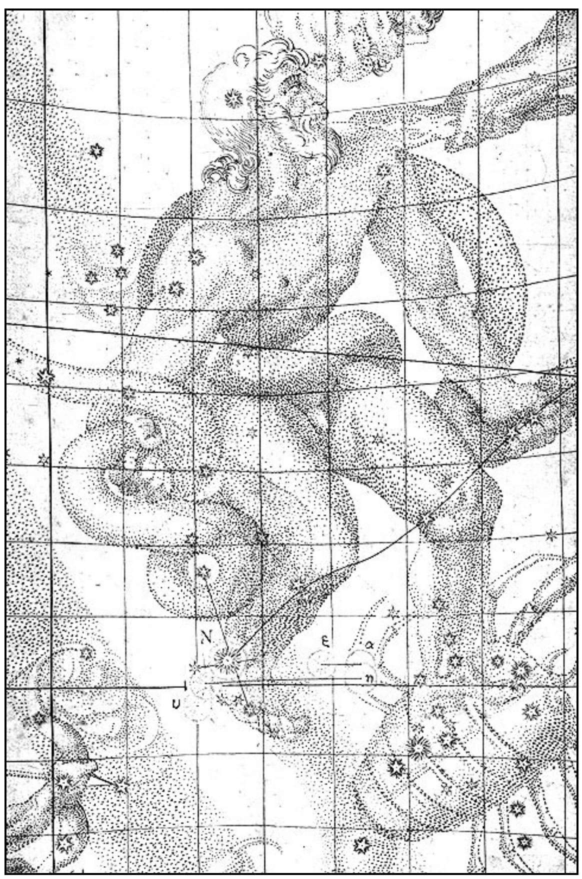
PTFIDE: Palomar Transient Factory Image Differencing and Extraction

Algorithms and Module Usage

Version 4.5, 02/14/2013

Frank Masci, IPAC - Caltech



DE STELLA NOVA IN PEDE SERPENTARII, KEPLER (1604)

Revision History

Date	Version	Author	Description
May 27, 2012	1.0	Frank Masci	Initial test/dev version
June 8, 2012	2.0	Frank Masci	Dev version: resamples sci frame into ref image system; position refinement step uses non-optimal resampling function from PDL::Transform
July 9, 2012	3.0	Frank Masci	Many updates: switched to resampling ref image into sci image system; more robust position and gain refinement, PSF-matching and image differencing.
September 4, 2012	4.0	Frank Masci	**First alpha-test version:** Read science-image masks; more robust saturation estimation; extract transient & variable candidates from diff images and perform both PSF-fit and aperture photometry using *DAOPhot*; generate QA metrics on difference images.
September 14, 2012	4.1	Frank Masci	* Further tuned *hotpants* parameters using a global optimization method; * Updated constraints on kernel model Gaussian widths following dynamic rescaling; * Don't perform source extraction if > 85% of diff-image pixels are masked; * Moved pre PSF-matched diagnostic diff-image out of debug mode; * Added new switch "*–wmode*" to use pixel modes (instead of medians) for differential background estimation; * Perform more robust outlier-trimming for differential background estimation
October 9, 2012	4.2	Frank Masci	**First beta-test version:** * Implemented new PSF-

			matching algorithm (Pixelated Convolution Kernel – PiCK); * Optimized some parameters; * Added OBSMJD to headers of catalog tables; * Speed up of automatic PSF-creation by limiting 'faint neighbors' to subtract in 2nd iteration; * Expand saturated pixels in resampled ref-image mask for more complete blanketing * Account for exposure-time if ZP keyword exists in input science image FITS header
January 27, 2013	4.3	Frank Masci	* Added metrics to primary output psf-fit photometry table to support candidate vetting via machine learning; * Propagated aperture photometry measurements for specific aperture size (–apnum) to output PSF-fit photometry table; * Added more QA metrics to the output *_*diffqa.txt* files; * Updated –cn input parameter defaults (i.e., RA, Dec cols)
February 12, 2013	4.4	Frank Masci	* Updated thresholds for "stellarity" and source "size" thresholds supplied to –catfilt to support gain matching and relative pointing refinement
February 14, 2013	4.5	Frank Masci	* Enabled "BIGPDL" piddle storage functionality to handle lots of extractions per image, e.g., in galactic plane

Table of Contents

1 INTRODUCTION

PTFIDE (or *ptfide*) is a standalone script that performs differencing of science (or program) images with a reference (or template) image and then extracts *candidate* transients and variables from the differenced images. This is to support the Palomar Transient Factory's real-time processing pipeline, but it can also be run offline to support ancillary research. We stress that the purpose is to detect transient and variable candidates and measure their differential fluxes, with no further dissemination to determine if these are reliable to warrant further characterization. Even though our goal in general is to maximize the reliability of detections using a number of pre-calibration steps, the weeding out the false-positives will be performed by a downstream process. Details on the PTF survey and instrumental calibration performance are given in Law et al. (2009) and Ofek et al. (2012). An overview of the current science and reference-image pipelines can be found in Laher et al. (2013).

In summary, *ptfide* performs photometric throughput (or Zero-Point) matching; relative astrometric refinement; resampling (or reinterpolation) of the reference image onto the science image grid; differential spatially-dependent background matching; PSF-matching; image differencing; and source extraction from the differenced images. It operates on a list of pre-calibrated science frames that overlap with an existing reference image (e.g., a co-add). The PSF-matching is performed using either the algorithm developed by Alard & Lupton (1998; and extended by Alard 2000) as implemented in HOTPANTS (Becker 2009), or a method similar to that described by Bramich (2008) that involves direct estimation of Pixelated Convolution Kernels over image regions. We refer to the latter as the "PiCK" method. If requested, an automated procedure is used to decide if the science or reference image should be convolved with the PSF-matching kernel. The image differencing is bidirectional: "science – reference" and "reference – science". Candidates are detected and photometered on each difference image using either PSF-fit photometry, aperture photometry, or both. See below for details.

The key to good image differencing is in the preparation and pre-conditioning of the input images. We have strived in this tool to go beyond the capabilities of existing image-differencing tools by including a number of pre-calibration steps to optimally match a program image to a reference image. This includes matching the astrometry, photometric throughput (gain), and background as a function of position on the focal plane. The goal is to obtain difference images where systematic errors are minimized, i.e., where noise is purely dominated by Poisson and/or random instrumental (e.g., read-noise) fluctuations.

This document describes the processing steps in *ptfide* and its usage. We also discuss known caveats, potential pitfalls, liens, parameter tuning, quality assurance, and results from preliminary testing. We also present examples where known variable sources are recovered.

2 DEPENDENCIES

ptfide threads together both newly developed in-house software and a number of external public tools and libraries. Table 1 shows the dependencies. References for the external modules (where available) are also given.

Dependency / version	Purpose
Perl vsn >= 5.8.0	Primary language used for scripting and ancillary mathematical/image operations.
PDL vsn 2.4.10: built with "bad-value" support turned on.	Perl library for object-oriented vector and matrix operations.
Astro-WCS-LibWCS vsn 0.93. Requires *wcstools* library vsn >= 3.8.1.	Perl library to support World Coordinate System (WCS) & image-pixel transformations.
Ptfutils, *Pars*	In-house developed Perl modules specific to PTF operations.
SWarp vsn 2.19.1	Image resampling/interpolation tool that interprets the PV-distortion convention (Bertin, 2010).
SExtractor vsn 2.8.6	For preliminary fast source detection, aperture photometry, and shape characterization (Bertin, 2009).
xy2xytrans vsn 1.0	Fast transformation of a list of *x,y* source positions from one image with given WCS to another with different WCS.
HOTPANTS vsn 5.1.10. Requires *cfitsio* library.	Computes model-dependent, spatially-varying PSF-matching kernel between two images and performs image subtraction (Becker, 2009).
DAOPhot vsn II, 1/15/2004 [with minor modifications on filename string lengths made by F. Masci]. Requires *cfitsio* library. Automated via Perl script by F. Masci.	Source detection, aperture photometry and preparation of file inputs to support PSF-generation and PSF-fit photometry (Stetson, 1987, 2000).
allstar vsn 2/7/2001 [with minor modifications on filename string lengths made by F. Masci]. Requires *cfitsio* library. Automated via Perl script by F. Masci.	PSF-fit photometry tool. Also used for PSF-generation (Stetson, 1987, 2000).

Table 1: dependencies for the *ptfide* program

3 INPUT/OUTPUT SUMMARY

3.1 Inputs

Here are the primary inputs. See Section 4 for a full synopsis. Example inputs (and outputs) can be found in the RTB directories listed in Section 7.

- List of pre-processed science images in FITS format that have been astrometrically (and optionally photometrically calibrated), and quality assured. This includes all the necessary instrumental calibration steps: bias subtraction (both static and dynamic removal of any spatially-varying bias), flat-fielding, and anything else specific to the detectors.

- List of accompanying mask images in FITS format that encode for example: the locations of bad hardware pixels, saturated pixels and associated bleed artifacts, aircraft/satellite streaks, and cosmic ray hits.

- A template or reference image in FITS format representing a "static" representation of the sky that overlaps with the input (program) science images. This is typically a co-add of several or more high-quality science images, i.e., those with the *best seeing* and best instrumental, photometric and astrometric calibrations. See Section 3.2 for details.

- A source catalog table for the reference image to support refinement of the relative gain and astrometry. This is expected to have been generated by SExtractor since *ptfide* requires specific source metrics. For the required metrics and format, see the descriptions for the –catef and –cn input fields in the synopsis section (Section 4).

- Configuration files for the various sub-modules, thresholds, processing parameters, and switches to control the processing (see synopsis in Section 4).

3.2 Details regarding the Input Reference Image

Above, we mentioned that the reference image (co-add) should ideally be made from frames with the "best seeing". In general, it is desirable to create a reference image where the effective point source FWHM is generally smaller than that in the program science images. This is so the reference image is preferentially convolved during the PSF-matching step. However, this is not a strict requirement since *ptfide* can use internal metrics to decide which image to convolve. A smaller reference image FWHM is desirable since it minimizes the potential for error when selecting the image to convolve. This is because the metrics used to select the "best" image to convolve are themselves inherently noisy. A co-add of several well-calibrated frames with the best seeing (e.g., with FWHM <~ 1.8″ for the PTF observing sight, if available) will usually suffice for the reference-image.

Proper filtering of instrumental glitches and outliers (e.g., cosmic rays) is also important when creating reference images. Their presence will contaminate all program images that are differenced with it. This is particularly important in transient/variability surveys that often observe the same piece of sky with little dithering between frames and then reference images (co-adds) are made therefrom. The almost perfectly aligned frames can make detector and optical artifacts (e.g., glints, ghosts, source persistence, and intermittently bad pixels) difficult to remove using conventional outlier-detection methods based on frame-stack statistics.

Furthermore, *ptfide* requires a Zero-Point keyword (whose exact name can be specified via the *–zprkey* input parameter), SATURATE keyword, and WCS keywords in the reference image header. The ZP and WCS must be accurate since these determine the photometric and astrometric accuracy of the extracted transient/variable candidates.

3.3 Output Products

The core primary products generated by *ptfide* (unless specified otherwise) are summarized in Table 2. These are written to the directory specified by "*–od*". In general, the product filename is a concatenation of the input science image root filename and some mnemonic indicating the flavor of product, for example:

<path>/PTF_201204172687_i_p_scie_t062652_u013094258_f02_p004335_c08**_pmtchscimref.fits**

which represents a "PSF-matched science *minus* reference difference image" in FITS format pertaining to the science image with root filename shown. In the tables below, this root filename is given by <sci_img_fname>.

Product filename	Description
<sci_img_fname>_pmtchscimref.fits	PSF-matched 'science *minus* reference' difference image.
<sci_img_fname>_pmtchrefmsci.fits	PSF-matched 'reference *minus* science' difference image.
<sci_img_fname>_pmtchdiffunc.fits	Uncertainty image storing 1-σ uncertainty estimates for the difference images above.
<sci_img_fname>_pmtchdiffmsk.fits	Image mask for the difference images above (0 => good pixel; 1 => bad where difference images have "-999999" at corresponding locations).
<sci_img_fname>_pmtchscimrefpsffit.tbl	*Only if –psffit switch was set*: text file in IPAC table format storing results of PSF-fit photometry for the 'science *minus* reference' difference image.
<sci_img_fname>_pmtchrefmscipsffit.tbl	*Only if –psffit switch was set*: text file in IPAC table format storing results of PSF-fit photometry for the 'reference *minus* science' difference image
<sci_img_fname>_pmtchscimrefpsffit.reg	*Only if –psffit switch was set*: DS9 region-overlay file for PSF-fit photometry file:

	"_pmtchscimrefpsffit.tbl", created from science *minus* reference image difference.
<sci_img_fname>_pmtchrefmscipsffit.reg	*Only if –psffit switch was set*: DS9 region-overlay file for PSF-fit photometry file: "_pmtchrefmscipsffit.tbl", created from reference *minus* science image difference.
<sci_img_fname>_pmtchscimrefapphot.tbl	*Only if –apphot switch was set*: text file in IPAC table format storing results of DAOPhot aperture photometry for six apertures on the 'science *minus* reference' difference image.
<sci_img_fname>_pmtchrefmsciapphot.tbl	*Only if –apphot switch was set*: text file in IPAC table format storing results of DAOPhot aperture photometry for six apertures on the 'reference *minus* science' difference image.
<sci_img_fname>_pmtchscimrefapphot.reg	*Only if –apphot switch was set*: DS9 region-overlay file for aperture photometry file: "_pmtchscimrefapphot.tbl", created from science *minus* reference image difference.
<sci_img_fname>_pmtchrefmsciapphot.reg	*Only if –apphot switch was set*: DS9 region-overlay file for aperture photometry file: "_pmtchrefmsciapphot.tbl", created from reference *minus* science image difference.
<sci_img_fname>_pmtchconvrefdao.psf	*Only if –psffit switch was set*: text file storing PSF generated by *DAOPHot* from the kernel-convolved reference image and used in PSF-fit photometry. See Table 3 for FITS-formatted equivalent. **If sci image was convolved, file will be named "_pmtchconvscidao.psf".
<sci_img_fname>_pmtchconvsci.fits	The "_newscibmtch.fits" image product (defined below) *convolved with the PSF-matching kernel*. **Only generated if _newscibmtch.fits was actually selected for the convolution.
<sci_img_fname>_pmtchconvref.fits	Resampled ("*swarped*") reference image *convolved with the PSF-matching kernel*. **Only generated if the resampled reference image was actually selected for the convolution.
<sci_img_fname>_resampref.fits	Resampled ("*swarped*") reference image in the coordinate frame of the science image (input into the PSF-matching step), after science img WCS was refined to match the reference image WCS.
<sci_img_fname>_resamprefunc.fits	Uncertainty image storing 1-σ uncertainty estimates corresponding to the resampled

	reference image (_resampref.fits).
<sci_img_fname>_badmskref.fits	Bad-pixel image mask corresponding to the resampled reference image (0 => good pixel; 1 => bad). Currently, only saturated-pixels are tagged.
<sci_img_fname>_newscibmtch.fits	Science image that has its pixels gain and background-matched to the reference image, and with WCS refined to align with the reference image WCS (input into the PSF-matching step).
<sci_img_fname>_newsciuncbmtch.fits	Uncertainty image storing 1-σ uncertainty estimates corresponding to _newscibmtch.fits.
<sci_img_fname>_badmsksci.fits	Bad-pixel image mask corresponding to _newscibmtch.fits (0 => good pixel; 1 => bad).
<sci_img_fname>_diffbmtch.fits	'science *minus* reference' difference image *before* any PSF-matching. I.e., where the science image is WCS-refined, gain and background-matched to the ref image, and the ref image was resampled. This can be compared to the "*after* PSF-matched" difference (_pmtchscimref.fits) for QA.
<sci_img_fname>_diffqa.txt	*Only if –qa switch was set*: text file storing QA metrics on difference images before and after PSF-matching (for details, see Section 6).

Table 2: primary products from *ptfide* for a given input science frame with overlapping reference image. See Table 3 for additional products generated under debug mode.

If the debug switch (–d) is set, the diagnostic files listed in Table 3 are generated (in addition to the primary products listed in Table 2). Note that output files pertaining to either PSF-fitting and/or aperture photometry (from *DAOPhot*/*Allstar*) are only generated if either the *–psffit* and/or *–apphot* switch is set respectively.

Product filename	**Description**
<sci_img_fname>_inpsvb.fits	Regularized difference image of "gain and WCS-matched science *minus* resampled reference" with pixel outliers replaced by a local mode or median. This is used to compute the differential Spatially-Varying Background (SVB) map.
<sci_img_fname>_newscitmp.fits	Input science image to support the ref-image resampling. Has been gain and WCS-matched.
<sci_img_fname>_pmtchconvref.als	Intermediate *Allstar* output file from PSF generation off convolved reference image. This lists intermediate PSF-fit photometry results. **If sci image was convolved, file will be named "_pmtchconvsci.als".
<sci_img_fname>_pmtchconvref.ap	Intermediate *DAOPhot* output file from PSF generation off convolved reference image. This lists intermediate aperture photometry results. **If sci image was convolved, file will be named "_pmtchconvsci.ap".
<sci_img_fname>_pmtchconvref.coo	Intermediate *DAOPhot* output file from PSF generation off convolved reference image. This lists is the initial detections. **If sci image was convolved, file will be named "_pmtchconvsci.coo".
<sci_img_fname>_pmtchconvrefdaopsf.fits	FITS image showing postage stamps of spatially-varying PSF used for PSF-fitting. This is generated from DAOPhot's archane text file: "_pmtchconv[ref]dao.psf". **If sci image was convolved, file will be named "_pmtchconvscidaopsf.fits".
<sci_img_fname>_pmtchconvrefdaosub.fits	Intermediate *DAOPhot* output file from PSF generation off convolved reference image. **If sci image was convolved, file will be named "_pmtchconvscidaosub.fits".
<sci_img_fname>_pmtchconvref.lst	Intermediate *DAOPhot* output file from PSF generation off convolved reference image. This lists the stars picked for PSF generation. **If sci image was convolved, file will be named "_pmtchconvsci.lst".

<sci_img_fname>_pmtchconvref.lst.reg	DS9 region-overlay file corresponding to "_pmtchconvref.lst" defined above. **If sci image was convolved, file will be named "_pmtchconvsci.lst.reg".
<sci_img_fname>_pmtchconvref.nei	Intermediate *DAOPhot* output file from PSF generation off convolved reference image. This lists the *neighbors* of the stars picked for PSF generation. **If sci image was convolved, file will be named "_pmtchconvsci.nei".
<sci_img_fname>_pmtchconvrefxymag.txt	Text file listing *x*, *y* grid positions and mags for creating the FITS image of PSF postage stamps (_pmtchconvrefdaopsf.fits defined above). **If sci image was convolved, file will be named "_pmtchconvscixymag.txt".
<sci_img_fname>_pmtchdiffchisq.fits	FITS image of binned "pseudo chi-square values" for difference image after PSF-matching. See Section 6 for description.
<sci_img_fname>_pmtchrefmsci.als	Intermediate *Allstar* output file for reference *minus* science image difference. Lists raw PSF-fit photometry results.
<sci_img_fname>_pmtchrefmsci.ap	Intermediate *DAOPhot* output file for reference *minus* science image difference. Lists raw aperture photometry results.
<sci_img_fname>_pmtchrefmsci.coo	Intermediate *DAOPhot* output file for reference *minus* science image difference. Lists initial detections.
<sci_img_fname>_pmtchrefmscidaosub.fits	Intermediate *Allstar* output file for reference *minus* science image difference.
<sci_img_fname>_pmtchscimref.als	Intermediate *Allstar* output file for science *minus* reference image difference. Lists raw PSF-fit photometry results.
<sci_img_fname>_pmtchscimref.ap	Intermediate *DAOPhot* output file for science *minus* reference image difference. Lists raw aperture photometry results.
<sci_img_fname>_pmtchscimref.coo	Intermediate *DAOPhot* output file for science *minus* reference image difference. Lists initial detections.
<sci_img_fname>_pmtchscimrefdaosub.fits	Intermediate *Allstar* output file for science *minus* reference image difference.
<sci_img_fname>_resamprefwt.fits	Intermediate output weight image for resampled reference image from "*swarp*". Used internally for ref-image mask generation.
<sci_img_fname>_scisatpixels.fits	Image showing locations of just saturated pixels in input science image as tagged in

	accompanying mask image. Used for internal estimation of saturation level.
<sci_img_fname>_resamprefbck.fits	*Only if "–pmeth 2" was set*: image of spatially variable background estimated from resampled reference image and used to regularize sci and ref image inputs to optimize kernel derivations
<sci_img_fname>_newscibmtchmod.fits	*Only if "–pmeth 2" was set*: regularized sci image used to derive optimum convolution kernel together with "_resamprefmod.fits"
<sci_img_fname>_resamprefmod.fits	*Only if "–pmeth 2" was set*: regularized resampled-ref image used to derive optimum convolution kernel together with "_newscibmtchmod.fits"
<sci_img_fname>_pmtchkerncube.fits	*Only if "–pmeth 2" was set*: FITS cube of dimensions "*–kersz* × *–kersz* × *–kerXY*" storing spatially-dependent convolution kernels for the "PiCK" PSF-matching method
<sci_img_fname>_svb.fits	Image of the differential spatially varying background estimated from the input intermediate image: "_inpsvb.fits".
<sci_img_fname>_sxbck.fits	Diagnostic background image created by SExtractor run on input science image.
<sci_img_fname>_sxbckrms.fits	Diagnostic background RMS image created by SExtractor run on input science image.
<sci_img_fname>_sxobjects.fits	Ancillary image showing raw objects detected by SExtractor run on input science image.
<sci_img_fname>_sxrefremap.tbl	Table (text) file of positions and fluxes of *filtered* reference-image sources (read from "sx_ref_filt.tbl") transformed (using *xy2xytrans*) to the science image coordinate frame.
<sci_img_fname>_sx.tbl	Table (text) file output from SExtractor containing source metrics for science-to-(filtered) reference image source associations.
<sci_img_fname>_sx.reg	DS9 region-overlay file corresponding to "_sx.tbl".
sx_ref_filt.tbl	Table (text) file of positions and fluxes of sources selected from the input *–catref* reference image table file using the filtering parameters specified in *–catfilt*.
sx_ref_filt.reg	DS9 region-overlay file corresponding to "sx_ref_filt.tbl".

Table 3: additional products generated if the debug switch (–d) is specified

4 FULL I/O SPECIFICATION AND SYNOPSIS

ptfide is a Perl script that takes all of its input from the command-line. This command-line can be set-up and executed via a shell script or pipeline wrapper. A description of all inputs is given below. Required inputs and default values are indicated. Note that the default values shown here are not necessarily optimal. See the example call in Section 4.2 for the best parameters known at the time of writing.

4.1 Synopsis

The synopsis below can also be obtained by executing "*ptfide*" at the Unix prompt with no command-line arguments.

```
ptfide: version 4.5; last modified: 02.14.13
by F. Masci (fmasci@caltech.edu)
Purpose: resample, match, and difference calibrated science frames
with a reference image (co-add), then extract transient candidates

Usage: ptfide -param_names <value> ...
where -param_names, datatypes [and defaults] are below

-scilst:          str
 Input filename listing science image FITS files; possibly pre-
calibrated; assumed to overlap with reference image specified by '-
ref'; required

-msklst:          str
 Input filename listing mask image FITS files accompanying -scilist;
required

-ref:             str
 Input FITS filename of reference image (co-add); required

-catref:          str
 Input reference image source catalog. Required format is:
CATALOG_TYPE = ASCII_HEAD from SExtractor and must contain at least
these columns: FLAGS; XWIN_IMAGE; YWIN_IMAGE; ALPHAWIN_J2000;
DELTAWIN_J2000; FWHM_IMAGE; CLASS_STAR; FLUX_AUTO; FLUXERR_AUTO;
ISOAREAF_IMAGE. The -cn input specifies their column ordering

-cn:              list of int = 2,3,4,9,10,23,24,25,26,37
 Array of numbers defining the location order of the required column
names listed above as they appear in the -catref input table; column
#1 corresponds to the left-most column

-catfilt:         list of dbl = 0.8,90
 Thresholds for filtering input reference catalog: minimum and maximum
tolerable values for 'CLASS_STAR' and 'ISOAREAF_IMAGE' respectively

-od:              str
```

Directory name for output products (including any debug output); required

-cfgswp: str
Input configuration filename for SWarp; required

-cfgsex: str
Input configuration filename for SExtractor to support position/gain matching; required

-cfgcol: str
Input SExtractor column name configuration file to support position/gain matching; required

-cfgfil: str
Input SExtractor filter/convolution file; required

-cfgnnw: str
Input SExtractor neural network star/gal classification file; required

-cfgdao: str
Input generic DAOPHOT parameter file; required

-cfgpht: str
Input DAOPHOT photometry parameter file; required

-tmaxpsf: dbl = 300.0
Maximum threshold [#bckgnd sigma] above background in reference image for defining maximum usable pixel value when creating PSF off reference image

-tdetpsf: dbl = 10.0
DAOPHOT find-threshold [#bckgnd sigma] for PSF creation off reference image

-tmaxdao: dbl = 1000.0
Maximum threshold [#bckgnd sigma] above zero-background in difference image for defining maximum usable pixel value for source extraction/photometry off difference image

-tdetdao: dbl = 4.0
DAOPHOT find-threshold [#bckgnd sigma] for source extraction off difference image

-tchi: dbl = 5.5
Upper absolute threshold on 'chi' metric from ALLSTAR program below which extractions off difference image are retained; larger => more non-psf-like profiles are retained; will depend on quality of image subtractions where bad subtraction => larger residuals about PSF-fit and hence larger 'chi' values; absolute values will also depend on quality of pixel uncertainty estimates; calibrate offline first

-tshp: dbl = 1.2
Upper absolute threshold on 'sharp' metric from ALLSTAR program where extractions off difference image with -tshp <= sharp <= +tshp are retained; values closer to zero => sources have profiles closer to that of a 2D Gaussian with FWHM obtained from the mean FWHM of the corresponding science frame

-tsnr: dbl = 3.7
Threshold on flux signal-to-noise ratio from psf-fit photometry above which extractions off difference image are retained

-fatbits: list of int = 0,8,9,10
Fatal bits to mask in image products as encoded in input masks specified by -msklist; for no masking set to "-1"

-satbit: int = 8
Saturation bit number encoded in masks specified by -msklist; used for determining the saturation pixel value in science images

-expnbad: int = 0
Force an additional 'expnbad x expnbad - 1' pixels around each masked science frame pixel (according to -fatbits) and each saturation-threshold pixel in resampled reference image to also be bad; provides more complete blanketing, e.g., for saturated sources; expnbad = 0 => no additional pixels are masked in products; if set, must be an odd positive integer >= 3

-eg: dbl = 1.5
Electronic gain [e-/ADU] corresponding to native detector counts in ADU; used for internal pixel-uncertainty estimation

-sxt: dbl = 20
SExtractor detection threshold [#sigma] to support position/gain matching (DETECT_THRESH internal parameter)

-rad: dbl = 4.0
Match radius [pixels] to associate reference and science frame extractions to support position/gain matching

-nmin: int = 200
Minimum number of reference-to-science image source matches above which to proceed with position/gain matching

-dgt: dbl = 1.0
Minimum relative gain factor [%] above which to proceed with relative gain correction; only relevant if input science image has a valid ZP value that's expected to be correct; otherwise gain factor can be large if performing a recalibration with respect to reference image extractions

-dpt: dbl = 0.1

Minimum differential offset [pixels] above which to proceed with relative offset corrections (dX or dY or both)

-dgsnt: dbl = 5.0
Minimum S/N ratio in relative gain factor [#sigma] above which to proceed with relative gain correction

-dpsnt: dbl = 5.0
Minimum S/N ratio in differential offsets [#sigma] above which to proceed with relative offset corrections (dX or dY or both)

-gridXY: list of int = 4,8
Number of partitions per axis of input frame for local thresholding of pixels to support differential SVB computation

-tpix: dbl = 2.0
Threshold t[#sigma] for replacing pixel values > mode + t*sigma in a partition with partition mode; to support differential SVB computation

-tmode: dbl = 7.0
Threshold t[%] for replacing all pixels of a partition with global mode if its local mode is > (1 + t[%]/100) * global mode; to support differential SVB computation

-tsig: dbl = 100
Threshold t[%] for replacing all pixels of a partition with local mode if its robust sigma is > (1 + t[%]/100) * 'median of all partition sigmas'; to support differential SVB computation

-rfac: int = 16
Down-sampling factor along X and Y to speed up filtering during differential SVB computation

-szker: int = 41
Median-filter size (odd positive integer) to apply to downsampled image to support differential SVB computation [pixels]

-ker: str = LANCZOS3
Interpolation kernel for SWarp

-zpskey: str = IMAGEZPT
Keyword name for photometric Zero Point in input headers of _science_ FITS images listed in -scilst. Optional; if present, we compute a delta-gain correction with respect to reference image ZP and recalibrate to match ref-image ZP if -pg switch is set and correction is significant. If not present, we simply calibrate the science frame pixels to match ref-image ZP nonetheless

-zprkey: str = IMAGEZPT
Keyword name for photometric Zero Point in input header of _reference_ FITS image specified by -ref; required

```
-pmeth:           int = 2
 Method to derive PSF-matching kernel between science and reference
images: 1 => 'hotpants' program; 2 => Pixelated Convolution Kernel
(PiCK) method similar to Bramich (2008)

-hot1:            list of dbl = 8,14,16,32,2,2
 For -pmeth 1: parameter input array 1 for 'hotpants' program in this
order: [-r,-rss,-nsx,-nsy,-mins,-ko] = convolution kernel half width
[pixels]; source-extraction substamp region half width [pixels];
number of partitions along X-axis; number of partitions along Y-axis;
factor of convolution kernel half width at which to expand masked
saturated pixels in internally created _reference_ image mask and in
difference image products; polynomial order for the spatial variation
in the convolution kernel

-hot2:            list of dbl = 4,4,0.40625,4,0.65,2,1.04,2,1.664
 For -pmeth 1: parameter input array 2 for 'hotpants' program: [-ng] =
#gaussians,polydegree0,sigmagauss0,...,polydegreeN,sigmagaussN; where
the sigmagauss_i are in units of pixels

-calfwhms:        list of dbl = 2.52,1.82
 For -pmeth 1: point source profile FWHMs [pixels] for the science and
reference images (in this order) used to tune the Gaussian kernel
widths specified in -hot2; these are used to rescale the widths from -
hot2 according to the actual input science and reference image FWHMs
if the -adapt switch is set

-conv:            str = ref
 For -pmeth 2: image to convolve; can be 'sci', 'ref', or 'auto'. The
'auto' option uses the relative sizes of the sci and ref FWHM measures
whereby sci is convolved only if FWHM(sci) <= 1.1*FWHM(ref), otherwise
the ref is convolved

-kersz:           int = 7
 For -pmeth 2: linear size of kernel stamps in pixels, must be odd
number >= 5

-kerXY:           list of int = 5,10
 For -pmeth 2: number of grid partitions along X,Y axes in which to
compute local kernels; >=1 for each axis

-nbreftb:         int = 65
 For -pmeth 2: number of pixel rows to force as "bad" on top and
bottom of internal images used for kernel derivation to account for
edge effects in resampled reference image

-nbreflr:         int = 35
 For -pmeth 2: number of pixel rows to force as "bad" on left and
right of internal images used for kernel derivation to account for
edge effects in resampled reference image

-bckwin:          int = 31
```

For -pmeth 2: linear window size [downsampled pixels] for median filtering downsampled reference image when computing spatially varying background image; note: downsampling factor for resampled ref image is internally fixed at 16x per axis. Must be odd positive integer

-tsat: dbl = 0.65
For -pmeth 2: factor threshold to perform more conservative tagging of resampled ref image pixels satisfying >= tsat * saturate where tsat <= 1 and 'saturate' is from resampled ref-image header after ref image was made using (preferably) 'mkcoadd' tool then resampled onto sci frame using SWarp

-qas: list of int
Coordinate range of rectangular region in image for computing QA metrics in difference images if -qa switch was set; format is: "xmin,xmax,ymin,ymax" where pixel numbering is unit based, i.e., 1 <= xmin < xmax <= NAXIS1, 1 <= ymin < ymax <= NAXIS2; default is to use whole image if -qas not specified, but -qa was

-apnum: int = 4
Aperture (line) number in input -cfgpht configuration file for which photometry information for this sized aperture should be propagated to output psf-fit photometry table

-wmode: switch
Switch to use local and global image modes (instead of medians) to support differential SVB computation

-adapt: switch
For -pmeth 1: switch to dynamically rescale the Gaussian kernel widths specified by -hot2 (which are tuned to the FWHM values specified by -calfwhms) according to the actual observed FWHM values of the input science and reference images

-psffit: switch
Switch to perform psf-fit photometry on difference images with prior psf estimation off [convolved] reference image

-apphot: switch
Switch to perform aperture photometry on difference images

-pg: switch
Switch to compute sci-to-ref image position and gain corrections and apply if significant. Must be specified if ZP keyword (-zpskey input) is missing from science headers so images can be photometrically calibrated

-pcln: switch
Switch to 'pre-clean' (remove) output directory specified by -od

-qa: switch

```
 Switch to generate QA metrics off difference images before and after
PSF-matching within image slice defined by -qas string; results are
written to stdout and a text file of the form: <sci-img-
rootname>_diffqa.txt

-d:              switch
 Switch to generate debug diagnostic files

-v:              switch
 Switch to increase verbosity to stdout
```

4.2 Example Call

Here's a typical command line (delimited by "\") with best-optimized parameters known at the time of writing. This can be called from a shell script or a higher-level pipeline wrapper. Note: lines starting with a " # " should be omitted from production. They're included here for completeness.

```
/home/ptffm/bin/ptfide \
  -scilst SciList.txt \
  -msklst MskList.txt \
  -ref /stage/ptf_eng_users_frank/M13coad/trimaverage.fits \
  -catref /stage/ptf_eng_users_frank/M13coad/sx_ref.tbl \
  -cn 2,3,4,9,10,23,24,25,26,37 \
  -catfilt 0.8,90 \
  -od diffproducts \
  -cfgswp /home/ptffm/cfg/swarp_params.txt \
  -cfgsex /home/ptffm/cfg/sex_params.txt \
  -cfgcol /home/ptffm/cfg/sex_outcols_assoc.txt \
  -cfgfil /home/ptffm/cfg/default.conv \
  -cfgnnw /home/ptffm/cfg/default.nnw \
  -cfgdao /home/ptffm/cfg/daophot.opt \
  -cfgpht /home/ptffm/cfg/photo.opt \
  -tmaxpsf 2000.0 \
  -tdetpsf 50.0 \
  -tmaxdao 1000.0 \
  -tdetdao 4.0 \
  -tchi 5.5 \
  -tshp 1.2 \
  -tsnr 3.7 \
  -fatbits 8,9,10,12 \
  -satbit 8 \
  -expnbad 3 \
  -eg 1.5 \
  -sxt 8.0 \
  -rad 4.0 \
  -nmin 200 \
  -dgt 1.5 \
  -dpt 0.07 \
  -dgsnt 5.0 \
```

```
  -dpsnt 5.0 \
  -gridXY 4,8 \
  -tpix 2.0 \
  -tmode 500.0 \
  -tsig 100 \
  -rfac 16 \
  -szker 41 \
  -ker LANCZOS3 \
  -zpskey IMAGEZPT \
  -zprkey IMAGEZPT \
  -pmeth 2 \
  -hot1 8,14,16,32,2,2 \
  -hot2 4,4,0.40625,4,0.65,2,1.04,2,1.664 \
  -calfwhms 2.52,1.82 \
  -conv ref \
  -kersz 7 \
  -kerXY 5,10 \
  -nbreftb 65 \
  -nbreflr 35 \
  -bckwin 31 \
  -tsat 0.65 \
  -qas 41,2008,41,4056 \
  -apnum 4 \
#  -wmode \
#  -adapt \
  -psffit \
#  -apphot \
  -pg \
  -pcln \
  -qa \
#  -d \
  -v
```

5 PROCESSING FLOW

Figure 1 gives a summary of the processing steps in *ptfide*. We expand on some of the "non-trivial" steps further below.

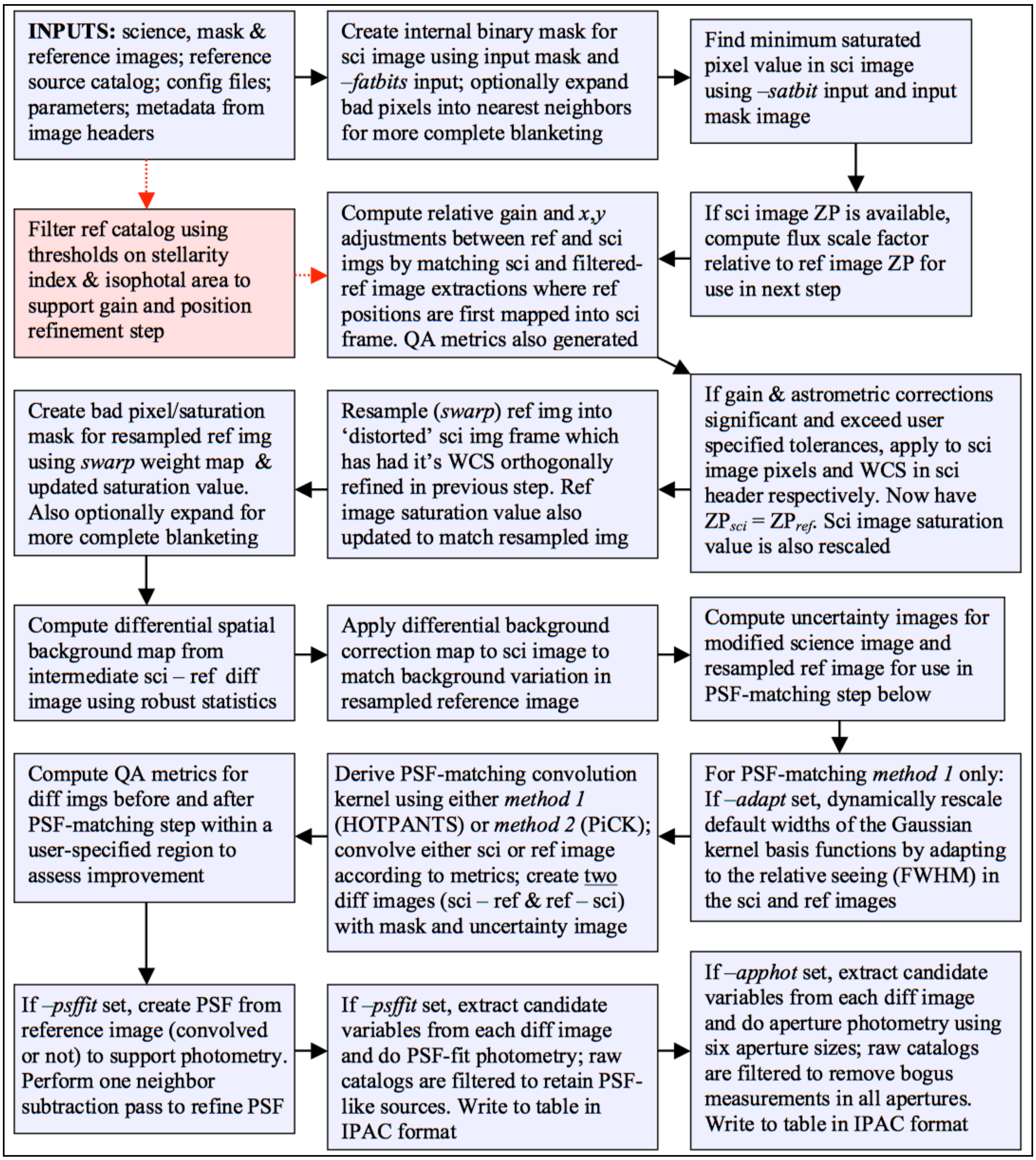


Figure 1 – Processing flow in *ptfide* for a given input science (program) image read from an input list. The step in the red box is performed early in the program and is outside the main processing loop.

5.1 Optional Bad-Pixel Expansion

The purpose of this step is to force an additional *N* x *N* - 1 pixels around each masked input science-image pixel (according to the fatal bits in *–fatbits*) to also be "bad", where *N* = input from *–expnbad*. A value of zero implies no expansion. This provides more complete blanketing of bad pixel regions, e.g., for saturated sources in particular whose unmasked edges and associated bleed artifacts may lead to residuals in the difference images and hence unreliable detections. This expansion operation is also performed on the *resampled* reference image after tagging saturated pixels by thresholding on the SATURATE keyword value. Note that both the reference image resampling and its convolution in the PSF-matching step causes bad pixel regions to "grow" to an area which may not match the nominally-masked regions in the input science image. The expansion step was implemented for conservatism. The new (internally created) science-image mask is propagated along to the PSF-matching step and also written to a file in FITS format (output **_badmsksci.fits*).

5.2 Relative Gain and Astrometric Refinement

First, the input reference catalog (*–refcat*) is filtered to retain primarily isolated point sources in uncrowded regions of the reference image using the SExtractor metrics: "stellarity" (CLASS_STAR column) and "isophotoal area" (ISOAREAF_IMAGE). The respective thresholds for these metrics are specified by *–catfilt* (defaults = 0.8,90). An intermediate filtered reference catalog table is made (with filename *sx_ref_filt.tbl* that is only visible under debug mode). Next, we map the positions in this filtered catalog to the coordinate frame of the science image using the "*xy2xytrans*" utility. The reason for this is to support the efficient and robust source-association step within *SExtractor* since this software only performs source matching in the image *x*, *y* frame. A new intermediate catalog table is made (with filename * *_sxrefremap.tbl* that is also only visible in debug mode). Next, *SExtractor* is run in "association mode" by extracting sources from the science image above a S/N threshold specified by *–sxt* (default = 20) that *only* match the sources in the filtered reference catalog within a radius of *–rad* (default = 4 pixels). For multiple matches within this radius, the nearest source is selected. A source-matched catalog table is generated (**_sx.tbl*) that is only visible under debug mode. To support offline debugging, DS9 region files corresponding to these table outputs with extension *.reg* are also generated.

Following the source matching, the science image source fluxes are rescaled according to any ZP difference between the science and reference images, but only if a ZP was available for the input science image. If available, this ZP need not be accurate since *ptfide* will always refine the relative gain between science and reference image such that the science image attains the *supposedly more accurate* reference image ZP. If a ZP is not present, a gain calibration factor is still computed and the ZP's matched. The gain factor (*Dg*) is computed using the median of the ratio of science to reference image source fluxes in the matched catalog table. Orthogonal position differences *Dx* and *Dy* are also computed and the median of these differences computed to obtain the overall position shift of the science image relative to the reference. Uncertainties for these quantities are also computed using a robust statistic – the rescaled Median Absolute Deviation (MAD) divided by the square root of the number of matches. These are used to decide whether the corrections should be applied or not (see below). Furthermore, 5th – 95th percentile ranges of the flux ratios and source position shifts in x, y are also computed for QA (see Section 6 for details).

The relative gain estimate Dg is used to rescale the science image pixels only if the following criteria are satisfied: the number of matches from which it was derived exceeds *–nmin* (default = 200); 100*|1 - Dg| exceeds the threshold *–dgt* (default = 1%); and its S/N (= $|1 - Dg|/\sigma_{Dg}$) exceeds *–dgsnt* (default = 5). Similarly, the orthogonal position corrections Dx, Dy are applied to the science image WCS keyword values (CRPIX1, CRPIX2 respectively) only if the following criteria are satisfied: the number of matches also exceeds *–nmin*; either $|Dx|$ or $|Dy|$ exceed *–dpt* (default = 0.1 pixel); and S/N (= $|Dx|/\sigma_{Dx}$ or $|Dy|/\sigma_{Dy}$) exceeds *–dpsnt* (default = 5).

5.3 Differential Spatially Varying Background (SVB) correction

We match any spatially-varying background differences between the rescaled, astrometrically-refined science image and resampled reference image using a robust image-partitioning method. The background correction map is estimated from a preliminary, pre-conditioned *sci – ref* difference image (debug output file *_inpsvb.fits*). Operating off a difference image minimizes any biases from bright extended emission (e.g., galaxies).

The input difference image is first partitioned into M x N rectangles where M, N are specified by *–gridXY*. Pixel modes (or medians) are computed globally (using the entire image) and within each partition separately using only good (non-NaN'd) pixels. Modes are only computed if the *–wmode* switch is specified, otherwise medians are computed. In the following description, "mode" can be interchanged with "median" if the *–wmode* switch wasn't specified.

First, we replace all pixel values in any partition with the global mode if its local mode exceeds or is below the global mode by a relative percentage specified by *–tmode*, i.e., if 100*|*mode* - *globalmode*|/*globalmode* > *tmode* is satisfied. Furthermore, we replace pixel values in those partitions with the *localmode* if |*pixelvalue* - *localmode*| > $t\sigma$ is satisfied, where t is a threshold specified by *–tpix* and σ is a robust local RMS estimated from a trimmed standard-deviation of low-tail values below the local mode. The resulting outlier-trimmed modal map is further regularized by resetting all pixels in those partitions with local RMS > *ts* * "median of all partition sigmas" to the global image mode where *ts* is a relative threshold specified by *–tsig*. This avoids any noisy regions (due to e.g., excessive Poisson noise from bight emission) from having an adverse affect on the background estimate. Next, we down-sample this preliminary background map using the binning factor provided in *–rfac* (default = 16). This binning uses a local average and is performed to speed up the filtering done next. The down-sampled map is median filtered using a window size specified by *–szker* (default = 41 x 41 *down-sampled* pixels) and then the resulting image (whose partition boundaries are smoothed out) is up-sampled back to the original image dimensions for use downstream.

At the end, the resulting differential background correction map (debug output file *_svb.fits*) is subtracted from the science image so its background (and any variations) match those in the accompanying reference image.

5.4 Pixel Uncertainty Estimation for (modified) Image Inputs

We compute uncertainty images corresponding to the *not-yet*-PSF-matched, but rescaled, background-matched science image, and resampled reference image. These are written to FITS files for input into the PSF-matching step below (primarily HOTPANTS for PSF-matching method 1 [–*pmeth* 1]).

We use a semi-empirical robust method to compute the pixel uncertainties. First, we estimate the robust background RMS (σ^2_{bcksci}) and modal value (m_{bcksci}) in a partition of the science image where these estimates are expected to be the "lowest" over the whole image and hence conservative in the sense that biases from bright emission and/or source-confusion is minimal. The coordinate ranges defining this partition are returned by the background matching step upstream. The 1-sigma uncertainty for a pixel signal in the science image is approximated as:

$$\sigma_{sci} \approx \left[S\left(\frac{DN - m_{bcksci}}{g} \right) + \sigma^2_{bcksci} \right]^{1/2},$$

where S is the scale factor used for the ZP-matching step above and is needed here since the actual counting of electrons (for the Poisson term) is with respect to the native detector ADU counts (or *DN*). g is the detector's electronic gain in *e*-/DN (input –*eg*). We have subtracted a robust background from the pixel signals since any Poisson-noise from background is already implicitly included in the σ^2_{bcksci} term and we remove any unknown (or hidden) bias level that is not photo-electron generated. Furthermore, σ^2_{bcksci} also implicitly includes the read-noise component. Note that pixels where $\mathrm{DN} - m_{bcksci} < 0$ are reset to zero.

For the reference image pixel uncertainties, we first compute a robust background RMS (σ^2_{bckref}) using the same ("source-deficient") partition region as above. Since the reference image may have been created offline from a co-add of science frames with unknown ZP-scaling factors, the Poisson-noise contribution will be difficult to compute precisely. Instead, we approximate the 1-sigma uncertainty for a pixel signal in the reference image by scaling from the science image uncertainties (σ_{sci}) estimated above:

$$\sigma_{ref} \approx \sigma_{sci}\left(\frac{\sigma_{bckref}}{\sigma_{bcksci}} \right).$$

This is expected to be a reasonable approximation since the pixel signals (DN) are known to be conserved between the *rescaled* science image and internally resampled reference image. In other words, the Poisson component is not expected to change much between these images and any ($1/\sqrt{N}$) diminution in the overall noise in the reference image due to co-addition is effectively handled by the scaling factor $\sigma_{bckref} / \sigma_{bcksci}$.

5.5 PSF Matching Method 1: HOTPANTS

We first experimented with the algorithm implemented in HOTPANTS. This can be called from *ptfide* by specifying *–pmeth* 1. This program creates a PSF-matching convolution kernel and an initial difference image. First, we use customized Perl code to prepare the images and parameters for input into HOTPANTS. The outputs are then massaged to generate an accompanying uncertainty image and mask. A summary of HOTPANTS can be found in Becker (2009). Here we summarize some of the assumptions and details specific to our needs.

HOTPANTS is based on the algorithm of Alard & Lupton (1998) and extended by Alard (2000). In brief, it attempts to derive a model convolution kernel $K_{ai}(u,v)$ by minimizing the following cost function:

$$C = \sum_{x,y}\left[I(x,y) - K_{a_i}(u,v) \otimes R(x,y)\right]^2, \qquad (1)$$

where $I(x,y)$ is the science (program) image and $R(x,y)$ the reference (template) image. Note that $I(x,y)$ could have been convolved here instead (if determined *a priori* to have better seeing than the reference) with no change in the end result. At the time of writing, we have parameterized the kernel model as a function of **four** Gaussians multiplied by “shape-morphing” polynomials:

$$\begin{aligned} K_{a_i}(u,v) = & \sum_{p1,q1} a_{p1q1} u^{p1} v^{q1} \exp\left[\frac{-\left(u^2+v^2\right)}{2\sigma_1^2}\right] \\ & + \sum_{p2,q2} a_{p2q2} u^{p2} v^{q2} \exp\left[\frac{-\left(u^2+v^2\right)}{2\sigma_2^2}\right] \\ & + \sum_{p3,q3} a_{p3q3} u^{p3} v^{q3} \exp\left[\frac{-\left(u^2+v^2\right)}{2\sigma_3^2}\right], \\ & + \sum_{p4,q4} a_{p4q4} u^{p4} v^{q4} \exp\left[\frac{-\left(u^2+v^2\right)}{2\sigma_4^2}\right], \qquad (2) \end{aligned}$$

where for PTF image data, we recommend the following polynomial orders:

$$0 \le p_1 + q_1 \le 4$$
$$0 \le p_2 + q_2 \le 4$$
$$0 \le p_3 + q_3 \le 2$$
$$0 \le p_4 + q_4 \le 2$$

and Gaussian widths:

$$\sigma_1 = 0.40625 \text{ pixels}$$
$$\sigma_2 = 0.65 \text{ pixels}$$
$$\sigma_3 = 1.04 \text{ pixels}$$
$$\sigma_4 = 1.664 \text{ pixels}.$$

You may have noticed that these form a geometric series:

$$\sigma_n = \sigma_1 \beta^{n-1}, \tag{3}$$

where $n = 1, 2, 3, 4$, and $\beta = 1.6$. This relation was used in the simulation work of Israel et al. (2006) and it greatly simplified our parameter tuning. See Section 7 for our parameter optimization procedure.

The prescription above is specified by the *ptfide* parameter set: "*–hot2* 4, 4, 0.40625, 4, 0.65, 2, 1.04, 2, 1.664", where the first "4" is the number of basis functions, followed by pairs of parameters specifying the polynomial order and Gaussian width for each component. Note that these polynomials determine the spatial variation (and skew) of each basis function in Eq. (2) *only* within a predefined footprint for the kernel model, which we fix at 17 × 17 pixels. As parameterized above, the terms with the narrowest Gaussian widths are allowed to have the fastest *local* variation, and the widest have the slowest variation.

If both the science and reference images had spatially invariant PSFs over the focal plane, a constant kernel model as represented by Eq. (2) would have sufficed. However, we know this is not the case. To account for spatial dependencies, each basis coefficient $a_{p,q}$ in Eq. (2) is modeled as a polynomial in x, y over an image:

$$a_{p_n q_n}(x,y) = \sum_{r_n s_n} a_{r_n s_n} x^{r_n} y^{s_n}, \tag{4}$$

where

$n = 1,\ 2,\ 3,\ 4$ and

$0 \le r_n + s_n \le 2.$

That is, we fix the polynomial order for the spatial dependence at 2. This is controlled by the "–*hot1*" *ptfide* parameter set. We advise keeping this order at ≤ 2 since the number of free parameters in the kernel model grows dramatically with increasing order for the spatial dependence, causing one to "over-fit" when sources are scarce. In general, we found that higher-order polynomials for the spatial dependence did not significantly reduce the overall difference-image RMS (when other parameters were at their optimum values).

The solution to the minimization of Eq. (1) therefore amounts to solving for the coefficients $a_{r,s}$ when Eq. (4) is expanded into Eq. (2) for each n. The linearity in the model parameters makes this a straightforward matrix inversion problem. With the spatial dependence included, the number of coefficients to solve (per science/reference image pair) is 252! Thankfully, there are usually enough good quality sources in PTF images to ensure a sufficient number of degrees-of-freedom for such a fit. For comparison, if we were to assume a spatially uniform kernel model, the number of free parameters would drop to 42.

Note that HOTPANTS also allows one to model a differential background variation in the difference image using a separate polynomial, but we force it to fit a constant. This is because our background-matching method (as applied upstream; see Section 5.3) is better at catching local variations and is more robust against outliers, errors in the relative gain, and the presence of bright extended emission.

Before proceeding to solve for the model coefficients in Eqs (2) and (4) using HOTPANTS, *ptfide* has an option to dynamically rescale the default Gaussian widths σ_n in Eq. (2) according to the relative seeing-FWHM in the input science and reference images (let us label these FW_{sci} and FW_{ref} respectively). This rescaling is only performed if the *–adapt* switch is set. The science-image FWHM is obtained from its FITS header and the reference image FWHM is the median of the FWHM measurements from the input reference catalog (*–catref*) *after* filtering for "stellarity" and isophotoal area using the thresholds in *–catfilt*. The input default widths (σ_n) specified in *–hot2* are assumed to be optimized for a science and reference image with FWHMs specified in the *–calfwhms* input (see Section 7 for our optimization procedure). Let us label these FW_{calsci} and FW_{calref} respectively. The new widths of the Gaussian basis functions for use in HOTPANTS are then given by:

$$\sigma_n(\text{new}) = \frac{\sqrt{\left|FW_{sci}^2 - FW_{ref}^2\right|}}{\sqrt{\left|FW_{calsci}^2 - FW_{calref}^2\right|}}\sigma_n, \qquad (5)$$

i.e., the scale factor is simply the ratio of the expected width of the new convolution kernel to that used to tune the input σ_n for a specific science and reference image. This is still an approximation since it depends on the accuracy of the input FWHM measures estimated from the profiles of selected *point* sources in each image. Nonetheless, this rescaling is expected to get us closer to the optimal solution for the science and reference frame in question.

To avoid numerical instability and erroneous FWHM measures from giving absurd scale factors in Eq. (5), we constrain the *new* width of the first (narrowest) basis function to fall within $0.3 \leq \sigma_1(\text{new}) \leq 4.0$ pixels. If the scale factor results in a $\sigma_1(\text{new})$ outside this range, we force it to either the minimum or maximum value of this range, whichever is closest. The scale factor is also modified accordingly before rescaling the other σ_n (for $n > 1$). The lower-bound on $\sigma_1(\text{new})$ is driven primarily by a suspected anomaly in HOTPANTS whereby small Gaussian widths result in convolved-images with all pixels set to NaN. The upper-bound is a conservative limit selected from testing. In the end, for a typical input reference image with FWHM <~ 1.5 pixels, the allowed range for σ_l(new) implies we can reliably derive PSF-matching kernels for science and reference image pairs whose ratio in FWHM values span the range ~ 1.1 – 5.9. This is expected to cover most images in general. The performance and usefulness of *ptfide*'s dynamic rescaling option is further discussed in Section 7.

5.6 PSF Matching Method 2: the Pixelated Convolution Kernel method (PiCK)

To compare with the Alard & Lupton (1998) algorithm, we have implemented a new PSF-matching algorithm that has recently gained some popularity in the time-domain community. This method was described by Bramich (2008) and we have implemented an enhanced version in *ptfide* that we have dubbed "PiCK". This can be called from *ptfide* by specifying *–pmeth* 2.

One difficulty with the Alard & Lupton (1998) or Alard (2000) construction is the prior specification of the basis functions, primarily the Gaussian widths, the orders of the shape-morphing polynomials, and their spatial dependence. One cannot be sure one has found the most optimal parameter set to describe the bulk of images in a survey acquired under different conditions and behaviors of the instrumentation. There is always the danger of tuning to a fixed set of observations (or even worse, simulations) that may not be representative of the survey in general. This difficulty is described in our HOTPANTS parameter tuning in Section 7. We are not saying that HOTPANTS does not satisfy our needs, we just don't know if we've made the right choice *for all time* since a large amount of experimentation on a large dataset is needed. There are times when HOTPANTS works perfectly, but there are more times when it fails completely, which we usually attribute to instrumental calibration errors. We are after a method that is robust against errors in the calibration. The PiCK method appears to achieve this.

In a nutshell, the PiCK method attempts to find a linear-least squares solution *for each individual pixel* of the convolution kernel K_{lm} by minimizing the following cost function:

$$C = \sum_{i,j}\left[I_{ij} - \left(\left\{K_{lm} \otimes R_{ij}\right\} + dB\right)\right]^2, \qquad (6)$$

where I_{ij} are pixels in the science image, R_{ij} the reference image (resampled in the science frame), and dB is a differential background that is assumed to be constant over the region spanned by pixels i,j. The l,m are column and row pixel indices in the convolution kernel satisfying $-(S_K - 1)/2 \leq l,m \leq (S_K - 1)/2$, where S_K is the linear dimension of the square kernel footprint (input parameter *–kersz* with default = 7 pixels). This is an odd number so that the kernel is centered at l,m = 0,0. If the quantity in square brackets in Eq. (6) were divided by prior pixel uncertainties (for both the science and convolved reference image if non-negligible), it would become the standard χ^2 metric. We do not include pixel uncertainties in the cost function since previous experience has show that this leads to biased solutions when weighting by (*heteroskedastic*) Poisson uncertainties. Also, Eq. (6) could be rewritten so that I_{ij} is convolved instead (i.e., if determined *a priori* to have better seeing than the reference). Equation (6) can be rewritten:

$$C = \sum_{i,j} \left[I_{ij} - dB - \sum_{l} \sum_{m} K_{lm} R_{(i+l)(j+m)} \right]^2, \tag{7}$$

where the resulting model image is given by:

$$M_{ij} = dB + \sum_{l} \sum_{m} K_{lm} R_{(i+l)(j+m)} \tag{8}$$

We allow the kernel K_{lm} to be of any shape, with asymmetry included. One can add a regularizing constraint for the kernel shape in Eq. (7), but we prefer to keep its estimate as flexible as possible. In the language of basis functions (analogous to Eq. 2), the K_{lm} are just the amplitudes of a two-dimensional array of delta-functions: $K(u,v) = K_{lm}\delta(u - l)\delta(v - m)$. The problem reduces to solving for the $S_K \times S_K$ pixel values in the kernel and a constant dB over pixels i,j in an image such that C is a minimum. Computing the partial derivatives of C with respect to an arbitrary pixel value $K_{lo,mo}$ and offset dB, and setting to zero yields the following relations:

$$K_{lo,mo} \sum_{i,j} R_{(i+lo)(j+mo)} R_{(i+l)(j+m)} + dB \sum_{i,j} R_{(i+lo)(j+mo)} = \sum_{i,j} I_{ij} R_{(i+lo)(j+mo)} \tag{9}$$

$$\sum_{lo} \sum_{mo} K_{lo,mo} \sum_{i,j} R_{(i+lo)(j+mo)} + dB = \sum_{i,j} I_{ij}. \tag{10}$$

These can be transformed into a system of $S^2_K + 1$ equations in $S^2_K + 1$ unknowns for each l_o,m_o in the range $-(S_K - 1)/2 \leq l_o,m_o \leq (S_K - 1)/2$ and each l,m satisfying $-(S_K - 1)/2 \leq l,m \leq (S_K - 1)/2$. These indices form respectively the rows and columns of a matrix equation **A.K = B** which can be inverted to solve for the vector **K** whose elements correspond to the kernel pixel values K_{lm} and offset dB.

The sum of the resulting kernel pixel values,

$$K_{SUM} = \sum_{l} \sum_{m} K_{lm} \tag{11}$$

is a measure of the relative gain between the reference and science image pixels. This can be used as a diagnostic to assess the accuracy of the relative photometric calibration upstream (Section 5.2). Values of K_{SUM} that significantly deviate from unity may indicate a problem with the upstream calibration (if any was attempted).

To account for spatial variations in the convolution kernel, we solve for **K** in image partitions over a $N_x \times N_y$ grid, where N_x, N_y are specified by the *–kerXY* input parameter. For partition boundaries internal to the image edges, we allow for an overlap of $(S_K - 1)/2$ pixels (half the linear size of the kernel) between partitions along each axis. This ensures continuity across partition boundaries and minimizes boundary effects following the convolution. Prior to constructing the matrix elements for an image partition defined by Equations (9) and (10), we first regularize the input science and resampled reference images to ensure a more optimal solution.

The regularization steps are as follows:

- Mask bad/saturated pixels in the science and reference images and mask additional neighboring pixels for more complete blanketing (step in Section 5.1). Currently a 3 × 3 pixel filter is used to mask additional bad pixels around each nominal bad pixel. For reference images in particular, we also lower the saturation threshold level for conservatism when tagging saturated pixels. The factor by which the nominal reference image saturation level (the SATURATE keyword value) is lowered is specified by the –*tsat* parameter.

- Mask all bad/saturated pixels in the science image where the calculation of the corresponding model pixel M_{ij} (Eq. 8) includes a bad/saturated pixel in its convolution domain on the reference image. The reverse is performed if the science image were convolved instead. This step involves convolving the reference image bad-pixel mask with a mock convolution kernel filled with 1's (the same size as that to be derived) and masking the additional pixels that were spread by the convolution. This implies that a single bad pixel on the reference image can affect a large number of pixels on the science image if the convolution kernel is too large. We therefore advise keeping the kernel size (parameter *–kersz*) as small as possible. We found that a linear size of ~3.5 × the FWHM of the observed PSF was reasonable. Also, a large kernel can severely impact the runtime.

- Mask the borders of the reference image by an amount that will catch most of the "crust". I.e., since the reference image is expected to be a co-add of multiple science images, imperfect dithering will cause non-uniform coverage around the edges. To avoid these regions from affecting the kernel solutions, we mask "*–nbreflr*" pixels from the left and right edges, and "*–nbreftb*" pixels from the bottom and top edges.

- We compute a local smooth varying background (SVB) from the *reference* image by first replacing bad pixels with the global mode value, down-sampling the reference image using bin-averaging (for speed), median filtering using a window of linear size *–bckwin*, and then up-sampling back to the original image dimensions. If the debug switch is set (–d), the SVB is written to a FITS image with filename **_resamprefbck.fits*. This same SVB image is subtracted from both the science and reference images to accentuate point sources. Otherwise, complex background emission will have an adverse affect on the kernel solution. Note that the same (reference image-derived) SVB image is subtracted from both images since the spatially-varying background in the science image was already accurately matched to that in the reference image upstream (Section 5.3). Retaining the same background level difference as present in the input images further minimizes errors in the kernel solution.

- All bad pixels tagged in the masking operations above are assigned values of zero in the reference and science images so they will have no contribution to the pixel sums in Equations (9) and (10). Also, all pixels in the science image that have a bad pixel at the same location in the reference image are set to zero. Likewise, all pixels in the reference image that have a bad pixel at the same location in the science image are set to zero. Failure to enforce this mutual masking leads to biases in the kernel solutions. If the debug switch is set, the final regularized science and reference images for input into the kernel derivation step are written to FITS images with filenames **_newscibmtchmod.fits* and **_resamprefmod.fits* respectively. Note that these specially regularized images are *only* used to derive the kernels. The original science and resampled reference images (prior to regularization) are used in the final convolution and image-differencing steps (see below).

The convolution kernel is estimated for each image partition by solving the matrix system defined by Equations (9) and (10). If the debug switch is set (–d), the kernel stamp images from all $N_x \times N_y$ image partitions are stored in a FITS cube and written to the file "**_pmtchkerncube.fits*". Note that we found it unnecessary (so far) to interpolate the partitioned kernel solutions to estimate the kernel at any *x,y* image location. We see no or very little boundary effects after the image partitions are convolved with their respective kernels, are background adjusted (using *dB*) and differenced. Using the gridded kernel solutions directly is still an approximation, but the grid is fine enough to properly model the local kernel to good accuracy since a sufficient number of point sources always exist within a partition.

As each partition's kernel is estimated, it is normalized to unity by dividing by the sum of the pixel values (Eq. 11). Given this normalization factor represents the relative gain factor between the science and reference images (or a refinement thereof), this is propagated along and used to rescale *only* the science image pixels. We do not modify the gain in the reference image since this entails adjusting its photometric zero-point. The input resampled reference image forms the basis of the difference image calibration (for astrometry and photometry) and we choose to only modify the science image pixels (see below). The operation of modifying the pixels in a science image partition depends on whether the science or reference image was selected for convolution. The image to convolve can be controlled via the *–conv* input parameter, which can equal either "ref", "sci", or "auto". The "ref" and "sci" options force the reference and science image to be convolved respectively. The "auto" option automatically selects the image to convolve according to the relative size of the median FWHM measured from point sources in each image, i.e.,

if($FWHM_{sci} \leq 1.1*FWHM_{ref}$) {
  convolve the science image
} else {
  convolve the reference image
}

The "1.1" factor allows for noise in the derived FWHM measures since we would prefer not to convolve the "noisier" science image unless we really need to, i.e., only if its FWHM was indeed significantly smaller than that in the reference image (which we expect to be rare). The "1.1" was derived empirically from histogramming the FWHM measures from many images and selecting a conservative cutoff to allow for noise.

If the reference image were convolved (or some partition therein), the difference image that minimizes the cost function in Eq. (7) would be:

$$D_{ij} = I_{ij} - dB - K_{sum} \sum_{l} \sum_{m} K_{(norm=1),lm} R_{(i+l)(j+m)}, \qquad (12)$$

where K_{sum} is the normalization or gain factor (Eq. 11) and $K_{(norm=1),lm}$ is the convolution kernel normalized to unity. This is written this way so we can make the gain and offset (*dB*) operate on the science image pixels instead of the convolved reference. By dividing through by K_{sum}, we define a new difference image:

$$D_{ij}^{new} = \frac{D_{ij}}{K_{sum}} = \left[\frac{I_{ij} - dB}{K_{sum}}\right] - \sum_{l} \sum_{m} K_{(norm=1),lm} R_{(i+l)(j+m)}. \qquad (13)$$

The quantity in square brackets is the modified science image from which we subtract the convolved reference image to construct the final difference image. This new difference image is now a minimum for a different cost function (i.e., a rescaled version of Eq. 7), but is still optimal in the least-squares sense.

If however the science image were convolved (or some partition therein) to represent the model image (analogous to Eq. 8), the difference image would be:

$$D_{ij} = \left[dB + K_{sum} \sum_{l} \sum_{m} K_{(norm=1),lm} I_{(i+l)(j+m)}\right] - R_{ij}, \qquad (14)$$

where K_{sum} is the normalization or gain factor (Eq. 11) and $K_{(norm=1),lm}$ is the convolution kernel normalized to unity. This is already in the correct form whereby the *convolved* science image pixels are only modified. The quantity in square brackets becomes the new science image from which we subtract the unmodified reference image.

Some examples from the PiCK method with comparisons to *method 1* are given in Section 7.1. Here's a summary of the pros and cons of the PiCK method compared to PSF-matching *method 1* (HOTPANTS; Section 5.5):

- The PiCK method implicitly corrects for spatially dependent registration residuals (i.e., badly calibrated distortion and/or pointing solutions in the science image) on spatial scales where the convolution kernel was derived. The compensation for registration error comes from the allowed flexibility in kernel shape within its pre-defined footprint.

- The PiCK method also corrects for any spatially dependent multiplicative (gain) factors between the two images on spatial scales where the kernel is derived. These factors are provided by the kernel sum measure. The method in general allows one to refine the relative gain between two images so the difference image photometric zero-point can be tied to the zero-point of the reference image.

- The PiCK method does not require one to specify a parametric model for the kernel, i.e., a linear combination of basis functions. All that is required is that one of the input images contains point sources with effective widths smaller than in the other image. This makes the kernel solution very flexible and general. Pure random input noise will lead to a noisy kernel solution. This will have no effect if the inputs were noise-dominated anyway.

- If the input images are properly regularized beforehand and there is no ambiguity on which image to convolve, the kernel solution will reflect whatever information is contained in the input images (or regions). You'll get what's needed to match the PSFs, like it or not, however unphysical the solution looks. This is in contrast to the model-driven approach (*method 1*) where the chosen parameterization might not apply to a specific image region. I.e., there is less "hoping and guessing" with the PiCK method. This is related to its flexibility mentioned earlier.

- The number of free parameters in the PiCK method depends only on the footprint size (in pixels) assumed for the convolution kernel. Its spatial dependence does not require further parameters if the image region is partitioned judiciously (depending on the expected level of variation and density of point sources to ensure an accurate solution). For PTF, we need to solve for no more than 50 parameters (7×7 kernel pixels and a differential background) per image partition. For comparison, the best-fitting HOTPANTS model requires 252 free parameters (Section 5.5), where the input data are postage stamps of point sources from the entire image. This is not a fair comparison since we need to compare the effective number of degrees-of-freedom, i.e., the number input data elements minus the number of free parameters. For the PTF construction, the number of degrees-of-freedom *per image partition* is probably effectively the same as for HOTPANTS. However, see below for a potential caveat regarding the number of free parameters in the PiCK method for the general case.

- For PTF, the number of free parameters in the PiCK method can be kept at a manageable number (hence ensuring enough degrees-of-freedom) because the PSF sampling is relatively coarse but still at the Nyquist rate at the median seeing. A 7×7 pixel kernel is sufficient to capture the convolution kernel, given that a reference image is expected to have better image quality in general than the observed image. A PSF that is over-sampled by a factor of 5 above Nyquist for example (i.e., 0.2″/pixel versus 1.01″/pixel for PTF when the seeing is ~ 2″ [FWHM]) will require a kernel of at least 35×35 pixels. That's 1225 free parameters! For the next generation of large CCDs, estimating such a kernel will be prohibitively slow unless some clever multi-threading technique is devised.

Another possibility is the use of multicore (e.g., GPU) architectures recently suggested by Hartung et al. (2012) in the context of transient detection using difference imaging.

- A slight caveat of the PiCK method is that kernel solutions may not turn out to be smooth since they are sensitive to the pixel noise relative to the density of point sources in an image (or region thereof). More point sources will be needed if the pixel noise is appreciable. The kernel is also sensitive to the presence of unmasked detector artifacts and cosmic rays. These effects will lead to noise in the convolved image and hence difference image.

- Another caveat related to the previous one is that noisier kernels appear to be more prevalent when derived from image regions with a high source density (see examples in Section 7.1). This leads to high frequency structure (on single-pixel scales) in the difference images at the locations of point sources. We have found that this residual noise does not pose a problem when detecting transient/variable candidates using a matched-filter followed by PSF-fitting to estimate fluxes. Filtering using metrics specific to PSF-fitting (Section 5.9) can avoid most of the spurious detections.

5.7 Image Differencing & Uncertainties

The PSF-matching and image differencing steps produce the following products: two image differences: "*sci – ref*" and "*ref – sci*" with *either* the science or reference image convolved with the PSF-matching kernel; a pixel uncertainty image for both difference images; a mask image for both difference images; and the actual convolved image: "$K \otimes sci$" or "$K \otimes ref$", where we expect the reference image to be convolved most of the time (see Section 3.2). The output mask image propagates bad pixels from both the input science and reference images, and includes the inevitable "expansion" of bad pixels due to the convolution process. The difference image pixel uncertainties are computed by RSS'ing the input image uncertainties with a correction for correlated-noise due to the convolution:

$$\sigma_{diff} = F_c\sqrt{\sigma_{sci}^2 + \sigma_{ref}^2},$$

where σ_{sci} and σ_{ref} were defined in Section 5.4, and the correlated-noise correction factor F_c is approximated from the ratio of the robust background RMS in the difference image to the RSS'd background RMS's in the modified science and reference images (*prior* to convolution):

$$F_c \approx \frac{bckRMS(diff)}{\sqrt{bckRMS(sci)^2 + bckRMS(ref)^2}}.$$

5.8 PSF Determination and PSF-fit Photometry

DAOPhot and its subsidiary PSF-fitting program *Allstar* (Stetson 1987, 1989, 2000) is used for all source extraction and photometry on the difference images. First, Stetson's original Fortran code was slightly modified to handle long input/output paths and filenames. Second, Perl routines were implemented to automate all the steps necessary to generate PSFs and perform photometry on the fly. Sanity checking and automated QA on intermediate outputs was also implemented to ensure no hiccups in the processing flow. User-friendly messages (we hope!) are written to standard output if things go wrong.

Note that PSF generation and subsequent PSF-fit photometry is only performed if the *–psffit* switch is set, and aperture photometry (as a separate process that generates separate products) is only generated if the *–apphot* switch is set (see Section 5.10). The reason for separating the two flavors of photometry is that one may want to run one method and not the other. It is not known yet if both methods are needed. As a further detail, if >85% of the pixels in a difference image are tagged as "bad" (where 85% is currently hard-coded), no PSF generation and no source extraction/photometry whatsoever is performed. *Ptfide* writes a warning to standard output and continues onto the next science image if it exists in the input list, otherwise, it terminates gracefully after the image-differencing step.

The parameters specific to *DAOPhot* and *Allstar* reside in the configuration files specified by *–cfgdao* and *–cfgpht*, with some of the more important parameters therein overridden by the command-line inputs: *–tmaxpsf*, *–tdetpsf*, and *–tmaxdao*. Some *DAOPhot* parameters are dynamically computed within *ptfide* and also override those in the input configuration files. Furthermore, we hardcode some parameters we expect to be relatively static and data-independent. These also override equivalent parameters in the input configuration files. So, why bother with the *DAOPhot* configuration files at all? That's because *DAOPhot* requires them on startup.

Automatic PSF generation is performed by the *createpsf* subroutine in *ptfide*. Currently, the PSF is allowed to spatially vary according to a 2^{nd} order polynomial in x, y. The PSF is always generated off the *resampled* reference image, or the kernel-convolved version if it was indeed convolved. If you recall, the reference image is always expected to be convolved (Section 3.2), so assume this is the norm. The reason for using the (convolved) resampled reference image is because this is expected to have a higher pixel signal-to-noise ratio than the science image. In the end, we want to estimate the PSF off an image whose point-source profiles match those in either difference image (*sci – ref* or *ref – sci*). In theory, since the (modified) input images for the differencing were gain and PSF-matched, and differencing is a linear operation, either input image should suffice for PSF generation.

At the time of writing, we ensure that no more than 200 point sources are automatically picked per image to generate the initial PSF. This is refined in a second iteration by subtracting neighboring sources (affecting the wings) to the PSF-picked stars and then re-estimating the PSF. This second iteration can be rather slow if there are many neighbors since it involves a PSF-fitting step to obtain accurate fluxes and positions for the neighbors. To speed up the PSF-generation, we regulate the number of neighbors by only subtracting the *brightest* 1000 neighbors to all the PSF-picked stars. So for example, for a random distribution of sources and <~ 200 sources picked for the PSF generation, >~ 5 (brightest) sources on average will be subtracted prior to refinement of the PSF. This makes the PSF-generation step relatively fast and robust, and there are always enough stars in a PTF field to yield a reasonably accurate PSF for the purpose of PSF-fit photometry off the difference images.

The default PSF product is stored in the default *DAOPhot* format (output file **_pmtchconvrefdao.psf*) which consists of a look-up table of corrections to a best-fitting Gaussian model. Other basis functions can be selected, but we found a Gaussian to work reasonably well for PTF data. If the debug switch (–d) is set, the *DAOPhot*-formatted PSF file is converted to a FITS image of postage stamps for visualization of the spatial variation (output file **_pmtchconvrefdaopsf.fits*).

Following creation of the PSF, sources are extracted and PSF-fit photometry is performed off each difference image to create astrometrically and photometrically calibrated catalogs (tied to the reference image WCS and ZP). We use *DAOPHot* to perform a single "find" (detection) run and then estimate PSF-fit fluxes using *Allstar* on the initial detection list. The PSF-fitting also returns refined positions for the sources. We do not iteratively subtract the PSF and do another find to uncover hidden sources in the wings of brighter sources, as done in practice. This is because we are extracting sources from a *hopefully* uncrowded difference image where the static confusing signal is expected to be gone. The only culprits are instrumental glitches and residuals where the deblending of real sources from any instrumental artifacts in the simultaneous PSF-fitting is advantageous.

5.9 Filtering of Transient/Variable Candidates

The raw *Allstar* output catalogs from PSF-fit photometry (one for each difference image) are filtered on the source "sharpness" and "chi" metrics using the input thresholds specified by *–tshp* and *–chi*. The goal is to retain sources that are approximately PSF-like since the majority of transients and variables will be point sources. Caution must be exercised here since these metrics will depend on the quality of the image subtractions where a bad subtraction will lead to larger PSF-fit residuals and hence larger "chi" values. The "chi" values will also depend on the internal pixel uncertainty estimates where underestimated uncertainties will also lead to larger "chi" values. Therefore, it is advised to first calibrate the "chi" and "sharpness" filtering thresholds offline to avoid filtering genuine variables in the production run. Additionally, for flexibility, we also filter on the PSF-fit photometric signal-to-noise ratio (input threshold *–tsnr*). We calibrate the PSF-fit instrumental magnitudes according to the difference image ZP (inherited from the reference image ZP) and the filtered extractions are written to two tables in IPAC format: **_pmtchrefmscipsffit.tbl* and **_pmtchscimrefpsffit.tbl*. An example of an output table is shown on page 40 (note: due to its width, the rows have been wrapped around). These tables also contain many other metrics to support the vetting of transient/variable candidates downstream, e.g., for input into a machine-learning algorithm.

5.10 Aperture Photometry

If the *–apphot* switch is set (independently of *–psffit*), we generate aperture photometry tables on both difference images using the six aperture sizes defined in the *–cfgpht* configuration file. Note that these are raw aperture measurements and **not curve-of-growth corrected**. Simple filtering is performed to remove extractions where *all* six aperture measurements are bad (e.g., contain bad pixels or are too close to an edge). Two tables in IPAC format are generated: **_pmtchrefmsciapphot.tbl* and **_pmtchscimrefapphot.tbl*. Note that the aperture photometry is not subject to the same post-filtering as performed on the PSF-fit photometry output using the fitting metrics (Section 5.9). The aperture photometry tables will therefore contain more spurious detections (particularly if the difference-image quality was bad). Furthermore, these tables do not contain any ancillary metrics to support any candidate vetting downstream. We therefore advise using the tables from PSF-fit photometry whenever possible.

Note that even if the aperture photometry step is not explicitly run (by specifying *–apphot*), aperture photometry information for a single aperture size is still propagated to the PSF-fit photometry tables (**_pmtchrefmscipsffit.tbl* and **_pmtchscimrefpsffit.tbl*). The aperture size of interest is specified by the *–apnum* parameter. This represents the line number of the specific aperture listed in the input *–cfgpht* configuration file. These aperture measurements alone could be used to vet candidate extractions on "bad" difference images that contain systematic positive/negative residual source pairs (e.g., due to bad image registration). Depending on the aperture size, the aperture flux is expected to be consistent with zero (within the noise) on a positive/negative residual pair, while a flux excess is expected at the location of a true transient or variable source. This "cancellation" is not possible with PSF-fit photometry since the fit is likely to be biased towards a positive flux residual with little impact from the neighboring negative residual. Hence, PSF-fit photometry has a much stricter requirement for difference image quality.

```
\ Filtered daophot/allstar PSF-fit photometry results
\ Generated by ptfide, 2013-02-15 at 19:53:41
\ For image file: PTF_201205293511_i_p_scie_t082533_u013363434_f02_p004561_c02_pmtchscimref.fits
\ Used [spatially variable] PSF in PTF_201205293511_i_p_scie_t082533_u013363434_f02_p004561_c02_pmtchconvrefdao.psf
\ Number of sources = 65
\ Magnitude Zero-Point (magnitude for 1 DN of signal) = 27
\ Observation Modified Julian Date [OBSMJD] = 56076.35108 days
\ Column definitions:
\ xpos, ypos = source PSF-fit centroid position in x,y system of difference image
\ ra, dec = corresponding J2000 equatorial coordinates
\ magpsf, sigmagpsf = PSF-fit magnitude and 1-sigma uncertainty
\ flxpsf, sigflxpsf = PSF-fit flux and 1-sigma uncertainty corresponding to PSF-fit magnitude
\ magap, sigmagap = aperture magnitude and 1-sigma uncertainty using aperture radius: A4 = 6 pixels;
\                   *not* curve-of-growth corrected; 99.999 +/- 9.9999 => bad!
\ flxap, sigflxap = aperture flux and 1-sigma uncertainty corresponding to aperture magnitude
\ snrpsf = signal-to-noise ratio from PSF-fitting
\ sky = local background value from mode of pixel-flux distribution in annulus
\ nneg = number of negative pixels in a 7 x 7 pixel region centered on source
\        "-999" => measurement not possible, e.g., too close to an edge
\ nbad = number of bad pixels in a 7 x 7 pixel region centered on source
\        "-999" => measurement not possible, e.g., too close to an edge
\ dnear = distance to nearest extraction in reference image
\ magnear, sigmagnear = mag and uncert. of nearest extraction in ref-image (mag_auto); 99.999 +/- 9.9999 => bad!
\ chi = robust estimate of ratio: RMS in PSF-fit residuals / expected RMS according to uncertainties
\ sharp = sqrt[fwhm_obs^2 - fwhm_PSF^2];
\       ~ 0  => perfect (source is PSF-like);
\       >> 0 => extended source or bad kernel solution;
\       << 0 => cosmic ray, glitch, or bad kernel solution
\
|sourceid    |xpos        |ypos        |ra          |dec         |magpsf      |sigmagpsf   |flxpsf      |sigflxpsf   |magap
|sigmagap    |flxap       |sigflxap    |snrpsf      |sky         |nneg        |nbad        |dnear       |magnear     |sigmagnear  |chi
|sharp       |
|i           |r           |r           |r           |r           |r           |r           |r           |r           |r           |r
|r           |r           |r           |r           |i           |i           |r           |r           |r           |r           |r
|
|            |pix         |pix         |deg         |deg         |mag         |mag         |DN          |DN          |mag         |mag
|DN          |DN          |            |DN          |            |            |arcsec      |mag         |mag         |            |
|
 0            338.808      14.815       186.4061082   44.9931146  19.184       0.105             1337.827      129.384 99.999       9.9999
0.000        0.000   10.34       -3.073       14          6                 3.331     14.1306       0.0078     2.096        -0.098
 1            978.12       22.578       186.6602813   44.9928877  20.902       0.2542             274.916       64.383 20.489       0.4874
402.161      180.535    4.27       -2.109        24          0                 1.517     19.6136       0.0222     1.12         -0.938
 2            1559.598     69.425       186.8915612   44.9809621  18.749       0.152             1997.101      279.706 19.114       0.14
1426.921     183.994    7.14       -3.246        14          0                 21.094     18.8023       0.0105     4.261        -0.798
 3            1920.819     117.017      187.0352253   44.9681254  19.983       0.2162             640.914      127.672 20.927       0.7662
268.658      189.591    5.02       -1.426        25          0                 0.649     15.2644       0.0072     2.268        -0.555
 4            1683.753     164.994      186.9411054   44.9543159  20.939       0.209              265.705       51.196 20.839       0.6856
291.340      183.970    5.19       0.562         20          0                 13.900     20.9311       0.0373     0.839        -0.47
 5            1557.925     423.886      186.8916914   44.8813790  20.923       0.2553             269.650       63.447 21.606       1.363
143.747      180.456    4.25       0.073         19          0                 20.256     21.0471       0.0590     1.102        -0.575
 6            1589.966     457.948      186.9044823   44.8718634  21.037       0.287              242.773       64.226 22.157       2.2588
86.537       180.034    3.78       -0.068        24          0                 34.242     21.1572       0.0537     1.118        -0.878
 7            96.983       492.732      186.3123364   44.8579785  21.135       0.27               221.820       55.179 20.618       0.5548
357.108      182.479    4.02       1.377         22          0                 23.709     16.0410       0.0121     0.916        -1.168
```

6 QUALITY ASSURANCE

ptfide generates a number of metrics to assess the quality of difference images. These are written to two places, depending on their level of importance: a special QA text file and standout output during program execution. Additional metrics and diagnostic files are generated if the debug switch (–d) is specified. We summarize these below.

6.1 Output QA Diagnostics File

To assess the quality of the image-differencing and candidate extractions in general for a given input science frame, a text file containing various statistics is generated if the –qa switch is specified. This file is generically named `<sci_img_rootfilename>_diffqa.txt`. An example its contents is below. These metrics can be used as input to a machine-learning algorithm for vetting transient/variable source candidates.

```
Metrics off "sci - ref" diff image within slice "40:2007,40:4055" before PSF-matching:
Before psfmatch: num good pix = 7885268 pixels
Before psfmatch: num bad pix = 18220 pixels
Before psfmatch: median level = -3.0517578125e-05 DN
Before psfmatch: mean level = -0.0272447639626065 DN
Before psfmatch: median of squared diffs = 117.63060021773 DN^2
Before psfmatch: mean of squared diffs = 373.840593327112 DN^2
Before psfmatch: chi-square from median = 0.494687110185623
Before psfmatch: chi-square from mean = 1.21848905294679

Metrics off "sci - ref" diff image within slice "40:2007,40:4055" after PSF-matching:
After psfmatch: num good pix = 7885274 pixels
After psfmatch: num bad pix = 18214 pixels
After psfmatch: median level = -0.0155108943581581 DN
After psfmatch: mean level = -0.00643663290271417 DN
After psfmatch: median of squared diffs = 124.12934581683 DN^2
After psfmatch: mean of squared diffs = 306.488866985189 DN^2
After psfmatch: chi-square from median = 0.446775734424591
After psfmatch: chi-square from mean = 1.01037284523972

Following pertain to images after gain and background matching, but before PSF-
matching:
Background in science-image (mode of good pixels) = 336.9013671875 DN
Background in reference-image (mode of good pixels after resampling) = 338.892608 DN
Robust sigma/pixel in science-image = 15.070813827441 DN
Robust sigma/pixel in reference-image (after resampling) = 2.65715035096648 DN
Magnitude limit (5-sigma) in science-image = 21.2157008902342 mag
Magnitude limit (5-sigma) in reference-image = 23.089198895595 mag

Seeing (approx. point source FWHM) of raw input science image = 1.831683 pixels
Seeing (approx. point source FWHM) of raw input reference image = 1.85 pixels
Seeing (approx. point source FWHM) of reference image after *possible* convolution =
2.0421 pixels

In following, "after filtering" refers to internal thresholding on the "chi", "sharp",
and "snr" metrics from psf-fit photometry:
Number of candidates extracted from sci minus ref image with no filtering = 520
Number of candidates extracted from sci minus ref image after filtering = 65
Number of candidates extracted from ref minus sci image with no filtering = 415
Number of candidates extracted from ref minus sci image after filtering = 33
```

The before/after PSF-matching statistics at the top are computed within a rectangular region on the difference-images with coordinate ranges specified by the –qas input. If –qas is omitted, the whole image is used. The –qas specification allows the edges of an image to be excluded. Statistics are computed *before* and *after* the PSF-matching step. By "before", the input science and reference images would have only been gain and background matched, and astrometrically aligned. Therefore, all we are comparing here is the quality of the PSF-matching. Most of the contents are self-explanatory except perhaps for the chi-square metrics. These are defined as

$$\chi^2 = \left\langle \frac{D_i^2}{\sigma_{Di}^2} \right\rangle_i , \qquad (15)$$

where the angled brackets imply either a median or mean computed over all *good* pixels *i* within the image slice. D_i are the pixel differences and σ_{Di} their uncertainty. This quantity is more akin to a "reduced chi-square" metric where values closer to one imply that fluctuations in the difference image are more-or-less consistent with that expected from random fluctuations alone. Large values indicate problems, e.g., an excess of instrumental (systematic) residuals or bad PSF-matching.

When executing *ptfide* on many program images, one can examine the change in one or more of the metrics in the `*_diffqa.txt file` before and after PSF-matching (e.g., "`median of squared diffs`" and/or "`chi-square from mean`"). These changes can be used to calibrate thresholds for rejecting bad difference images in general. We have rarely come across a difference image that looked "perfect" *without* PSF-matching. So, the *change* in these metrics due to PSF-matching alone is very useful.

6.2 Other Diagnostic Information

Additional metrics are written to standard output by default, as well as to debug files if the –d switch is specified. These are as follows:

- To check the accuracy of the relative gain refinement or photometric calibration of the science image using the reference image extractions, the following line in the standard output will tell you:

  ```
  Stored 495 rows from <sci_img_rootfilename>_sx.tbl
  Dg = 0.5886 +/- 0.0015 (= 0.5753 mag); 5-95% range = 0.1221
  ```

  The first line shows the number of extractions that were matched between the science and (internally pre-filtered) reference image catalog tables. The second line shows the median of the ratio of science to reference image source fluxes (*Dg*) and its uncertainty. This value is later used to rescale the science image pixels so its ZP can match that of the reference image. We also report the $5^{th} – 95^{th}$ percentile range in this flux ratio. A relatively large value implies a large scatter and possibly an inaccurate estimate for *Dg*. This may result from significant spatial variations in the throughput, bad flat-fielding, or bad source matching due to bad astrometric calibration. Note that if the input science image header already contains an accurate ZP value, *Dg* above should be close to 1. If a

ZP is not present or is way off, *ptfide* will report a relatively large *Dg* as in the above example.

- If the debug switch (–d) is specified, we generate an image of the chi-square metric defined by Eq. (15) for the *entire* difference image *after* PSF-matching. This is done by boxcar-averaging over pixels *i* within 8 x 8 pixel bins (i.e., locally averaging the quantity in angled brackets in Eq. 15 over 64 pixel blocks). An example is shown in Figure 2. This image file is generically named:
  `<sci_img_rootfilename>_pmtchdiffchisq.fits`
  Apart from the sensitivity of χ^2 to instrumental residuals, large local values in this image may also indicate underestimated Poisson noise in the error model and/or the actual presence of variable sources.

- Both HOTPANTS (*–pmeth* 1) and the PiCK method (*–pmeth* 2) compute a convolution kernel sum metric (KSUM) which represents the overall relative flux ratio between fitted point sources in the science and *post-convolved* reference image (e.g., Eq. 11). It is related to the image ZPs as follows: $2.5\log_{10}(\text{KSUM}) = \text{ZP}_{sci} - \text{ZP}_{ref}$. KSUM can be obtained from the following example lines in the standard output if HOTPANTS was run:

```
Sum Kernel at 0,0: 1.016868
Sum Kernel at 2047,4095: 1.016868
```

  It is also written to the header of the difference images as:

```
KSUM00  = '1.0169  '           / Kernel Sum
```

  Values that significantly deviate from 1 indicate a problem with either the relative gain calibration between science and reference images performed upstream, or with determination of the convolution kernel itself in HOTPANTS. Here, the residual is ~1.7%, which is tolerable given the accuracy of our photometric calibrations in general. If the PiCK method was executed (*–pmeth* 2), KSUM can be obtained from the following example lines in the standard output:

```
solvekern: Raw convolution kernel sum for partition 0 = 1.035901
solvekern: Raw convolution kernel sum for partition 1 = 1.039264
solvekern: Raw convolution kernel sum for partition 2 = 1.032326
etc...
```

- If the debug switch (–d) was specified, one can overlay and examine the candidate extractions onto their respective difference images using the following DS9 region files:

  If the –psffit (PSF-fit photometry) switch was specified:

```
<sci_img_rootfilename>_pmtchscimrefapphot.reg
<sci_img_rootfilename>_pmtchrefmsciapphot.reg
```

  If the –apphot (aperture photometry) switch was specified:

```
<sci_img_rootfilename>_pmtchscimrefpsffit.reg
<sci_img_rootfilename>_pmtchrefmscipsffit.reg
```

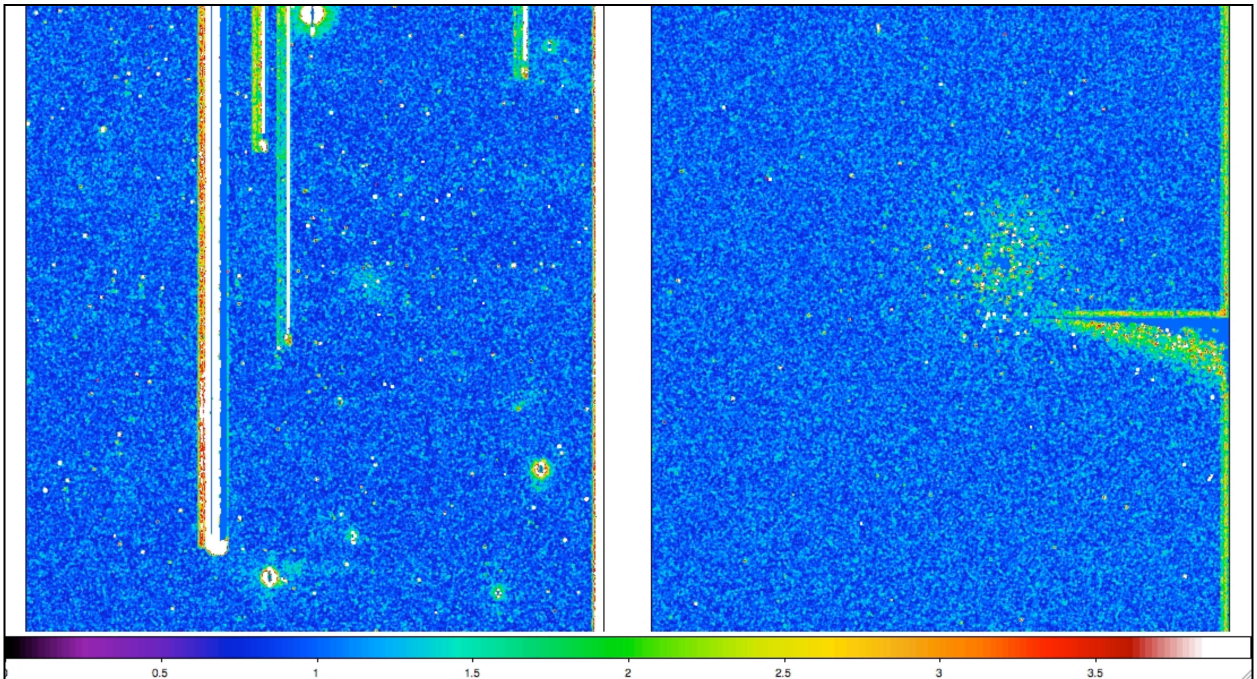


Figure 2 – Example "pseudo reduced chi-square" (χ^2) images represented by Eq. (15) computed using a boxcar-average over pixels *i* within 8 x 8 pixel bins. *Left*: M33 and *right*: M13 where zoomed-in difference images are shown in Figures 3 and 6 respectively. The expected χ^2 value in regions with pure random noise and static signal is 1 (shown by the color bar). See text for details.

7 TESTING AND OPTIMIZATION

A set of regression tests were constructed using archived science images from the project going back to early 2011. We focused on PTF fields containing the M33 galaxy and M13 globular cluster, which we hope provide the ultimate stress test for *ptfide*. Reference images (co-adds) and accompanying source catalogs were constructed offline using ~20 science images with reasonably good seeing (<~ 1.65″) and astrometry. In production, specially pre-processed science frames will be used for the difference-imaging pipeline, but a sufficient number of these were not available at the time of writing.

For those with access to the PTF operations environment at IPAC, two Regression Test Baselines (RTBs) were constructed using several archived program images for the M33 and M13 targets. The products from these runs (including the example script used to call *ptfide*: "*rundiff*" and all input files) are located under:

```
/stage/ptf_eng_users_frank/RTBdiff/M13/
/stage/ptf_eng_users_frank/RTBdiff/M33/
```

The science images used here were already photometrically calibrated (with a ZP in their FITS header). Therefore, *ptfide* simply refined the existing ZP relative to calibrated sources in the input reference image catalogs. In production, the science image inputs will not have been photometrically calibrated, in which case *ptfide* will automatically perform an absolute photometric calibration using the reference image extractions to derive an image ZP. From a few hundred science images selected at random for the M33 and M13 fields, *ptfide* applied relative ZP corrections of no more than ~2%. Furthermore, for the same set of images, *ptfide* computed astrometric shifts (along either *x* or *y*) of less than a tenth of a pixel relative to the reference image, indicating the archived science images are reasonably well calibrated.

For PTF science images in general, we found that the relative photometric gain between a science and matching reference image needs to be better than ~1.5% to obtain a good difference image (over the observed detector dynamic range) following PSF-matching. The relative astrometry needs to be better than ~0.1″ for a typical (median) seeing of FWHM ~ 2.5″ where the PSF is sampled by a factor of ~1.25 above Nyquist.

The *ptfide* parameters were optimized with one goal in mind: to minimize systematic residuals in difference images with respect to random noise fluctuations from the detector and photon noise in the input images at the locations of non-variable sources. Following the gain and astrometric refinement steps, the most important parameters that determined the difference-image quality were those used to derive the PSF-matching kernel. Below we only describe our optimization procedure for PSF-matching *method 1* (HOTPANTS; specified by *–pmeth* 1; Section 5.5) since this method is very parameter driven, while *method 2* (PiCK; *–pmeth* 2; Section 5.6) is mostly empirically driven, depending primarily on the quality of image inputs and accuracy of the regularization steps.

The parameters for PSF-matching *method 1* in *ptfide* (HOTPANTS) are those specified by "–*hot1*" [*default*: 8, 14, 16, 32, 2, 2] and "*–hot2*" [*default*: 4, 4, 0.40625, 4, 0.65, 2, 1.04, 2, 1.664]. See Sections 5.5 for a description. After fixing the more "trivial" parameters in "*–hot1*": primarily the size of modeled kernel; extracted source stamp size; number of partitions along *x* and *y* for fitting the spatial variation in the kernel, and the polynomial order for this variation, the "*–hot2*" parameters specifying the shape of the basis model functions were then tuned.

First, we optimized the "*–hot2*" parameters using a single program image of the M13 field that had a good photometric and astrometric calibration, and with seeing (FWHM) equal to the median of ~2.5″. This avoided biasing the parameters towards images with highly discrepant seeing values. We constructed a test grid of parameters consisting of σ_1 and β (see Eq. 3 in Section 5.5) that spanned physically plausible ranges. We also assumed two fixed sets of polynomial orders for the basis functions in Eq. (2): "6,4,2" and "4,4,2,2", i.e., a three- and four-component model respectively. The latter were motivated by the detailed simulation work of Israel et al. (2006). We executed HOTPANTS for each set of polynomial orders and plausible range of (σ_1, β) to create difference images, and then computed the RMS within regions containing no bad (or saturated) pixels. Contour plots of this RMS for the two sets of polynomial orders are shown in Figure 3.

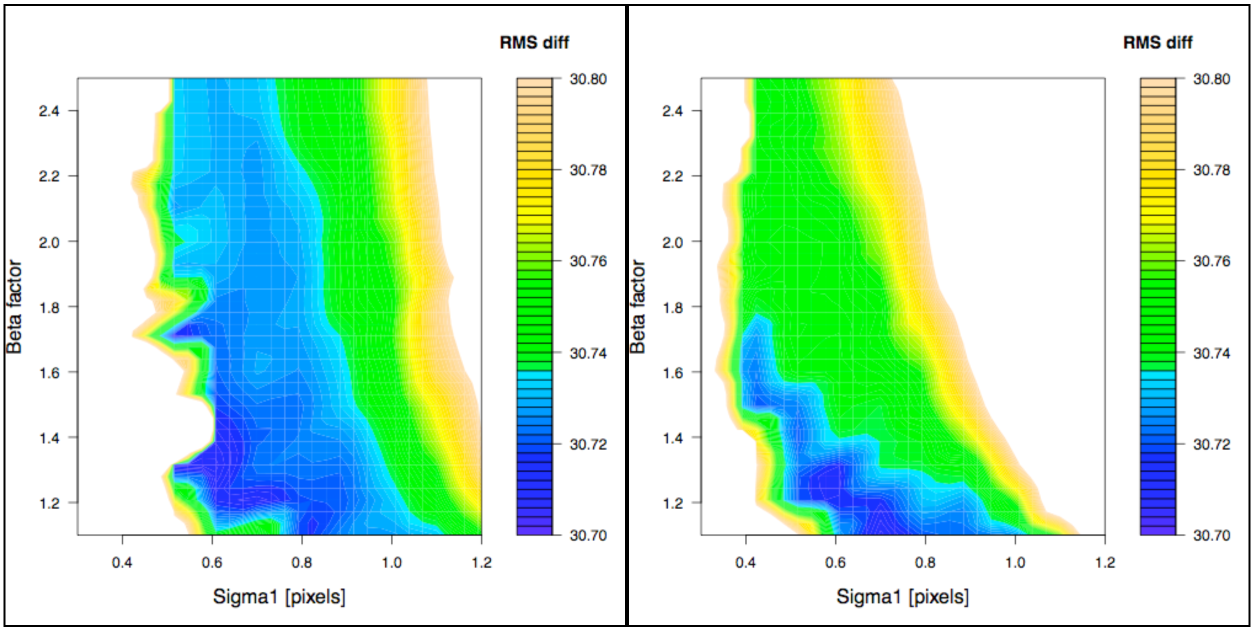


Figure 3 – Contour plots of the difference-image RMS as a function of the kernel model parameters σ_1 and β for two basis polynomial orders: 6,4,2 (*left*) and 4,4,2,2 (*right*). The color bars indicate the range in RMS values [ADU]. See text for details.

It is apparent that the "6, 4, 2" model (left panel in Figure 3) has a "broader minimum", therefore admitting a larger range of model parameters giving approximately the same difference-image RMS. Such a model therefore has more margin for error in the assumed kernel width parameters. Our initial idea was to calibrate these widths using a single representative program image (as here) and then use the dynamic rescaling option in *ptfide* to rescale these by adapting to the actual seeing (FWHM) in the input science and reference images (see Section 5.5). However, we discovered that this option does not work well when exercised on many science images. The additional degrees-of-freedom from the possibly inaccurate input FWHM estimates causes the difference-image RMS to float in an erratic manner, compared to the case where constant kernel model widths are assumed. This is shown in Figure 4 where we computed the difference in mean-square difference-image pixel values (in a selected region) between runs that used a static kernel model (*–hot2* "3, 6, 0.65, 4, 1.04, 2, 1.664") and one where the kernel widths (0.65, 1.04, 1.664) were dynamically rescaled. The static assumption gives the lowest difference image variance overall – our ultimate goal.

Other polynomial orders were also tested and in the end, we decided to adopt a *static* kernel model with polynomial orders "4, 4, 2, 2" and respective widths 0.40625, 0.65, 1.04, 1.664 pixels ($\Rightarrow \sigma_1 = 0.40625$ pixels and $\beta = 1.6$ in the language of Figure 3 or Eq. 3). This model came very close to the "6, 4, 2" three-component model mentioned above, but a four-component model offers more flexibility in the shape of the kernel that can be modeled, i.e., its additional narrower component with $\sigma = 0.40625$ pixels is expected to handle cases where the PSF has significant variation on this scale. In general, we find that the four-component model is relatively insensitive to the typical observed range of seeing values: ~1.3″ to ~4.5″ (FWHM).

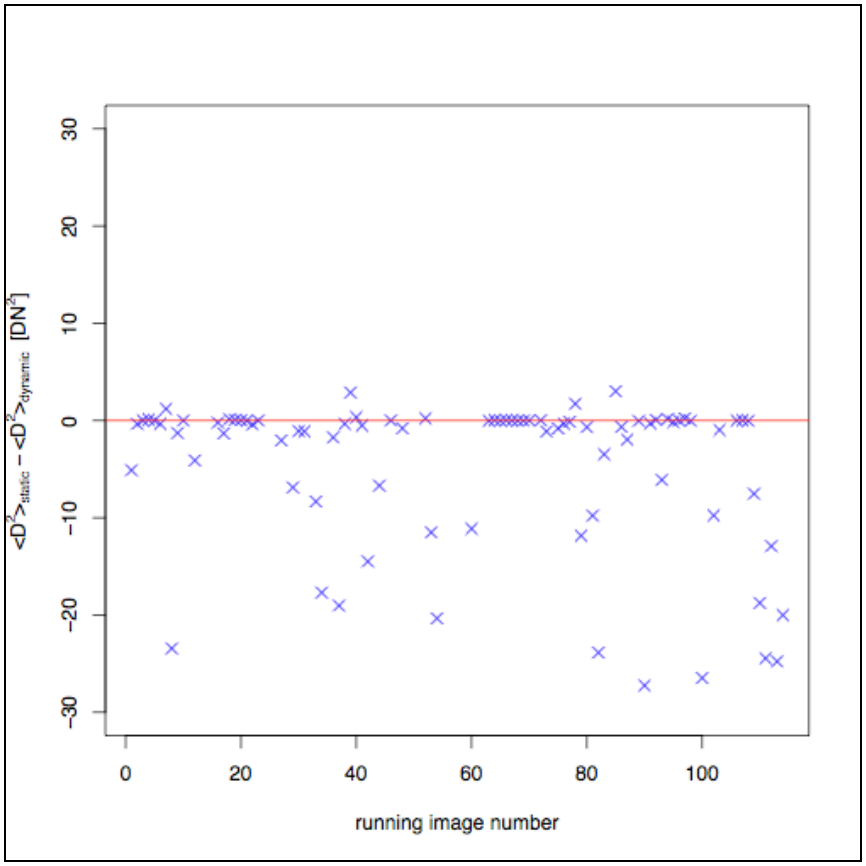


Figure 4 – Difference in mean-square difference-image pixel values (in a region free of bad-pixels) between a model where the kernel widths are assumed *static* and one where they are *dynamically* rescaled according to the prescription in Section 5.5. This comparison was performed for 114 science images of the M13 field (time-ordered running image number is on the horizontal axis). See text for details.

We also repeated this optimization exercise using the Difference Image Analysis pipeline tool (DIAPL) of Wozniak (2000). We obtained similar results since the underlying methodology is the same as that in HOTPANTS, i.e., both implement the Alard (2000) algorithm. In the end, we chose HOTPANTS for PSF-matching *method 1* due to its flexibility in deciding which image to convolve, speed, use of masks and uncertainties, and generation of QA metrics.

7.1 Example Difference Images: HOTPANTS versus PiCK

Figures 7 – 12 show examples of difference images from testing on PTF fields containing the M13 globular cluster and M33 galaxy, with reference images for the same regions shown in Figures 5 and 6. Program images from several observation epochs were used. PSF-matching methods 1 and 2 (Sections 5.5 and 5.6 respectively) are compared using the best parameters available at the time of writing. Also indicated are some previously discovered variable sources from the literature (note that our search for *real* variables was not exhaustive). We'll let you be the judge for each method from this handful of examples. Examining the results from larger test runs (available upon request), the author is slightly inclined towards *method 2* for production.

We have made no quantitative assessment of the completeness and reliability of the detections, although we expect the reliability to be quite good. The dark blotches in Figures 5 – 12 represent bad (mostly saturated) pixels which "grow" in the difference images due to (i) re-sampling of the reference onto the science frame using the Lanczos kernel (effectively a *sinc* kernel with truncated tails) and (ii) the convolution process when matching science and reference image PSFs. No doubt, this will impact the completeness of detections in highly saturated regions, as shown in Figures 7 and 8 for the core of the M13 globular cluster.

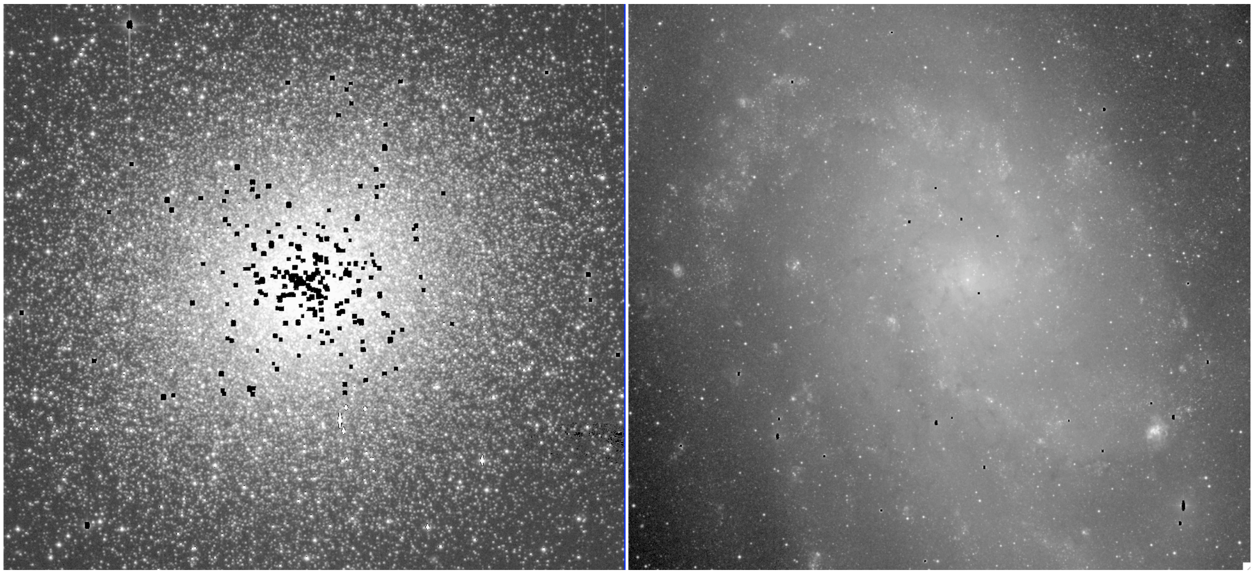


Figure 5 – Resampled reference images for a central 0.25°× 0.25° region on the M13 globular cluster (*left*) and a 0.5°× 0.5° region on the M33 galaxy (*right*). These are used to create the difference images in Figures 7 and 8 (for M13), and Figure 9 (for M33). Dark blotches represent bad/saturated pixels.

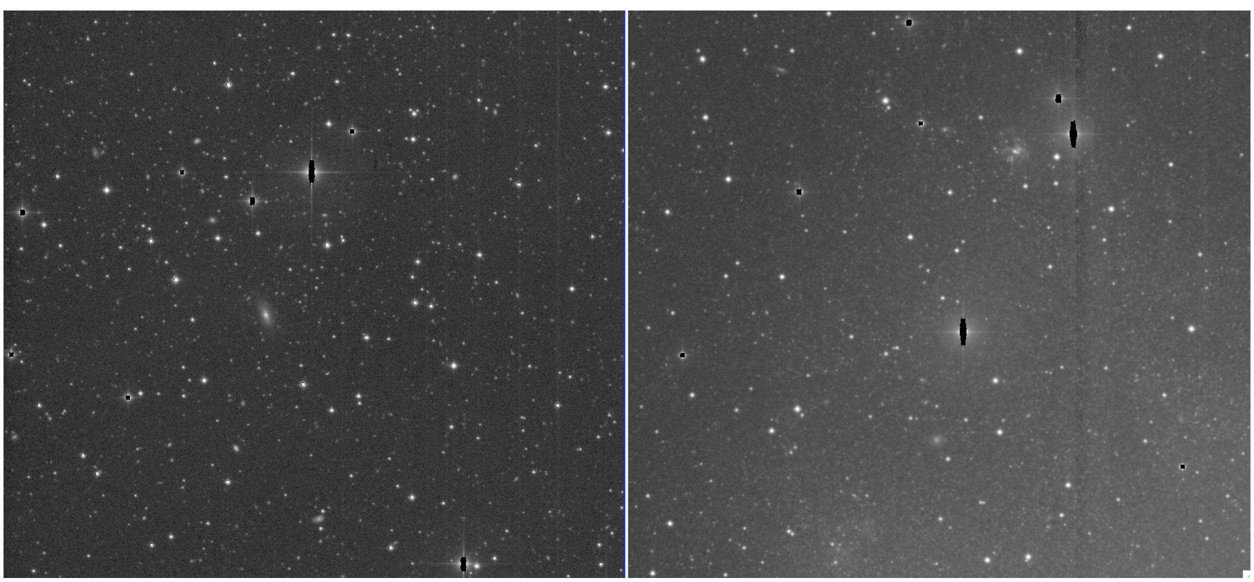


Figure 6 – Resampled reference images for other regions in the M13 globular cluster field (*left*) and the M33 galaxy field (*right*). Both regions are 0.25°× 0.25°. These are used to create the difference images in Figure 10 (for M13), and Figures 11 and 12 (for M33). Dark blotches represent bad/saturated pixels.

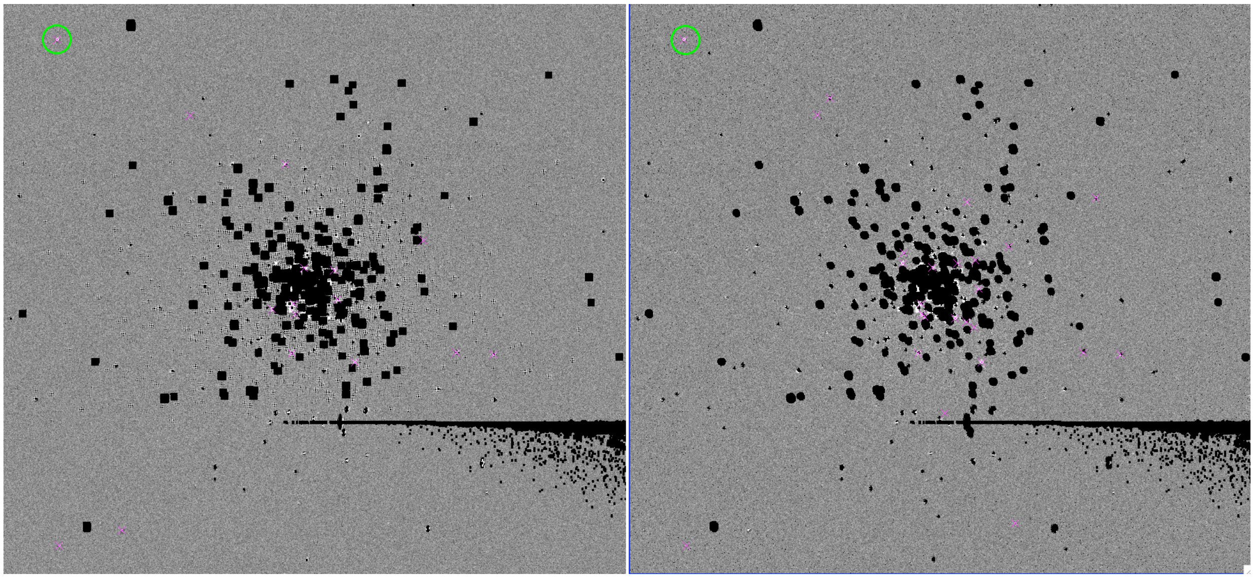


Figure 7 – M13 epoch 1 (science ‘minus’ reference image); *left*: PSF-matching method 2 (the PiCK method); *right*: PSF-matching method 1 (HOTPANTS). The reference image is shown in Figure 5 (left). Green circles are known variables from Kopacki et al. (2003) and magenta crosses are candidates extracted by *ptfide*. Dark blotches represent bad/saturated pixels. See text for details.

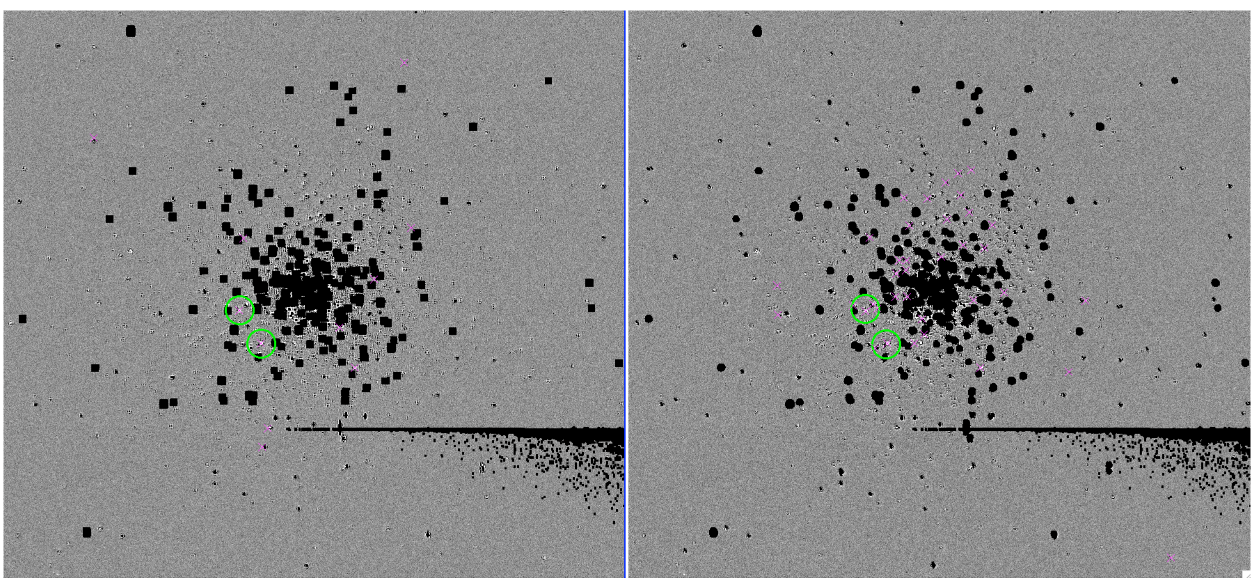


Figure 8 – M13 epoch 2 (science ‘minus’ reference image); *left*: PSF-matching method 2 (the PiCK method); *right*: PSF-matching method 1 (HOTPANTS). The reference image is shown in Figure 5 (left). Green circles are known variables from Kopacki et al. (2003) and magenta crosses are candidates extracted by *ptfide*. Dark blotches represent bad/saturated pixels. See text for details.

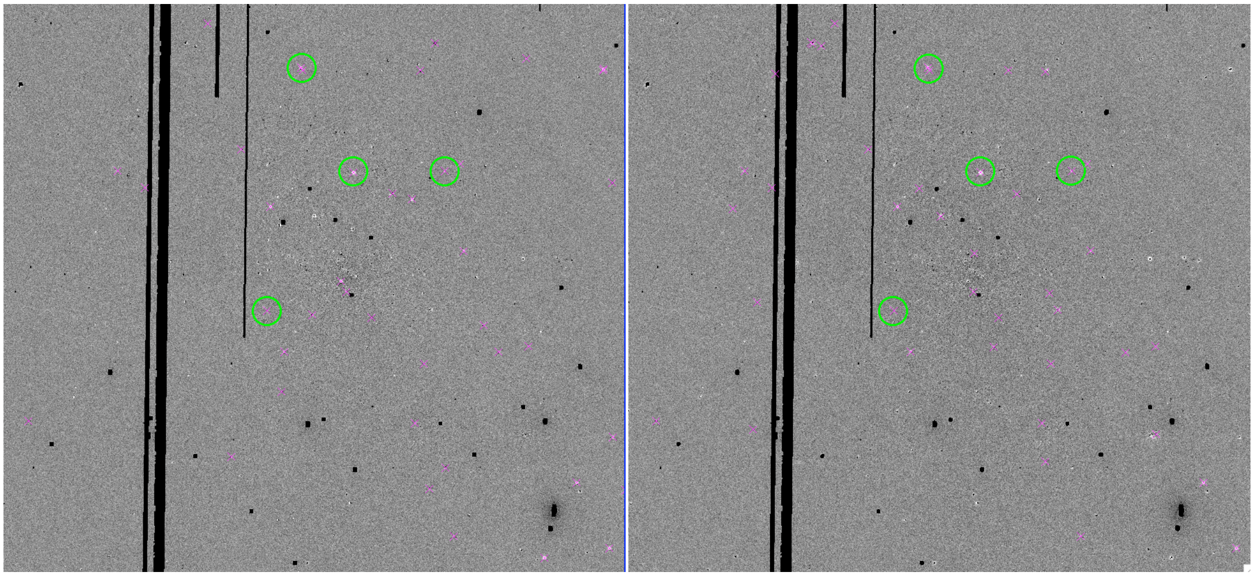


Figure 9 – M33 central region (science 'minus' reference image); *left*: PSF-matching method 2 (the PiCK method); *right*: PSF-matching method 1 (HOTPANTS). The reference image is shown in Figure 5 (right). Green circles are known variables from Hartman et al. (2006) and magenta crosses are candidates extracted by *ptfide*. Dark blotches represent bad/saturated pixels. See text for details.

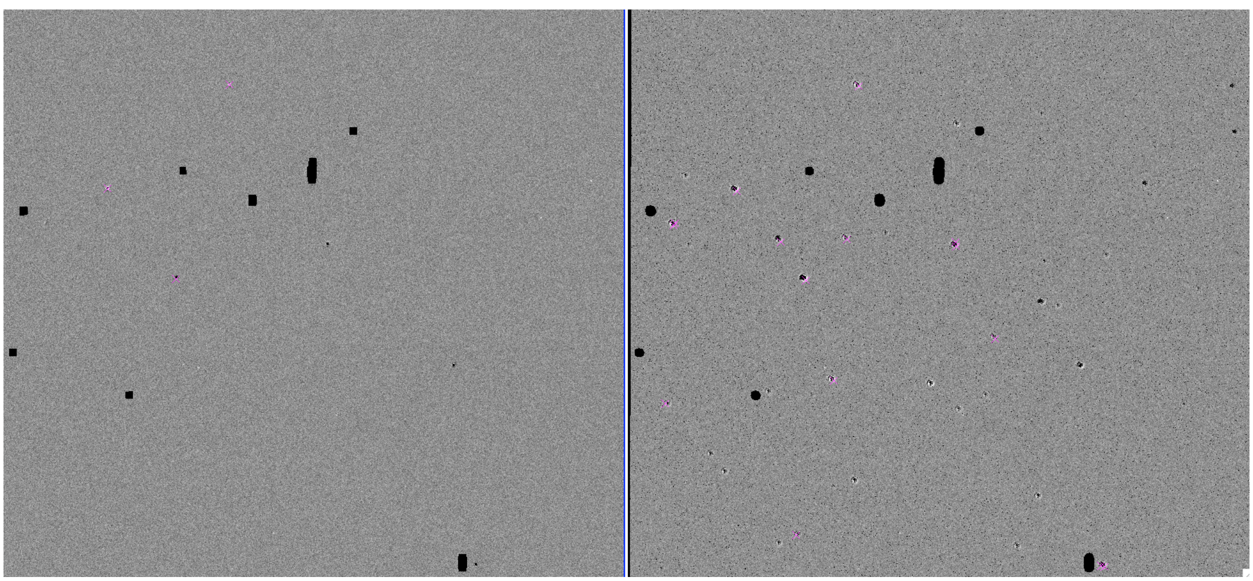


Figure 10 – Another region associated with the M13 field (science 'minus' reference image); *left*: PSF-matching method 2 (the PiCK method); *right*: PSF-matching method 1 (HOTPANTS). The reference image is shown in Figure 6 (left). Green circles are known variables from Kopacki et al. (2003) and magenta crosses are candidates extracted by *ptfide*. Dark blotches represent bad/saturated pixels. See text for details.

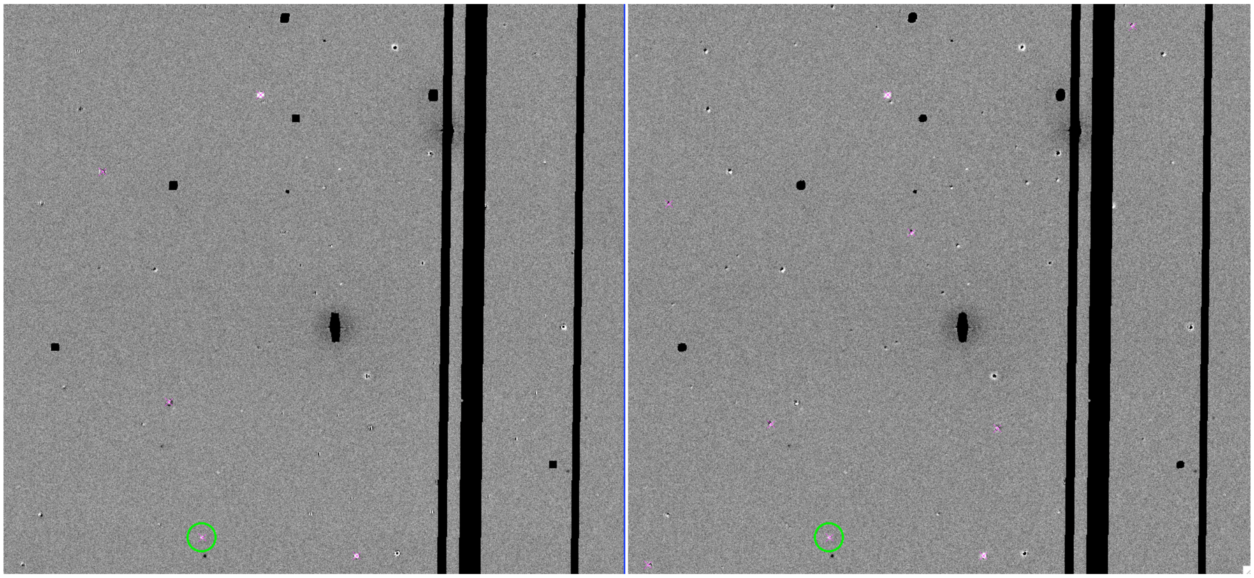


Figure 11 – Another region associated with the M33 field (science 'minus' reference image); *left*: PSF-matching method 2 (the PiCK method); *right*: PSF-matching method 1 (HOTPANTS). The reference image is shown in Figure 6 (right). Green circles are known variables from Hartman et al. (2006) and magenta crosses are candidates extracted by *ptfide*. Dark blotches represent bad/saturated pixels. See text for details.

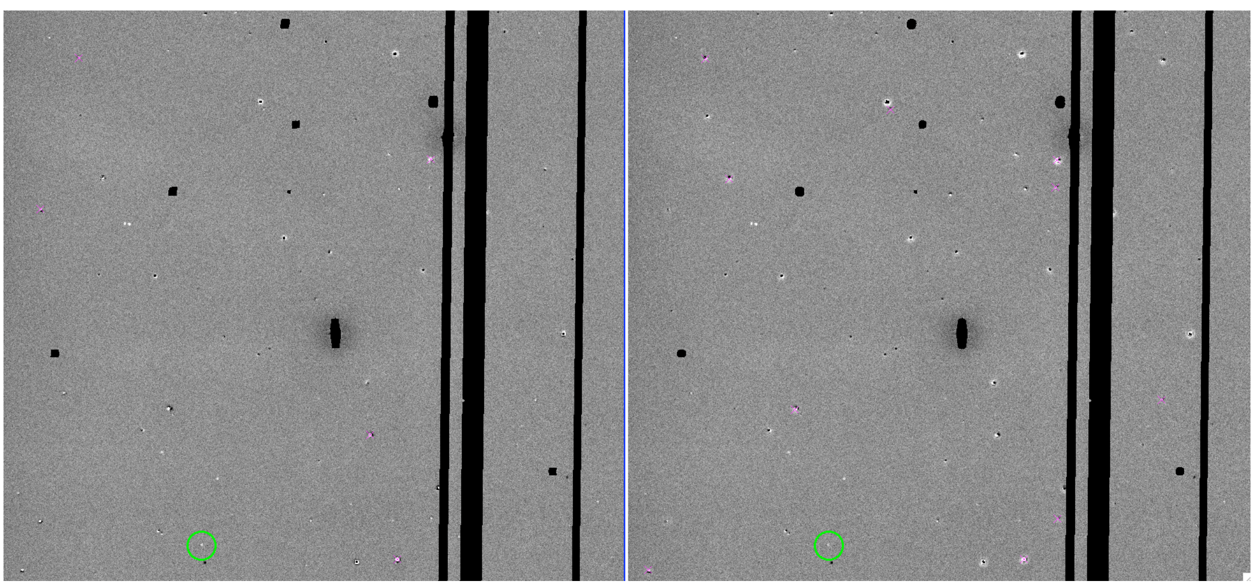


Figure 12 – Another region associated with the M33 field (science 'minus' reference image); *left*: PSF-matching method 2 (the PiCK method); *right*: PSF-matching method 1 (HOTPANTS). The reference image is shown in Figure 6 (right). Green circles are known variables from Hartman et al. (2006) and magenta crosses are candidates extracted by *ptfide*. Dark blotches represent bad/saturated pixels. See text for details.

8 WHAT CAUSES A "BAD" DIFFERENCE IMAGE?

From the testing done to date, below are some of the known causes for bad difference images in general. Here, "bad" is defined in a relative sense whereby the difference image contains residuals (unassociated with genuine flux variability) that significantly exceed the overall random noise fluctuations in the background and the additional (RSS'd) Poisson components from both input images at the location of non-variable sources.

- Inaccurate flat-field (relative pixel-to-pixel responsivity) calibration upstream. This will impact the relative gain calibration (or refinement) in *ptfide* since it assumes a single global ZP estimate with no spatial variation. See also the next bullet point.

- Spatially varying photometric throughput in the science frame, e.g., brought about by varying transparency across the field during an exposure. This will also impact the relative gain calibration.

- The science image has a bad astrometric solution, which includes a bad distortion calibration. Note that *ptfide* internally refines the relative registration *only* along the orthogonal *x-y* axes before resampling the reference image. This could indeed fail if there are an insufficient number of source matches. However, if the PV-distortion coefficients for the science image are inaccurate to start with, the reference image sources will be offset from their science image counterparts, therefore leading to large residuals when differenced.

- The model convolution kernel for matching PSFs in the science and reference image may not be parameterized to high enough order to "fit-out" all the systematic residuals seen at all spatial scales. I.e., the default assumed parameterization is not generic enough (even though efforts were made to ensure this) and there are bound to be images that need their own "special" model.

- Science images that are under-sampled (e.g., acquired under really good seeing combined with the relatively large pixel sizes of the PTF detectors) will generally be difficult to match to a reference image that presumably was also created from frames with the best seeing. A good PSF-matching kernel model will be difficult to estimate and one will have to live with an excess of systematic residuals in the difference image.

In summary, the best way to diagnose the cause for a bad difference image is to compare it to a "good" difference image and plainly figure out what's different between the two, e.g., seeing, accuracy of distortion calibration, astrometric quality, and/or gain (ZP) relative to the reference image. This is easier said than done! It's probably best to work through the list above to hypothesize the probable cause(s).

9 LIENS, POTENTIAL UPGRADES, and TBDs

- Consider applying a differential distortion correction to the science frame relative to the reference image using the input reference image extractions (e.g., via *scamp*)? Currently, the assumption is that the science image PV-distortion coefficients are accurate enough for resampling ("*swarping*") the reference image, and we only refine the registration along the orthogonal *x-y* axes before resampling. This is probably only needed for PSF-matching *method 1* (HOTPANTS), not *method 2* (PiCK) since the latter implicitly corrects for spatially dependent registration residuals.

- Implement better metrics to assess the quality of the PSF that's currently made on the fly to support PSF-fit photometry. Should we attempt to make a crude backup PSF based on the effective observed FWHM (after PSF-matching) in case the auto-generated PSF fails or is grossly wrong according to some metric? Currently, *ptfide* aborts if *DAOPhot* cannot make a PSF due to e.g., an insufficient number of "good" stars. Testing has shown this to be rare for PTF.

- Perform a thorough completeness and reliability analysis using a large number of difference-images containing known variables. This will be used to tune extraction thresholds, source-shape filtering thresholds, and pixel-masking parameters.

- For PSF-matching *method 2* (the PiCK method), should we *always* allow for automatic selection of the image to convolve (e.g., for production)? This can be specified by setting "*–conv* auto". Currently this is set as "*–conv* ref", i.e., the reference image will always be convolved since as outlined in Section 3.2, it is assumed to have been created from the best seeing frames available and therefore its convolution will be justified most of the time during a survey. The "*–conv* auto" can give erroneous results if the reference image was not created from the best seeing frames since the automatic selection is based on the relative FWHM measure and is subject to error.

- Optimize PSF-matching *method 2* (PiCK method) for cases when bright/extended emission is present in an image partition. Currently, this manifests itself as a high or low differential gain (kernel sum) estimate for the partition, leading to a bad convolution kernel. One improvement would be to threshold the kernel sum estimate relative to e.g., the median of all other "good" partitions, and if found to be too discrepant, adopt the kernel solution from an adjacent "good" partition. A similar method may also be used for partitions containing a high source density where currently the kernel solution is slightly noisier.

- *Low priority*: need better flux-uncertainty estimation from *DAOPhot*/*Allstar* photometry on the difference images. Currently, we use approximations based on the rescaled electronic gain and background noise using only the difference image (not the input components). We suspect the overall Poisson noise is being slightly underestimated in general. Better estimates will require us to dissect *DAOPhot* and recode its noise model.

10 REFERENCES

Alard, C. & Lupton, R. H., 1998, ApJ, 503, 325

Alard, C., 2000, A&A, 144, 363

Becker, A., 2009, HOTPANTS: *http://www.astro.washington.edu/users/becker/hotpants.html*

Bertin, E., 2009, SExtractor: *http://www.astromatic.net/software/sextractor*

Bertin, E., 2010, SWarp: *http://www.astromatic.net/software/swarp*

Bramich, D. M., 2008, MNRAS, 386, L77

Hartman, J. D., et al., 2006, MNRAS, 371, 1405

Hartung, S., et al., 2012, IEEE-ICIP (2012), arXiv:1209.5823

Israel, H., Hessman, F. V. & Schuh, S., 2006, Astronomische Nachrichten, 328, 16

Kopacki, G., et al., 2003, A&A, 398, 541

Laher, R. L., et al. 2013, PASP, *in preparation*

Law, N. M., et al., 2009, PASP, 121, 1395

Ofek, E. O., et al., 2012, PASP, 124, 62

Stetson, P. B., 1987, PASP, 99, 191

Stetson, P. B., 1989, lectures on stellar photometry:
http://ned.ipac.caltech.edu/level5/Stetson/Stetson_contents.html

Stetson, P. B., 2000, DAOPHOT II/AllStar manual: *http://www.astro.wisc.edu/sirtf/daophot2.pdf*

Wozniak, P. R., 2000 (DIAPL difference image analysis tool), Acta Astronomica, 50, 421

11 ACRONYMS

ADU	Analog Digital Units
CCD	Charge Coupled Device
DEC or Dec	Declination
DAOPhot	Dominion Astrophysical Obervatory’s Photometry tool
DIAPL	Difference Image Analysis PipeLine
DN	Data Number

DS9	SAOImage Deep Space 9 image visualization tool
FITS	Flexible Image Transport System
FOV	Field-Of-View
FWHM	Full Width at Half Maximum
GPU	Graphics Processing Unit
HOTPANTS	Higher Order Transform of PSF ANd Template Subtraction
I/O	Input / Output
IPAC	Infrared Processing and Analysis Center
M13	Messier 13 (the Hercules Globular Cluster)
M33	Messier 33 (the Triangulum Galaxy)
MAD	Median Absolute Deviation
NaN	Not-a-Number
PDL	Perl Data Language
PiCK	Pixelated Convolution Kernel method
PSF	Point Spread Function
PTF	Palomar Transient Factory
PTFIDE	PTF Image Differencing and Extraction
QA	Quality Assurance
RA	Right Ascension
RMS	Root-Mean-Square fluctuation
RSS	Root-Sum-Squared
RTB	Regression Test Baseline
S/N	Signal-to-Noise
SVB	Slowly (or Spatially) Varying Background
TBD	To Be Determined
TBR	To Be Resolved
WCS	World Coordinate System
ZP	Zero-Point magnitude

Acknowledgments

Many thanks go to Andy Becker for helpful discussions, Carl Grillmair for providing the *DAOPhot* code and example scripts, Branimir Sesar for Wozniak's *DIAPL* tool, and Eran Ofek for invaluable advice. This work was funded in part by the PTF and *Spitzer* projects at IPAC/Caltech.