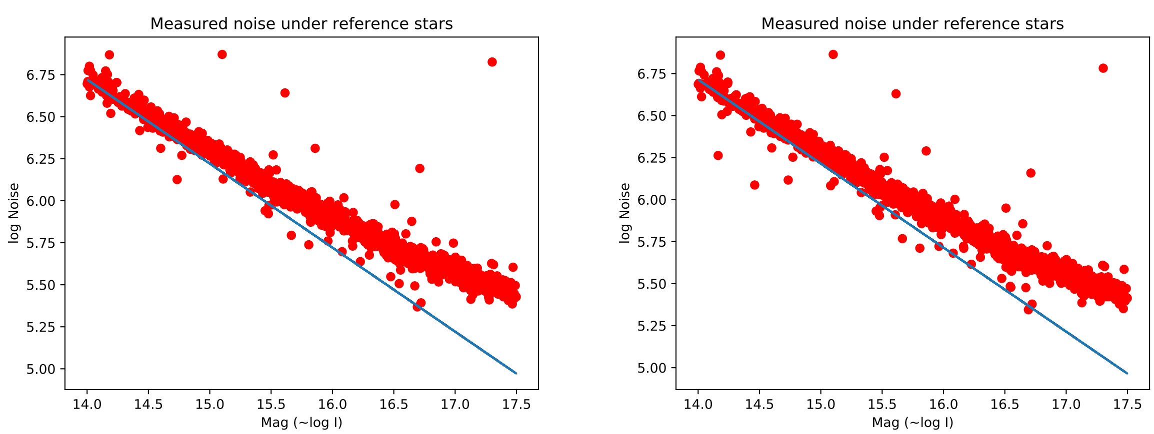


Figure 11: Comparison of measured noise (y-axes) to measured intensity (x-axes) for 1409 reference stars from 14.0 to 17.5 mags measured using aperture photometry (left) and PSF photometry (right) in an image with limiting magnitude 21.9 mag at S/N = 2. The blue solid line has slope of $-1/2$. For bright (low magnitude) stars, noise appears to scale with the square root of intensity, as expected in the Poissonian regime, whereas for dim (high magnitude) stars, noise flattens out, as expected in the background-dominated regime. Note that `MagCalc.py` can be made to generate plots like these by selecting `verbosity > 2`.

dominated. Slope $-1/2$ (blue solid line) is the expectation for the high-intensity regime. As seen in the figure, both aperture and PSF photometry recover the correct noise-intensity relation: the brighter-magnitude reference stars largely follow the blue solid line, while the noise flattens out for dimmer reference stars, consistent with background domination.

4.3 Image subtraction

To test image subtraction, we used a coadded reference image and 26 science images taken approximately one year later of the same sky. The science images contain a bright unsaturated transient source between ~ 19.5–20.5 mag, not present to deep limits in the reference image ($\gtrsim 23$ mag). First, we subtracted reference image from itself, expecting the residuals to contain nothing at all, and the entire differenced image is displayed in Figure 12 (left). The background noise level (standard deviation) in the reference image is ~ 4.0 while the noise in the image appears to be much less than that. Thus, the subtraction is almost complete, though some artefacts may be seen along the borders where the image has been divided into 4 subimages before subtraction. Second, we subtracted the reference image from a science image, with the entire differenced image shown in Figure 12 (right). Subtraction of saturated stars invariably leaves bright residuals, but unsaturated stable sources leave almost none.

We test the differened-image photometry routines by first measuring the magnitude of the transient source in the original un-subtracted science images. Then, after subtraction of the reference image from each science image, we apply the differenced-image photometry on differenced

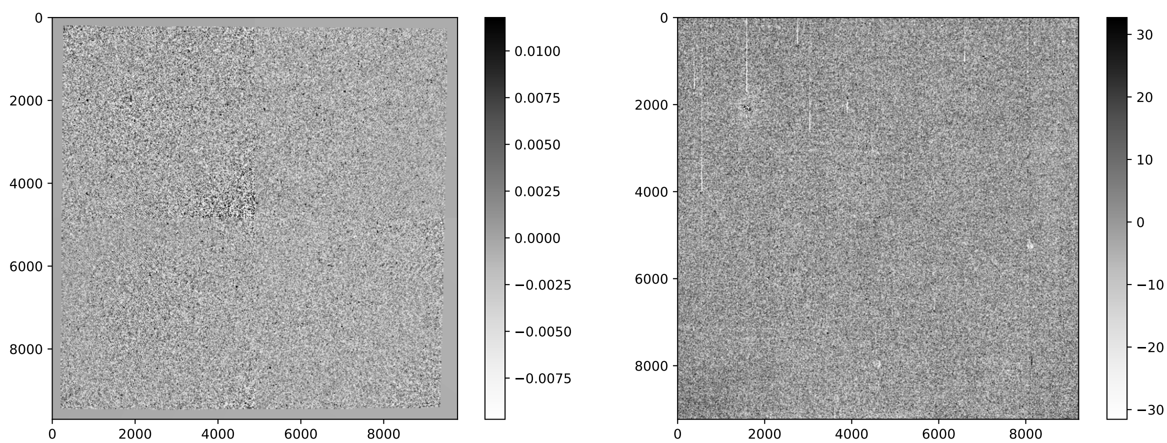
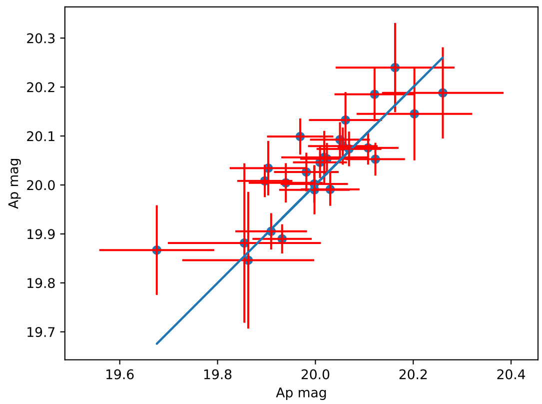


Figure 12: (Left) Reference image subtracted from itself using HOTPANTS. The background noise level in the original reference image is ∼4.0 data number, while the colorbar indicates that the noise level in the differenced image is much less than that. Looking at the vertical and horizontal bisecting lines, we can see slightly higher noise artefacts where the images were divided prior to subtraction. (Right) Differenced image with reference image subtracted from a science image taken of the same sky one year later. Negative vertical streaks caused by saturated stars are the most conspicuous artefacts that can be seen throughout the image.



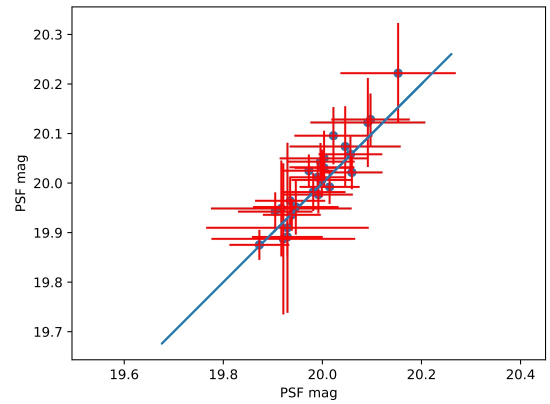


Figure 13: Comparison of calculated magnitudes of a single bright unsaturated transient source between ~ 19.5–20.5 mag in several science images before subtraction (y-axes) and in differenced and convolved images after HOTPANTS image subtraction (x-axes) using aperture (left) and PSF (right) photometry. Since the images were uncrowded, with no underlying host galaxy, this test isolates the ability of differenced image photometry to recover the magnitude of the source.

and convolved images. In Figure 13, we compare the measured magnitude of the transient source from the differenced and convolved images (x-axes) with those from the original image (y-axes) for both aperture (left) and PSF (right) photometry. The blue solid line shows y = x, where we expect the brightness to match before and after subtraction. As seen in the figure, magnitudes measured before and after subtraction apparently agree within error. PSF photometry performs slightly better than aperture photometry, with slightly less scatter about the blue solid line, probably due to its ability to isolate the PSF shape from subtraction artifacts.

5 Future Suggestions

The current iteration of `SNAP` performs a variety of tasks required for KSP photometry within tolerable benchmarks. However, several sugggestions come to mind for future additions to the code that can significantly improve KSP photometry.

1. Implementing some kind of bayesian parameter estimation software like Emcee [7] could make PSF fitting more robust and have more meaningful error estimation.

2. Currently aperture photometry performs simple summation of background subtracted apertures. Adding a step to clean hot pixels or cosmic rays would be much more robust.

3. Image subtraction with HOTPANTS is slightly outdated. Zackay et al.'s optimal image subtraction algorithm [8] seems to be favours by some transient pipelines. In the future, we can implement this or borrow an existing implementation.

4. Many routines are currently single threaded, though many parts such as PSF extraction can easily be made multi-threaded. The routines are intended to be used for constructing light curves, in which case it would make more sense to multi-thread the processing of each individual image (we can try to implement this in the future).

References

[1] H. S. Park, D.-S. Moon, D. Zaritsky, M. Pak, J.-J. Lee, S. C. Kim, D.-J. Kim, and S.-M. Cha, "Dwarf Galaxy Discoveries from the KMTNet Supernova Program. I. The NGC 2784 Galaxy Group," **848** no. 1, (Oct., 2017) 19, `arXiv:1708.07326 [astro-ph.GA]`.

[2] M. Stritzinger, M. Hamuy, N. B. Suntzeff, R. C. Smith, M. M. Phillips, J. Maza, L. G. Strolger, R. Antezana, L. González, M. Wischnjewsky, P. Candia, J. Espinoza, D. González, C. Stubbs, A. C. Becker, E. P. Rubenstein, and G. Galaz, "Optical Photometry of the Type Ia Supernova 1999ee and the Type Ib/c Supernova 1999ex in IC 5179," **124** no. 4, (Oct., 2002) 2100–2117, `arXiv:astro-ph/0206438 [astro-ph]`.

[3] P. G. van Dokkum, "Cosmic-Ray Rejection by Laplacian Edge Detection," **113** no. 789, (Nov., 2001) 1420–1427, `arXiv:astro-ph/0108003 [astro-ph]`.

[4] A. Becker, "HOTPANTS: High Order Transform of PSF ANd Template Subtraction." Astrophysics source code library, Apr., 2015.

[5] A. F. J. Moffat, "A Theoretical Investigation of Focal Stellar Images in the Photographic Emulsion and Application to Photographic Photometry," **3** (Dec., 1969) 455.

[6] R. G. Kron, "Photometry of a complete sample of faint galaxies.," **43** (June, 1980) 305–325.

[7] D. Foreman-Mackey, D. W. Hogg, D. Lang, and J. Goodman, "emcee: The MCMC Hammer," **125** (Mar., 2013) 306, `arXiv:1202.3665 [astro-ph.IM]`.

[8] B. Zackay, E. O. Ofek, and A. Gal-Yam, "Proper Image SubtractionOptimal Transient Detection, Photometry, and Hypothesis Testing," **830** (Oct., 2016) 27, `arXiv:1601.02655 [astro-ph.IM]`.