PSFEx Documentation

Release 3.18.2

E. Bertin, A. Moneti

Wed Jan 11 2017

Contents

CHAPTER 1

Contents

1.1 Introduction

PSFEx[1] (PSF Extractor) is a computer program that extracts precise models of the Point Spread Functions (PSFs)[2] from images processed by **SExtractor**[3] and measures the quality of images. The generated PSF models can be used for model-fitting photometry or morphological analyses. The main features of **PSFEx** are:

- Modelling of any arbitrary non-parametric or parametric, bandwidth-limited, PSF.
- Reconstruction of PSF from undersampled images using super-resolution on the pixel basis, the Gauss-Laguerre basis *[4]* or a user-provided vector basis.
- Modelling of PSF variations as a polynomial function of position in image, any **SExtractor** measurement, or any numerical FITS (Flexible Image Transport System)[4] parameter.
- Tracking of hidden PSF dependencies using Principal Component Analysis[5] *[5]*.
- Computation of PSF homogenisation kernels (to convert variable instrumental PSFs to constant round Moffat[6] *[2]* profiles).
- Automatic selection of point sources.
- Compatibility with **SExtractor** FITS or Multi-Extension FITS catalogue format in input,
- VOTable[7]-compliant XML (eXtensible Markup Language)[8] output of meta-data.
- XSLT (eXtensible Stylesheet Language Transformations)[9] filter sheet provided for convenient access to metadata from a regular web browser.

[1] http://astromatic.net/software/psfex
[2] http://en.wikipedia.org/wiki/Point_spread_function
[3] http://astromatic.net/software/sextractor
[4] http://fits.gsfc.nasa.gov
[5] http://en.wikipedia.org/wiki/Principal_component_analysis
[6] http://en.wikipedia.org/wiki/Moffat_distribution
[7] http://www.ivoa.net/documents/VOTable
[8] http://en.wikipedia.org/wiki/XML
[9] http://en.wikipedia.org/wiki/XSLT

1.2 License

PSFEx[10] *is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.* **PSFEx** *is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with* **PSFEx**. *If not, see* www.gnu.org/licenses/[11].

1.3 Installing the software

1.3.1 Hardware requirements

PSFEx runs in (ANSI) text-mode from a shell. A window system is not necessary for basic operation.

When it comes to memory usage, the amount required by **PSFEx** depends mostly on the number of point sources present in the input catalogues times the number of pixels in the small image that represents each of them. A typical figure is about 15 kbytes per point source; hence even on a modest computer with 1GB of memory, more than 20,000 point sources can easily be accommodated at once.

Note that **PSFEx** takes advantage of multiple CPU cores for some operations.

1.3.2 Obtaining PSFEx

For Linux users, the simplest way to have **PSFEx** up and running is to install the standard binary package the comes with your Linux distribution. Run, e.g., `apt-get psfex` (on Debian) or `dnf psfex` (Fedora) and **PSFEx**, as well as all its dependencies, will automatically be installed. If you decided to install the package this way you may skip the following and move straight to the *next section*.

However if **PSFEx** is not available in your distribution, or to obtain the most recent version, the **PSFEx** source package can be downloaded from the official GitHub repository[12] . One may choose one of the stable releases[13], or for the fearless, a copy of the current master development branch[14].

1.3.3 Software requirements

PSFEx has been developed on GNU/Linux[15] machines and should compile on any POSIX[16]-compliant system (this includes Apple OS X®[17] and Cygwin[18] on Microsoft Windows®[19], at the price of some difficulties with the configuration), provided that the *development* packages of the following libraries have been installed:

- **ATLAS**[20] V3.6 and above[28],
- **FFTw**[21] V3.0 and above[29],

[10] http://astromatic.net/software/psfex
[11] http://www.gnu.org/licences/
[12] https://github.com/astromatic/psfex
[13] https://github.com/astromatic/psfex/releases
[14] https://github.com/astromatic/psfex/archive/master.zip
[15] http://en.wikipedia.org/wiki/Linux
[16] http://en.wikipedia.org/wiki/POSIX
[17] http://www.apple.com/osx
[18] http://www.cygwin.com
[19] http://www.microsoft.com/windows
[20] http://math-atlas.sourceforge.net
[28] Use the `--with-atlas` and/or `--with-atlas-incdir` options of the **PSFEx** **configure** script to specify the **ATLAS** library and include paths if **ATLAS** files are installed at unusual locations.
[21] http://www.fftw.org
[29] Make sure that **FFTw** has been compiled with **configure** options `--enable-threads --enable-float`.

- **PLPlot**[22] V5.9 and above.

On Fedora/Redhat distributions for instance, the development packages above are available as `atlas-devel`, `fftw-devel` and `plplot-devel`. **PLPlot** is only required for producing diagnostic plots. Note that **ATLAS** and **FFTw** are not necessary if **PSFEx** is linked with Intel® 's MKL (Math Kernel Library)[23] library.

1.3.4 Installation

To install from the GitHub source package, you must first uncompress the archive:

```
$ unzip psfex-<version>.zip
```

A new directory called `psfex-<version>` should now appear at the current location on your disk. Enter the directory and generate the files required by the autotools[24], which the package relies on:

```
$ cd psfex-<version>
$ sh autogen.sh
```

A `configure` script is created. This script has many options, which may be listed with the `--help` option:

```
$ ./configure --help
```

No options are required for compiling with the default GNU C compiler (`gcc`) if all the required libraries are installed at their default locations:

```
$ ./configure
```

Compared to `gcc` and the librairies above, the combination of the Intel® compiler (`icc`) and the MKL[25] libraries can give the **PSFEx** executable a strong boost in performance, thanks to better vectorized code. If `icc` and the MKL are installed on your system[30] , you can take advantage of them using

```
$ ./configure --enable-mkl
```

Additionally, if the **PSFEx** binary is to be run on a different machine that does not have `icc` and the MKL installed (e.g., a cluster computing node), you must configure a partially statically linked executable using

```
$ ./configure --enable-mkl --enable-auto-flags --enable-best-link
```

In all cases, **PSFEx** can now be compiled with

```
$ make -j
```

An `src/psfex` executable is created. For system-wide installation, run the usual

```
$ sudo make install
```

You may now check that the software is properly installed by simply typing in your shell:

```
$ psfex
```

which will return the version number and other basic information (note that some shells require the `rehash` command to be run before making a freshly installed executable accessible in the execution path).

[22] http://www.plplot.org
[23] http://software.intel.com/intel-mkl
[24] http://en.wikipedia.org/wiki/GNU_Build_System
[25] http://software.intel.com/intel-mkl
[30] The Linux versions of the Intel® compiler and MKL are available for free to academic researchers, students, educators and open source contributors.

1.4 Getting started

PSFEx is run from the shell with the following syntax:

```
$ psfex Catalog1 [Catalog2 ...] -c configuration-file [-Parameter1 Value1 -
↪Parameter2 Value2 ...]
```

The parts enclosed within brackets are optional. The file names of input catalogues can be directly provided in the command line, or in lists that are ASCII files with each catalogue name preceded by `@` (one per line). One should use lists instead of the catalogue file names if the number of input catalogues is too large to be handled directly by the shell. Any *-Parameter Value* statement in the command-line overrides the corresponding definition in the configuration file or any default value (see below).

1.4.1 Input files

Catalogues

PSFEx does not work directly on images. Instead, it operates on **SExtractor**[32] catalogues that have a small sub-image ("vignette") recorded for each detection. This makes things much easier for **PSFEx** as it does not have to handle the detection and deblending processes. The catalogue files read by **PSFEx** must be in **SExtractor** FITS_LDAC binary format. This allows **PSFEx** to have access to the original image header content. The catalogues *must* contain all the following parameters in order to be processable by **PSFEx**:

- small image ("vignette") centered on the object, `VIGNET(w,h)`, where w and h are respectively the width and the height of the image in pixels,
- centroid coordinates, e.g. `X_IMAGE` and `Y_IMAGE`,
- half-light radius `FLUX_RADIUS`,
- Signal-to-noise ratio in a Gaussian window `SNR_WIN`
- flux measured through a fixed aperture, e.g. `FLUX_APER(1)`,
- flux uncertainty, e.g. `FLUXERR_APER(1)`,
- object elongation `ELONGATION`,
- extraction flags `FLAGS`.

The `VIGNET` dimensions w and `h` set the maximum size, in pixels, of the image area stored for each detection. It is advised to use square sub-images ($w = h$) with an odd number of pixels (for symmetry across the central pixel), and so that the sub-image covers much of the visible footprint of non-saturated stars.

The size of sub-images in the catalogue (here 35×35 pixels) must been chosen so that the frame encloses much of the visible footprint of non-saturated stars. It is recommended not to use excessively large sub-images as they lead to unpractically large catalogues and make the **PSFEx** PSF cleaning procedure less robust. In practice, values such as in `VIGNET(45,45)` for instance, will generally work well with most images.

SExtractor configuration settings for the pre-**PSFEx** run do not require much tuning in general. The **SExtractor** configuration file only need to differ from the default one on a few keywords. However these keywords must be set with care:

- `CATALOG_TYPE` should be set to `FITS_LDAC` (binary FITS[33] catalogue).
- `PHOT_APERTURE` defines the diameter of the circular aperture (in pixels) used as a reference for normalising the amplitude of the PSF model. It should be set to a value large enough so that variations due to seeing or aberrations are negligible at the level required for photometric analyses. But be aware that using excessively large apertures lead to noisy measurements and are more prone to light pollution by neighbouring sources. For professional, ground-based images, a value corresponding to an aperture diameter of $5''$ is often a good compromise.

[32] http://astromatic.net/software/sextractor
[33] http://fits.gsfc.nasa.gov

- Detector gain: The effective "gain" (or more exactly the conversion factor, in units of electrons/ADU) is required by **SExtractor** to compute the uncertainty on pixel values, especially with bright star images. The GAIN_KEY configuration parameter tells SExtractor what keyword in the original FITS image header carries the detector gain. The default string for GAIN_KEY is GAIN. In multi-CCD cameras, the gain can slightly vary from one CCD to another, and can also vary with time. When working with single exposures, it is recommended to let **SExtractor** read the gain value from the image header, if it is present. In some detector chips with multiple readouts, several values of the gain may be present in the same header under different keywords (e.g. GAINA, GAINB). Since the gain differences will often be negligibly small for **PSFEx** (10% or less), it is usually safe to use one single value for the whole chip (for example, GAINA). Note that what matters for reduced images is the "effective" gain, not the original detector gain. Dividing pixel values by some amount (e.g., the exposure time or a non-normalised flat field) multiplies the effective gain by the same amount. Combining several images also modifies the gain. For example if the final image is the mean of several images, the effective gain will be equal to the initial gain multiplied by the number of images. Taking the median or a weighted average also affect the gain (see, e.g., the SWarp[34] documentation). Recent versions of **SWarp** properly take into account the effect of flux scaling and stacking on the gain, and insert the average effective gain in the output image header. Finally if no keyword with the name specified by GAIN_KEY can be found in the FITS image header, **SExtractor** will fall back to using the gain value specified by the GAIN configuration parameter. The default fallback value is 0, which actually tells SExtractor that the gain should be considered infinite (bright pixels not noisier that faint pixels).
- Saturation level: **PSFEx** requires **SExtractor** to flag all saturated sources, which may otherwise contaminate the "clean" star sample used to compute the PSF model. **SExtractor** identifies saturated sources by checking if the value of at least one source pixel exceeds a given "saturation level". As for the gain above, **SExtractor** examines first the value of a FITS image header keyword to read the saturation level (in ADUs). The header keyword can be set with the SATUR_KEY **SExtractor** configuration parameter; the default string for SATUR_KEY is SATURATE. SATURATE is commonly found in image files released by observatories and pipelines. Unfortunately, in practice, it is often found to be set at a value higher than that at which the detector markedly starts to behave non-linearly. It is therefore highly recommended to examine *visually* saturated stars on images and check if pixel values systematically exceed the saturation level reported in the header. If not, it is advised to give SATUR_KEY a name unlikely to exist as a keyword in the FITS file (for example, DUMMY), and force the saturation level in **SExtractor** to a lower value using the SATUR_LEVEL fallback parameter. Note that some detector/amplifier combinations start becoming non-linear at levels below the apparent saturation limit, so it is always safer to give a saturation level about 10% lower than the lowest value derived from the visual examination of all images.

1.4.2 Output files

PSF model files (.psf)

The main purpose of **PSFEx** is to create a PSF model for each of the images from which the input catalogues were extracted. The PSF models are stored under file names that are given the .psf extension by default (this may be changed with the PSF_SUFFIX configuration parameter). The .psf files are FITS binary tables that can be read back into **SExtractor** to perform accurate model-fitting of the sources being detected. A detailed description of the .psf file format is given *in the Appendix*.

PSF homogenisation files (.homo)

This is presently an experimental feature. In addition to computing PSF models, **PSFEx** has the possibility to derive "PSF homogenisation kernels" for all input catalogues. A PSF homogenisation kernel is a (variable) convolution kernel which, when applied to an image, gives the point sources it contains a constant, arbitrary shape. For practical purposes the target shape will preferably be a perfectly round analytical function, such as a Moffat *[2]* profile: Homogenising the PSF of a set of images can allow for more consistent image combinations and measurements, once the consequences on noise have been properly taken into account.

PSFEx stores PSF homogenisation kernels as FITS data cubes. File names are given the .homo extension by default; this may be changed using the HOMOKERNEL_SUFFIX configuration parameter. .homo files can be read

[34] https://www.astromatic.net/pubsvn/software/swarp/trunk/doc/swarp.pdf

by the **PSFnormalize** software developed by Tony Darnell from the Dark Energy Survey data-management team to perform fast convolution of the original images *[3]*. The **SWarp** software may also later include this possibility.

Diagnostic files

Three types of files can be generated by **PSFEx**, providing diagnostics about the derived PSF and the modelling process:

- "Check-images" are basic FITS files containing images of the PSF model, fit residuals, etc.. Configuration parameters CHECKIMAGE_TYPE and CHECKIMAGE_NAME allow the user to provide a list of check-image types and file names, respectively, to be produced by **PSFEx**. A complete list of available check-image types is given in §[chap:paramlist]. Many check-images are actually aggregates of several small images; they may be stored as grids (the default) or as datacubes if the CHECKIMAGE_DATACUBE parameter is set to Y.
- "Check-plots" are graphic charts generated by PSFEx, showing maps or trends of PSF measurements. The CHECKPLOT_TYPE and CHECKPLOT_NAME configuration parameters allow the user to provide a list of check-plot types and file names, respectively. A variety of raster and vector file formats, from JPEG to Postscript, can be set with CHECKPLOT_DEV (the default format is PNG). See the *CHECKPLOT* section of the *configuration parameter list* below for details.
- An XML[35] file providing a processing summary and various statistics in VOTable[36] format is written if the WRITE_XML switch is set to Y (the default). The XML_NAME parameter can be used to change the default file name psfex.xml. The XML file can be displayed with any recent web browser; the XSLT stylesheet installed together with **PSFEx** will automatically translate it into a dynamic, user-friendly web-page (Fig. 1.1). For more advanced usages (e.g., access from a remote web server), alternative XSLT translation URLs may be specified using the XSL_URL configuration parameter.

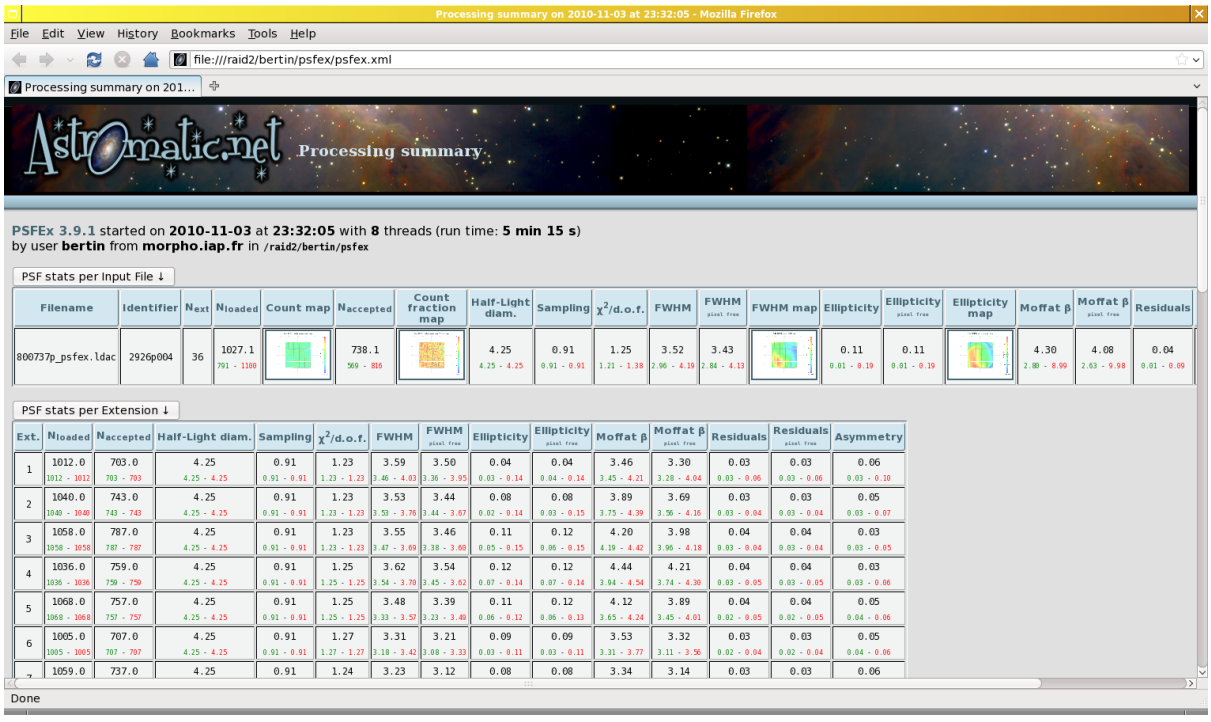

Fig. 1.1: Rendition of a psfex.xml XML-VOTable file generated by **PSFEx** with the Firefox[37] web-browser.

[35] http://en.wikipedia.org/wiki/XML
[36] http://www.ivoa.net/documents/VOTable
[37] http://www.mozilla.org/firefox

1.4.3 The Configuration file

Each time it is run, **PSFEx** looks for a configuration file. If no configuration file is specified in the command-line, it is assumed to be called `default.psfex` and to reside in the current directory. If no configuration file is found, **PSFEx** will use its own internal default configuration.

Creating a configuration file

PSFEx can generate an ASCII dump of its internal default configuration, using the `-d` option. By redirecting the standard output of **PSFEx** to a file, one creates a configuration file that can easily be modified afterwards:

```
$ psfex -d > default.psfex
```

and a more extensive dump with less commonly used parameters can be generated by using the `-dd` option.

Format of the configuration file

The format is ASCII. There must be only one parameter set per line, following the form:

```
Config-parameter      Value(s)
```

Extra spaces or linefeeds are ignored. Comments must begin with a # and end with a linefeed. Values can be of different types: strings (can be enclosed between double quotes), floats, integers, keywords or Boolean (*Y/y* or *N/n*). Some parameters accept zero or several values, which must then be separated by commas. Integers can be given as decimals, in octal form (preceded by digit 0), or in hexadecimal (preceded by *0x*). The hexadecimal format is particularly convenient for writing multiplexed bit values such as binary masks. Environment variables, written as `$HOME` or `${HOME}` are expanded.

Configuration parameter list

Here is a list of all the parameters known to **PSFEx**. Please refer to next section for a detailed description of their meaning. Some "advanced" parameters (indicated with an asterisk) are also listed. They must be used with caution, and may be rescoped or removed without notice in future versions.

Parameter `BADPIXEL_FILTER*`

Default `N`

Type *Boolean*

If true (`Y`), input objects with vignettes containing more than `BADPIXEL_NMAX` pixels flagged by **SExtractor** as bad or from deblended neighbours will be rejected.

Parameter `BADPIXEL_NMAX`

Default `N`

Type *Boolean*

Maximum number of bad pixels tolerated in the vignette before an object is rejected (`BADPIXEL_FILTER` must be set to `Y`)

Parameter `BASIS_NAME`

Default `basis.fits`

Type *String*

File name for the user-supplied FITS datacube of basis vector images (`BASIS_TYPE}` must have been set to `FILE`)

Parameter BASIS_NUMBER

Default 20

Type *integer*

Size of basis vector set: square-root of the number of pixels for BASIS_TYPE PIXEL, $n_{\max}$ for BASIS_TYPE GAUSS-LAGUERRE, or number of vectors for BASIS_TYPE FILE.

Parameter BASIS_SCALE

Default 1.0

Type *float*

Scale size of BASIS_TYPE GAUSS-LAGUERRE vector images.

Parameter BASIS_TYPE

Default BASIS_AUTO

Type *keyword*

Basis vector set:

- NONE: No basis; the PSF is derived solely from the robust
- PIXEL: Pixel basis (super-resolution)
- PIXEL_AUTO: Equivalent to NONE for properly sampled images; switches automatically to PIXEL (super-resolution) for critically sampled and undersampled data.
- GAUSS_LAGUERRE: Gauss-Laguerre basis (also known as *polar shapelets* in the weak-lensing community).
- FILE: User-supplied vector basis, in the form of a FITS datacube (see BASIS_NAME).

Parameter CENTER_KEYS

Default X_IMAGE, Y_IMAGE

Type *strings*

Catalogue Keys (**SExtractor** measurement parameters) that define the initial guess for the source coordinates. Note that all input vignettes are automatically re-centred by **PSFEx** using an iterative Gaussian-weighted algorithm, hence the centring parameter is not critical.

Parameter CHECKIMAGE_CUBE

Default N

Type *Boolean*

If true (Y), check-images will be saved as data-cubes.

Parameter CHECKIMAGE_NAME

Default chi.fits,proto.fits,samp.fits,resi.fits,snap.fits

Type *strings*

File name of the check-image (diagnostic FITS image) of each type (**`.fits extension`** is not required, as it is assumed by default).

Parameter `CHECKIMAGE_TYPE`

Default `CHI, PROTOTYPES, SAMPLES, RESIDUALS, SNAPSHOTS`

Type *keywords*

Types of check-images (diagnostic FITS images) to generate during **`PSFEx`** processing:

- `NONE` No check-image.
- `CHI` (square-root of) χ^2 maps for all input vignettes.
- `PROTOTYPES` Versions of input vignettes, recentred, rescaled and resampled to PSF resolution.
- `SAMPLES` Input vignettes in their original position, resolution and flux scaling.
- `RESIDUALS` Input vignettes with best-fitting local PSF models subtracted.
- `SNAPSHOTS` Grid of PSF model snapshots reconstructed at each position/context.
- `MOFFAT` Grid of *Moffat models* fitted to PSF model snapshots at each position/context.
- `-MOFFAT` Grid of PSF model snapshots reconstructed at each position/context with best-fitting *Moffat models* subtracted.
- `-SYMMETRICAL` Grid of PSF model snapshots reconstructed at each position/context with symmetrised image subtracted.
- `BASIS` Basis vector images used by **`PSFEx`** to model the PSF.

Parameter `CHECKPLOT_ANTIALIAS`

Default `Y`

Type *Boolean*

If true (`Y`), PBM, PNG and JPEG check-plots are generated with anti-aliasing. ImageMagick[38] 's `convert` tool must be installed.

Parameter `CHECKPLOT_DEV`

Default `PNG`

Type *keywords*

PLPlot devices to be used for check-plots (all devices may not be available, see PLPlot documentation for details):

- `NULL` No output
- `XWIN` X-Window
- `TK` Tk window (if available)
- `XTERM` XTerm window
- `AQUATERM` AquaTerm window (Mac OS X)
- `PLMETA` PLPlot .plm} meta-file
- `XFIG` XFig .fig} vector file

[38] http://www.imagemagick.org

- `LJIIP` HP LaserJet IIP .lj} bitmap file
- `LJ_HPGL` HP LaserJet .hpg HPGL vector file
- `IMP` Impress .imp} file
- `PBM` Portable BitMap .pbm image
- `PNG` Portable Network Graphics .png image
- `JPEG` JPEG .jpg image
- `PDF` Portable Document Format .pdf file
- `PS` Black-and-white .ps Postscript file
- `PSC` Colour .ps Postscript file
- `PSTEX` PSTeX (a variant of Postscript) .ps file

Parameter `CHECKPLOT_NAME`

Default `fwhm, ellipticity, counts, countfrac, chi, resi`

Type *strings*

File names for each series of check-plots. **PSFEx** will automatically insert the associated catalogue names, and append/replace file name extensions with the appropriate ones, depending on the chosen `CHECKPLOT_DEV(s)` (.png for PNG files, .jpg for JPEG, etc.).

Parameter `CHECKPLOT_RES`

Default `0`

Type *integers* ($n \leq 2$)

Check-plot x,y resolution for bitmap devices (`0` is equivalent to `800,600`).

Parameter `CHECKPLOT_TYPE`

Default `FWHM, ELLIPTICITY, COUNTS, COUNT_FRACTION, CHI2, RESIDUALS`

Type *keywords*

Diagnostic check-plots to be generated during **PSFEx** processing (**PSFEx** must have been configured without the `--without-plplot option`):

- `NONE` No plot.
- `FWHM` Map of the model PSF Full-Width at Half-Maximum over the field of view (one for each input catalogue).
- `ELLIPTICITY` Map of the model PSF ellipticity over the field of view (one for each input catalogue).
- `COUNTS` Map of the spatial density of point sources (initially) selected over the field of view (one for each catalogue).
- `COUNT_FRACTION` Map of the fraction of point sources accepted over the field of view (one for each catalogue).
- `CHI2` Map of the average $\chi^2/\mathrm{d.o.f.}$ over the field of view (one for each catalogue).
- `MOFFAT_RESIDUALS` Map of Moffat (eq.~[ref{eq:moffat}]) residual indices over the field of view (one for each catalogue).
- `ASYMMETRY` Map of asymmetry indices over the field of view (one for each catalogue).

Parameter HOMOBASIS_NUMBER

Default 10

Type *integer*

Size of the homogenisation kernel basis vector set: $n_{\max}$ for HOMOBASIS_TYPE GAUSS-LAGUERRE.

Parameter HOMOBASIS_SCALE

Default 1.0

Type *float*

Scale size of HOMOBASIS_TYPE GAUSS-LAGUERRE homogenisation kernel vector images.

Parameter HOMOBASIS_TYPE

Default NONE

Type *keyword*

Basis vector set for the homogenisation kernel:

- NONE No basis; no homogenisation kernel is computed.}
- GAUSS_LAGUERRE Gauss-Laguerre basis (also known as *polar shapelets* in the weak-lensing community).

Parameter HOMOKERNEL_SUFFIX

Default .homo.fits

Type *string*

Filename suffix of the homogenisation kernels computed by **PSFEx**.

Parameter HOMOPSF_PARAMS}*

Default 2.0, 3.0

Type *floats* ($n \leq 2$)

Moffat model Full-Width at Half-Maximum and β parameters of the idealised target PSF chosen for homogenisation.

Parameter MEF_TYPE

Default INDEPENDENT

Type *keyword*

How PSFEx should deal with multi-extension catalogues (extracted from mosaic camera images):

- INDEPENDENT Derive the PSF model for each extension independently.
- COMMON Derive a common PSF model for all extensions.

Parameter NEWBASIS_NUMBER

Default 8

Type *integer*

Size of the image vector set (number of basis vectors) derived by **PSFEx** from the input vignettes.}

Parameter NEWBASIS_TYPE

Default NONE

Type *keyword*

Type of image vector bases derived from input vignettes by |

- NONE No basis is computed.
- PCA_MULTI Karhunen-Lo‘eve basis from Principal Component Analysis on all FITS extensions.
- PCA_SINGLE Karhunen-Lo‘eve bases from Principal Component Analysis on individual FITS extensions.

Parameter NTHREADS

Default 0

Type *integer*

Number of threads (processes) to be used for parallel computation. **PSFEx** must have been configured with the --disable-threads option at compile time for this parameter to take effect. Note that multi-threading is disabled in the current version of **PSFEx**

Parameter PHOTFLUX_KEY

Default FLUX_APER(1)

Type *string*

Catalogue Key (**SExtractor** measurement parameter) that defines the flux of sources, and therefore the normalisation of the PSF amplitude. It is recommended to use a fixed aperture magnitude; the aperture diameter set in **SExtractor** should be large enough so that the fraction of flux enclosed stays constant from point source to point source, and small enough to preserve the signal-to-noise ratio.

Parameter PHOTFLUXERR_KEY

Default FLUXERR_APER(1)

Type *string*

Catalogue Key (**SExtractor** measurement parameter) that defines the flux measurement uncertainty on each source. It is used for computing the source signal-to-noise ratio.

Parameter PSF_ACCURACY

Default 0.01

Type *float*

Expected accuracy of vignette pixel values (standard deviation of the flux fraction).

Parameter `PSF_PIXELSIZE`

Default `1.0`

Type *float*

Effective pixel size (width of the top-hat intra-pixel response function) in pixel step units.

Parameter `PSF_RECENTER`

Default `Y`

Type *Boolean*

If true (`Y`), input vignettes are recentred at each iteration of the PSF modelling process.

Parameter `PSF_SAMPLING`

Default `0.0`

Type *float*

Sampling step of the PSF models, in pixels. Use `0` for automatic sampling.

Parameter `PSF_SIZE`

Default `25, 25`

Type *integers* ($n \leq 2$)

Dimensions of the tabulated PSF models, in PSF `pixels`.

Parameter `PSF_SUFFIX`

Default ‘‘ .psf‘‘

Type *string*

Filename suffix for PSF models computed by **`PSFEx`**.

Parameter `PSFVAR_DEGREES`

Default `2`

Type *integers* ($n = n_{\mathrm{groups}}$)

Degree of polynomial of each context group. `0` indicates a constant PSF.

Parameter `PSFVAR_GROUPS`

Default `1, 1`

Type *integers* ($n = n_{\texttt{PSFVAR\_KEYS}}$)

Polynomial group which each context key belongs to.

Parameter `PSFVAR_KEYS`

Default `X_IMAGE, Y_IMAGE`

Type *strings* ($n \leq 2$)

List of `keys` (**SExtractor** measurement parameters) on which the PSF is supposed to depend (e.g. `X_IMAGE, Y_IMAGE` for a spatial mapping of the PSF). Keywords preceded with a colon are interpreted as FITS image keywords instead of **SExtractor** parameters.

Parameter `PSFVAR_NSNAP`

Default `9`

Type *integer*

Number of PSF snapshots computed on each axis. This also defines the resolution of the grid on which diagnostics and check-plot maps are computed.

Parameter `SAMPLE_AUTOSELECT`

Default `Y`

Type *Boolean*

If true (`Y`), input vignettes are automatically selected based on the source FWHMs, inside the range specified by `SAMPLE_FWHMRANGE`, with fractional FWHM variability `SAMPLE_VARIABILITY`.

Parameter `SAMPLE_FLAGMASK`

Default `0x00fe`

Type *integer*

Bit mask applied to SExtractor flags for rejecting input vignettes.

Parameter `SAMPLE_FWHMRANGE*`

Default `2.0, 10.0`

Type *floats* ($n = 2$)

Range (in pixels) of source FWHMs (Full-Width at Half-Maximum) allowed for input vignettes. FWHMs are currently estimated based on **SExtractor**'s `FLUX_RADIUS` measurements.

Parameter `SAMPLE_MAXELLIP`

Default `0.3`

Type *float*

Maximum source ellipticity (i.e. $\frac{\texttt{A\_IMAGE}-\texttt{B\_IMAGE}}{\texttt{A\_IMAGE}+\texttt{B\_IMAGE}}$) allowed for input vignettes.

Parameter `SAMPLE_MINSN`

Default `20.0`

Type *float*

Minimum source Signal-to-Noise ratio allowed for input vignettes.

Parameter `SAMPLE_VARIABILITY`

Default `0.2`

Type *float*

Maximum fractional FWHM variability (1.0 = 100%) allowed for input vignettes.

Parameter `SAMPLEVAR_TYPE`

Default `SEEING`

Type *keyword*

Catalogue-to-catalogue variability criteria for vignette selection:

- `NONE` No differences between catalogues.
- `SEEING` Seeing (hence FWHM) is expected to vary.

Parameter `STABILITY_TYPE`

Default `EXPOSURE`

Type *keyword*

- `EXPOSURE` {???}
- `SEQUENCE` {???}

Parameter `VERBOSE_TYPE`

Default `NORMAL`

Type *keyword*

Degree of verbosity of the software on screen:

- `QUIET` No Output besides warnings and error messages
- `NORMAL` `Normal` display with messages updated in real time using ASCII escapes-sequences
- `LOG` Like `NORMAL`, but without real-time messages and ASCII escape-sequences
- `FULL` Everything

Parameter `WRITE_XML`

Default `Y`

Type *Boolean*

If true (`Y`), an XML summary file will be written after completing the processing.

Parameter `XML_NAME`

Default `psfex.xml`

Type *string*

File name for the XML output of **`PSFEx`**.

Parameter `XSL_URL`

Default `.`

Type *string*

URL of an XSL style-sheet for the XML output of **PSFEx**. This URL will appear in the `href` attribute of the `style-sheet` tag.

1.5 How PSFEx works

1.5.1 Overview of the software

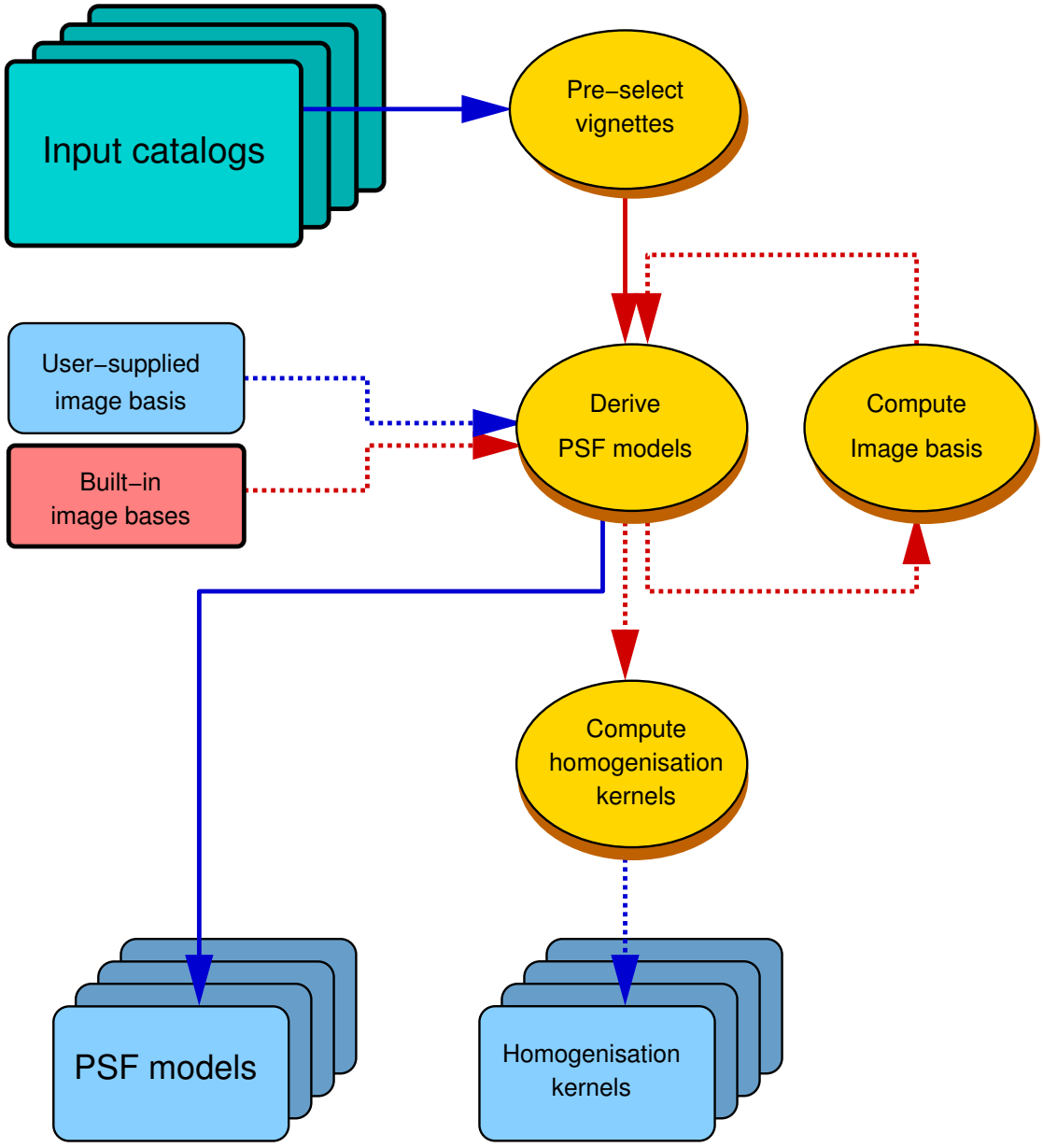

Fig. 1.2: Global Layout of **PSFEx**.

The global layout of **PSFEx** is presented in Fig. 1.2. There are many ways to operate the software. Let us now describe the important steps in the most common usage modes.

1. **PSFEx** starts by examining the catalogues given in the command line. In the default operating mode, for mosaic cameras, Multi-Extension FITS (MEF) files are processed extension by extension. **PSFEx** pre-selects detections which are likely to be point sources, based on source characteristics such as half-light radius and ellipticity, while rejecting contaminated or saturated objects.

2. For each pre-selected detection, the "vignette" (produced by **SExtractor**) and a "context vector" are loaded in memory. The context vector represents the set of parameters (like position) on which the PSF model will depend explicitly.
3. The PSF modelling process is iterated 4 times. Each iteration consists of computing the PSF model, comparing the vignettes to the model reconstructed in their "local contexts", and excluding detections that show too much departure between the data and the model.
4. Depending on the configuration, two types of Principal Component Analyses (PCAs) may be included at this stage, either to build an optimised image vector basis to represent the PSF, or to track hidden dependencies of the model. In both cases, they result in a second round of PSF modelling.
5. The PSF models are saved to disk. If requested, PSF homogenisation kernels may also be computed and written to disk at this stage. Finally, diagnostic files are generated.

1.5.2 Point source selection

PSFEx requires the presence of unresolved sources (stars or quasars) in the input catalogue(s) to extract a valid PSF model. In some astronomical observations, the fraction of suitable point sources that may be used as good approximations to the local PSF may be rather low. This is especially true for deep imaging in the vicinity of galaxy clusters at high galactic latitudes, where unsaturated stars may comprise only a small percentage of all detectable sources.

Selection criteria

To minimise as much as possible the assumptions on the shape of the PSF, **PSFEx** adopts the following selection criteria:

- the shape of suitable unresolved (unsaturated) sources does not depend on the flux.
- amongst image profiles of all real sources, those from unresolved sources have the smallest Full-Width at Half Maximum (FWHM).

These considerations as well as much experimentation led to adopting a first-order selection similar to the rectangular cut in the half-light-radius ($r_{\rm h}$) vs. magnitude plane, popular amongst members of the weak lensing community (Kaiser et al. 1995). **SExtractor**'s `FLUX_RADIUS` parameter with input parameter `PHOT_FLUXFRAC` $= 0.5$ provides a good estimate for $r_{\rm h}$. In **PSFEx**, the "vertical" locus produced by point sources (whose shape does not depend on magnitude) is automatically framed between a minimum signal-to-noise threshold and the saturation limit on the magnitude axis, and within some margin around the local mode on the $r_{\rm h}$ axis (Fig. 1.3). The relative width of the selection box is set by the `SAMPLE_VARIABILITY` configuration parameter (0.2 by default), within boundaries defined by half the `SAMPLE_FWHMRANGE` parameter (between 2 and 10 pixels by default). Additionally, to provide a better rejection of image artifacts and multiple objects, **PSFEx** excludes detections

- with a Signal-to-Noise Ratio (SNR) below the value set with the `SAMPLE_MINSN` configuration parameter (20 by default). The SNR is defined here as the ratio between the source flux and the source flux uncertainty.
- with **SExtractor** extraction `FLAGS` that match the mask set by the `SAMPLE_FLAGMASK` configuration keyword. The default mask (00fe in hexadecimal) excludes all flagged objects, except those with `FLAGS` = 1 (indicating a crowded environment).
- with an ellipticity exceeding the value set with the `SAMPLE_MAXELLIP` configuration parameter (0.3 by default). The ellipticity is defined here as $(A-B)/(A+B)$, where A and B are the lengths of the major and minor axes, respectively. The ratio A/B is also called the `ELONGATION`. Note that, for historical reasons, this definition differs from the one use in **SExtractor**, which is $(1 - B/A)$
- that include pixels that were given a weight of 0 (for weighted source extractions).

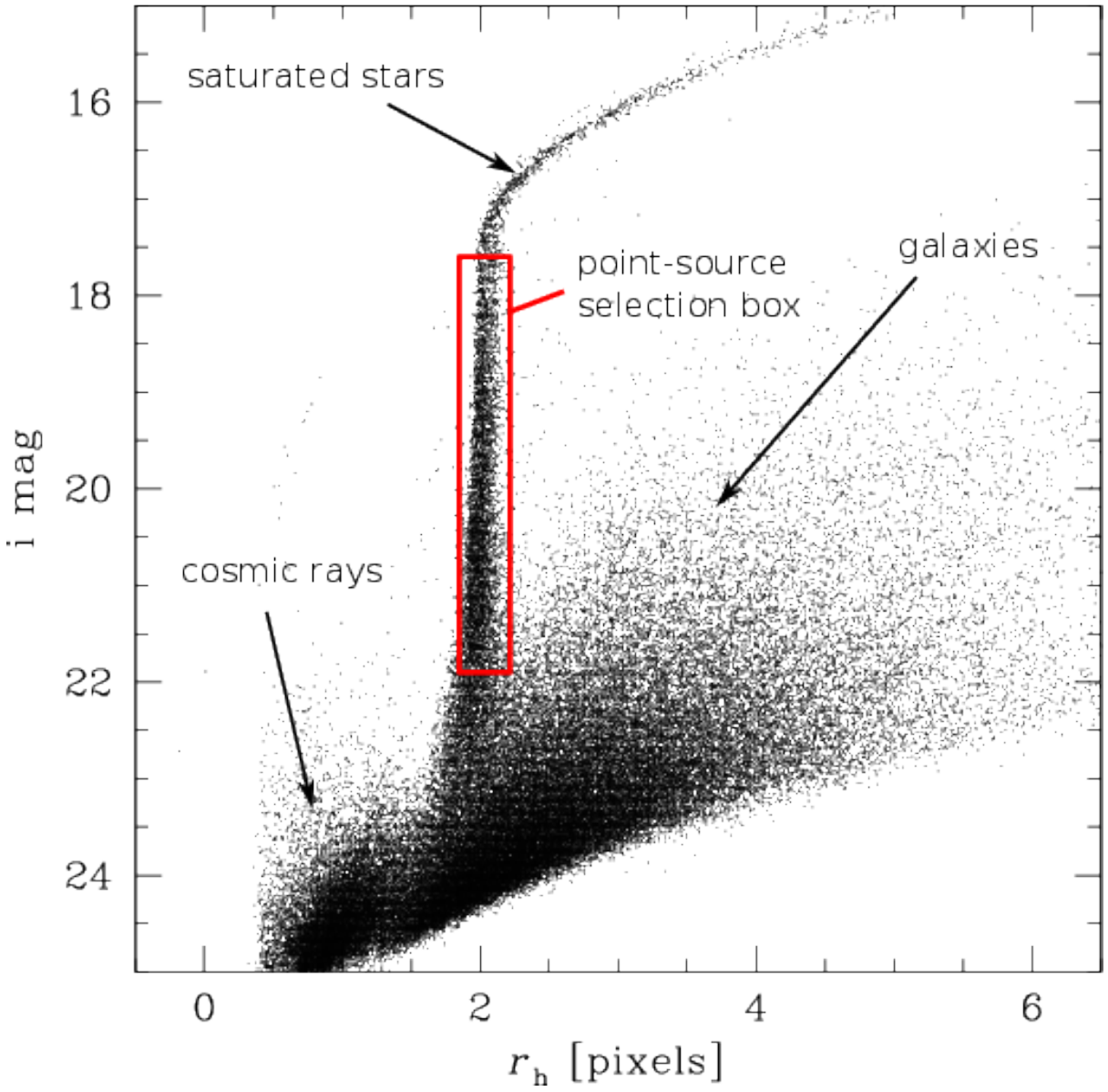

Fig. 1.3: Half-light-radius ($r_{\rm h}$, estimated by **SExtractor**'s FLUX_RADIUS) vs magnitude (MAG_AUTO) for a 520 s CFHTLS exposure at high galactic latitude taken with the Megaprime instrument in the i band. The rectangle enclosing part of the stellar locus represents the approximate boundaries set automatically by **PSFEx** to select point sources.

Iterative filtering

Despite the filtering process, a small fraction of the remaining point source candidates (typically 5-10% on ground-based optical images at high galactic latitude) is still unsuitable to serve as a realisation of the local PSF, because of contamination by neighbouring objects. Iterative procedures to subtract the contribution from neighbour stars have been successfully applied in crowded fields *[6][7]*. However these techniques do not solve the problem of pollution by non-stellar objects like image artifacts, a common curse of wide field imaging, and contaminated point sources still have to be filtered out.

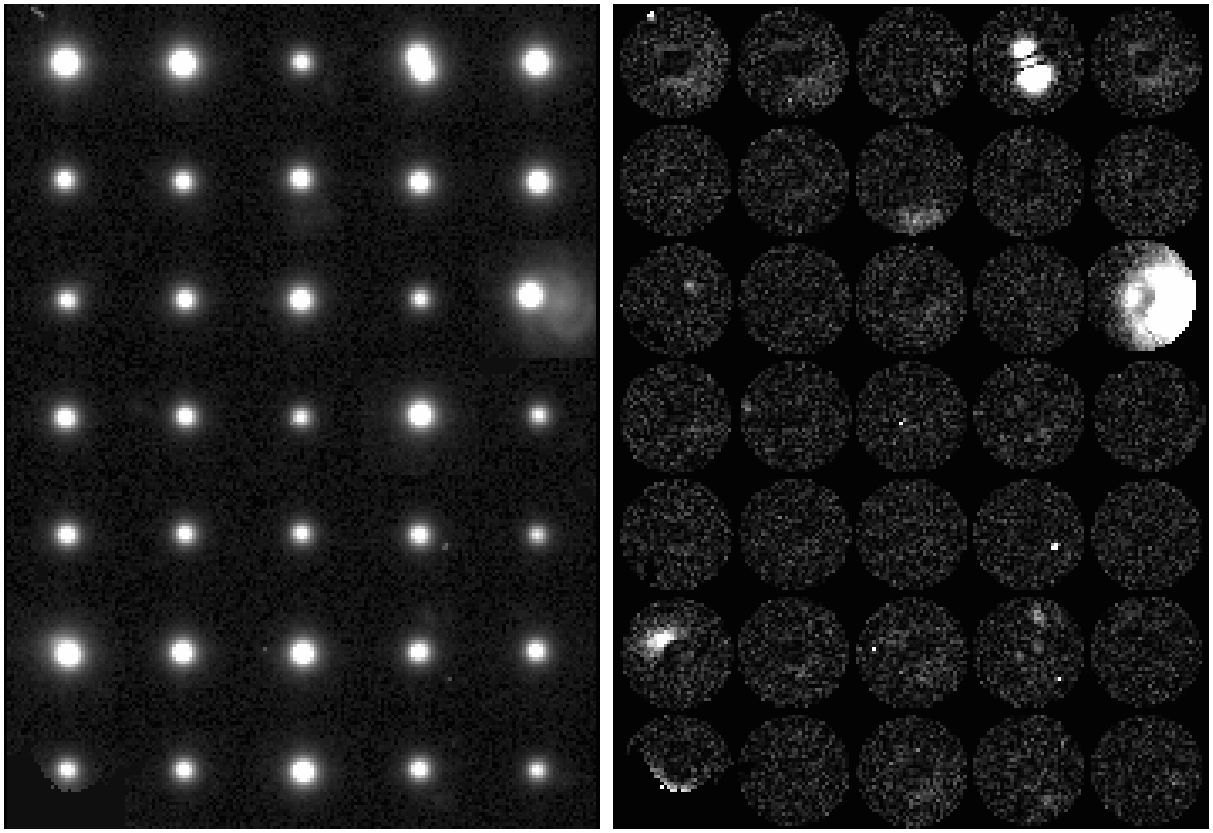

Fig. 1.4: *Left:* some source images selected for deriving a PSF model of a MEGACAM image (the basic rejection tests based on **SExtractor** flags and measurements were voluntarily bypassed to increase the fraction of contaminants in this illustration). *Right:* map of residuals computed as explained in the text; bright pixels betray interlopers like cosmic ray hits and close neighbour sources.

The iterative rejection process in **PSFEx** works by deriving a 1st-order estimate of the PSF model, and computing a map of the residuals of the fit of this model to each point source (Fig. 1.4): each pixel of the map is the square of the difference square of the model with the data, divided by the σ_i^2 estimate from equation (1.3). The PSF model may be "rough" at the first iteration, hence to avoid penalising poorly fitted bright source pixels, the factor α is initially set to a fairly large value, 0.1—0.3. Assuming that the fitting errors are normally distributed, and given the large number of degrees of freedom (the tabulated values of the model), the distribution of $\sqrt{\chi^2}$ derived from the residual maps of point sources is expected to be Gaussian to a good approximation. Contaminated profiles are identified using κ–σ clipping to the distribution of $\sqrt{\chi^2}$. Our experiments indicate that the value $\kappa = 4$ provides a consistent compromise between being too restrictive and being too permissive. **PSFEx** repeats the PSF modelling / source rejection process 3 more times, with decreasing α, before delivering the "clean" PSF model.

1.5.3 Modelling the PSF

In **PSFEx**, the PSF is modelled as a linear combination of basis vectors. Since the PSF of an optical instrument is the Fourier Transform of the auto-correlation of its pupil, the PSF of any instrument with a finite aperture is bandwidth-limited. According to the Shannon sampling theorem, the PSF can therefore be perfectly reconstructed

(interpolated) from an infinite table of regularly-spaced samples. For a finite table, the reconstruction will not be perfect: extended features, such as profile wings and diffraction spikes caused by the high frequency component of the pupil function, will obviously be cropped. With this limitation in mind, one may nevertheless reconstruct with good accuracy a tabulated PSF thanks to sinc interpolation *[8]*. Undersampled PSFs can also be represented in the form of tabulated data provided that a finer grid satisfying Nyquist's criterion is used *[9][10]*.

For reasons of flexibility and interoperability with other software, we chose to represent PSFs in **PSFEx** as small images with adjustable resolution. These PSF "images" can be either derived directly, treating each pixel as a free parameter ("pixel" vector basis), or more generally as a combination of basis vector images.

Pixel basis

The pixel basis is selected by setting `BASIS_TYPE` to `PIXEL`.

Recovering aliased PSFs - If the data are undersampled, unaliased Fourier components can in principle be recovered from the images of several point sources randomly located with respect to the pixel grid, using the principle of super-resolution *[11]*. Working in the Fourier domain, *[12]* shows how PSFs from the Hubble Space Telescope Planetary Camera and Wide-Field Planetary Camera can be reconstructed at 3 times the original instrumental sampling from a large number of undersampled star images. However, solving in the Fourier domain gives far from satisfactory results with real data. Images have boundaries; the wings of point source profiles may be contaminated with artifacts or background sources; the noise process is far from stationary behind point sources with high S/N, because of the local photon-noise contribution from the sources themselves. All these features generate spurious Fourier modes in the solution, which appear as parasitic ripples in the final, super-resolved PSF.

A more robust solution is to work directly in pixel space, using an interpolation function; we may use the same interpolation function later on to *fit* the tabulated PSF model for point source photometry. Let $\boldsymbol{\phi}$ be the vector representing the tabulated PSF, $h_s(\boldsymbol{x})$ an interpolation function, η the ratio of the PSF sampling step to the original image sampling step (oversampling factor). The interpolated value at image pixel i of $\boldsymbol{\phi}$ centered on coordinates $\boldsymbol{x}_s$ is

$$\phi'_i(\boldsymbol{x}_s) = \sum_j h_s\left(\boldsymbol{x}_j - \eta\left(\boldsymbol{x}_i - \boldsymbol{x}_s\right)\right)\phi_j \tag{1.1}$$

Note that η can be less than 1 in the case where the PSF is oversampled. Using multiple point sources s sharing the same PSF, but centred on various coordinates $\boldsymbol{x}_s$, and neglecting the correlation of noise between pixels, we can derive the components of $\boldsymbol{\phi}$ that provide the best fit (in the least-square sense) to the point source images by minimising the cost function:

$$E(\boldsymbol{\phi}) = \chi^2(\boldsymbol{\phi}) = \sum_s \sum_{i\in\mathcal{D}_s} \frac{\left(p_i - f_s\phi'_i(\boldsymbol{x}_s)\right)^2}{\sigma_i^2}, \tag{1.2}$$

where f_s is the integrated flux of point source s, p_i the pixel intensity (number of counts in ADUs) recorded above the background at image pixel i, and $\mathcal{D}_s$ the set of pixels around s. In the variance estimate of pixel i, σ_i^2, we identify three contributions:

$$\sigma_i^2 = \sigma_\mathrm{b}^2 + \frac{p_i}{g} + \left(\alpha\, p_i\right)^2 , \tag{1.3}$$

where σ_b^2 is the pixel variance of the local background, p_i/g, where g is the detector gain in e^-/ADU (which must have been set appropriately before running **SExtractor**, is the variance contributed by photons from the source itself. The third term in equation (1.3) will generally be negligible except for high p_i values; the α factor accounts for pixel-to-pixel uncertainties in the flat-fielding, variation of the intra-pixel response function, and apparent fluctuations of the PSF due to interleaved "micro-dithered" observations[41] or lossy image resampling. The value of α is set by user with the `PSF_ACCURACY` configuration parameter. Depending on image quality, suitable values for `PSF_ACCURACY` range from less than one thousandth to 0.1 or even more. The default value, `0.01`, should be appropriate for typical CCD images.

[41] Micro-dithering consists of observing n^2 times the same field with repeated $1/n$ pixel shifts in each direction to provide properly sampled images despite using large pixels. Although the observed frames can in principle be recombined with an *interleaving* reconstruction procedure, changes in image quality from exposure to exposure may often lead to jaggies (artifacts) along gradients of source profiles, as can sometimes be noticed in DeNIS or WFCAM images.

The flux f_s is measured by integrating over a defined aperture, which defines the normalisation of the PSF. Its diameter must be sufficiently large to prevent the measurement from being too sensitive to centering or pixelisation effects, but not excessively large to avoid too strong S/N degradation and contamination by neighbours. In practice, a $\approx 5''$ diameter provides a fair compromise with good seeing images (PSF FWHM $< 1.2''$), but smaller in very crowded fields.

Interpolating the PSF model - As we saw, one of the main interests of interpolating the PSF model in direct space is that it involves only a limited number of PSF "pixels". However, as in any image resampling task, a compromise must be found between the perfect Shannon interpolant (unbounded sinc function), and simple schemes with excessive smoothing and/or aliasing properties like bi-linear interpolation ("tent" function) *[13]*. Experimenting with the **SWarp**[39] image resampling prototype *[14]*, we found that the Lanczos4 interpolant

$$h(x) = \begin{cases} 1 & x = 0 \\ \mathrm{sinc}(x)\,\mathrm{sinc}\,(x/4) & 0 < |x| \leq 4 \\ 0 & |x| > 4 \end{cases}, \tag{1.4}$$

where $\mathrm{sinc}(x) = \sin(\pi x)/(\pi x)$[42], provides reasonable compromise: the kernel footprint is 8 PSF pixels in each dimension, and the modulation transfer function is close to flat up to $\approx 60\%$ of the Nyquist frequency (Fig. 1.5). A typical minimum of 2 to 2.5 pixels per PSF FWHM is required to sample an astronomical image without generating significant aliasing *[15]*. Consequently, an appropriate sampling step for the PSF model would be $1/4^{\mathrm{th}}$ of the PSF FWHM. This is automatically done in **PSFEx**, when the `PSF_SAMPLING` configuration parameter is set to 0 (the default). The PSF sampling step may be manually adjusted (in units of image pixels) by simply setting `PSF_SAMPLING` to a non-zero value.

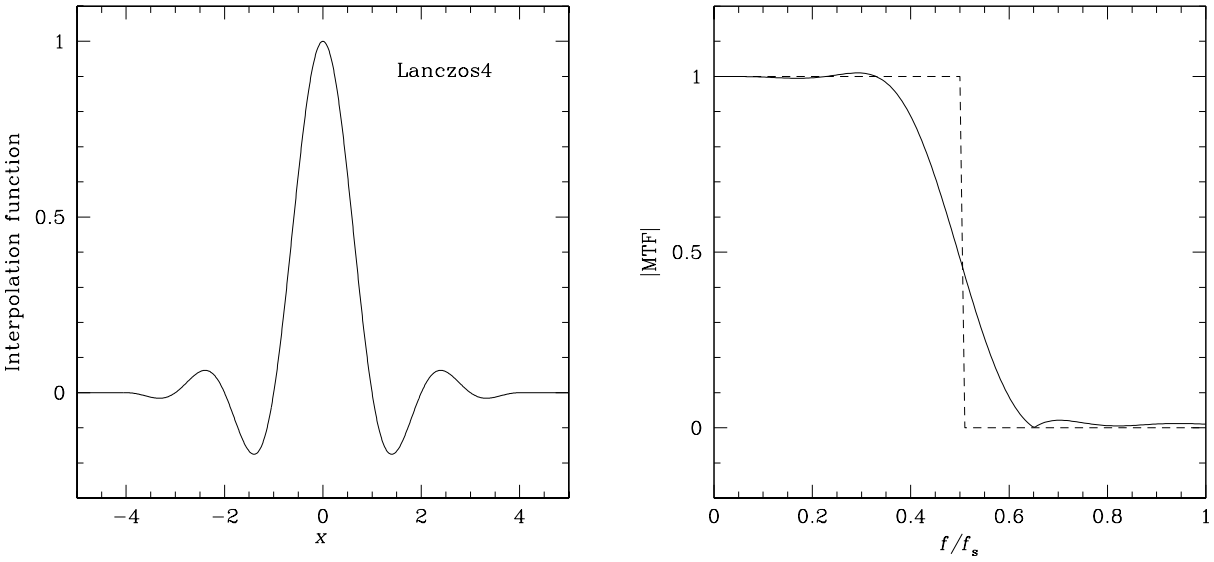

Fig. 1.5: The Lanczos4 interpolant in one dimension (left), and its modulation transfer function (right).

Regularisation - For $\eta \gg 1$, the system of equations obtained by minimising equation (1.2) becomes ill-conditioned and requires regularisation *[16]*. Our experience with **PSFEx** shows that the solutions obtained over the domain of interest for astronomical imaging ($\eta \leq 3$) are robust in practice, and that regularisation is generally not needed. However, it may happen, especially with infrared detectors, that samples of undersampled point sources are contaminated by image artifacts; and solutions computed with equation (1.2) become unstable. We therefore added a simple Tikhonov regularisation scheme to the cost function:

$$E(\boldsymbol{\phi}) = \chi^2(\boldsymbol{\phi}) + \|\mathbf{T}\boldsymbol{\phi}\|^2. \tag{1.5}$$

In image processing problems the (linear) Tikhonov operator T is usually chosen to be a high-pass filter to favour "smooth" solutions. **PSFEx** adopts a slightly different approach by reducing T to a scalar weight $1/\sigma_\phi^2$ and performing a procedure in two steps.

[39] http://astromatic.net/software/swarp

[42] This is the definition of the *normalised* sinc function, which should not be confused with the un-normalised definition of $\mathrm{sinc}(x) = \sin(x)/x$.

1. **PSFEx** makes a first rough estimate of the PSF by simply shifting point sources to a common grid and computing a median image $\boldsymbol{\phi}^{(0)}$. With undersampled data this image represents a smooth version of the real PSF.

2. Instead of fitting directly the model to pixel values, **PSFEx** fits the difference $\Delta\boldsymbol{\phi}$ between the model and $\boldsymbol{\phi}^{(0)}$. $E(\boldsymbol{\phi})$ becomes

$$E(\boldsymbol{\phi}) = \sum_s \sum_{i\in\mathcal{D}_s} \frac{\left[p_i - f_s\left(\phi_i'^{(0)}(\boldsymbol{x}_s) + \Delta\phi_i'(\boldsymbol{x}_s)\right)\right]^2}{\sigma_i^2} + \sum_j \frac{\Delta\phi_j^2}{\sigma_\phi^2}. \tag{1.6}$$

Minimising equation (1.6) with respect to the $\Delta\phi_j$'s comes down to solving the system of equations

$$\begin{aligned} 0 = & \frac{\partial E}{\partial \Delta\phi_k} \\ = & 2\, f_s \sum_s \sum_{i\in\mathcal{D}_s} \frac{1}{\sigma_i^2} h_s\left(\boldsymbol{x}_k - \eta\left[\boldsymbol{x}_i - \boldsymbol{x}_s\right]\right) \\ & \times \left(f_s \sum_j h_s\left(\boldsymbol{x}_j - \eta\left[\boldsymbol{x}_i - \boldsymbol{x}_s\right]\right)\left(\phi_j^{(0)} + \Delta\phi_j\right) - p_i\right) \\ & + \frac{2}{\sigma_\phi^2}\,\Delta\phi_k\,. \end{aligned}$$

In practice the solution appears to be fairly insensitive to the exact value of σ_ϕ except with low signal-to-noise conditions or contamination by artifacts. $\sigma_\phi \approx 10^{-2}$ seems to provide a good compromise by bringing efficient control of noisy cases but no detectable smoothing of PSFs with good data and high signal-to-noise.

The system in equation (1.7) is solved by **PSFEx** in a single pass. Much of the processing time is actually spent in filling the normal equation matrix, which would be prohibitive for large PSFs if the sparsity of the design matrix were not put to contribution to speed up computations.

Gauss-Laguerre basis

The pixel basis is quite a "natural" basis for describing in tabulated form bandwidth-limited PSFs with arbitrary shapes. But in a majority of cases, more restrictive assumptions can be made about the PSF that allow the model to be represented with a smaller number of components, e.g. a bell-shaped profile, a narrow scale range... Less basis vectors make for more robust models. For close-to-Gaussian PSFs, the Gauss-Laguerre basis is a sensible choice.

The Gauss-Laguerre basis is selected by setting `BASIS_TYPE` to be `GAUSS-LAGUERRE`. The Gauss-Laguerre functions, also known as *polar shapelets* in the weak-lensing community *[4]* provide a "natural" orthonormal basis for broadly Gaussian profiles:

$$\psi_{n,m}(r,\theta) = \frac{(-1)^{(n-|m|)/2}}{\sqrt{\pi}\,\sigma}\sqrt{\frac{[(n-|m|)/2]!}{[(n+|m|)/2]!}}\left(\frac{r}{\sigma}\right)^{|m|}\exp\left[-\frac{1}{2}\left(\frac{r}{\sigma}\right)^2 - im\theta\right] L^{|m|}_{(n-|m|)/2}\left(\frac{r^2}{\sigma^2}\right),$$

where σ is a typical scale for r, $(n-|m|)/2 \in \mathbb{N}$ and $L_n^k(x)$ is the associated Laguerre polynomial

$$\begin{aligned} L_n^k(x) = & \frac{1}{n!}\,x^{-k}\,\mathrm{e}^x\,\frac{\mathrm{d}^n}{\mathrm{d}x^n}\left(x^{n+k}e^{-x}\right) \\ = & \sum_{j=0}^{n}\frac{(k+n)!}{j!\,(n-j)!\,(j+k)!}\,(-x)^j. \end{aligned}$$

The number of shapelet vectors with $n \le n_{\max}$ is

$$N_{\max} = \frac{(n_{\max}+1)\,(n_{\max}+2)}{2}\,.$$

Shapelet decompositions with finite $n \le n_{\max}$ are only able to probe a restricted range of scales. *[17]* quotes $r_{\min} = \sigma/\sqrt{n_{\max}+1}$ and $r_{\max} = \sigma\sqrt{n_{\max}+1}$ as the standard deviation of the central lobe and the whole

shapelet profile, respectively (so that σ is the geometric mean of $r_{\min}$ and $r_{\max}$). In practice, the diameter of the circle enclosing the region where images can be fitted with shapelets is only about $\approx 2.5\, r_{\max}$. Hence modelling accurately both the wings and the core of PSFs with a unique set of shapelets requires a very large number of shapelet vectors, typically several hundreds.

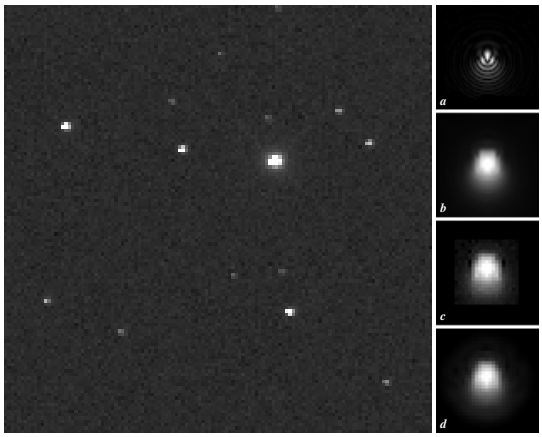

Fig. 1.6: *Left*: part of a simulated star field image with strong undersampling. *Right, from top to bottom*: (a) simulated optical PSF, (b) simulated PSF convolved by the pixel response, (c) PSF recovered by **PSFEx** at 4.5 times the image resolution from a random sample of 212 stars extracted in the simulated field above, using the *"pixel" vector basis*, and (d) PSF recovered using the *"shapelet" basis* with $n_{\max} = 16$.

Other bases

With the `BASIS_TYPE FILE` option, **PSFEx** offers the possibility to use an external image vector basis. The basis should be provided as a FITS datacube (the 3$^{\rm rd}$ dimension being the vector index), and the file name given to **PSFEx** with the `BASIS_NAME` parameter. External bases do not need to be normalised.

1.5.4 Managing PSF variations

Few imaging systems have a perfectly stable PSF, be it in time or position: for most instruments the approximation of a constant PSF is valid only on a small portion of an image at a time. Position-dependent variations of the PSF on the focal plane are generally caused by optics, and exhibit a smooth behaviour which can be modelled with a low-order polynomial.

The most intuitive way to generate variations of the PSF model is to apply some warping to it (enlargement, elongation, skewness, ...). But this description is not appropriate with **PSFEx** because of the non-linear dependency of PSF vector components towards warping parameters. Instead, one can extend the formalism of equation (1.6) by describing the PSF as a variable, linear combination of PSF vectors $\boldsymbol{\phi}_c$; each of them associated to a basis function X_c of some parameter vector $\boldsymbol{p}$ like image coordinates:

$$E(\boldsymbol{\phi}) = \sum_s \sum_{i\in\mathcal{D}_s} \frac{\left(p_i - f_s \sum_c X_c(\boldsymbol{p}) \left(\phi'^{(0)}_{ci}(\boldsymbol{x}_s) + \Delta\phi'_{ci}(\boldsymbol{x}_s)\right)\right)^2}{\sigma_i^2} + \sum_j \sum_c \frac{\Delta\phi^2_{cj}}{\sigma^2_\phi} \,. \tag{1.7}$$

The basis functions X_c in the current version of **PSFEx** are limited to simple polynomials of the components of $\boldsymbol{p}$. Each of these components p_l belongs to a "PSF variability group" $g = 0, 1, ..., N_g$, such that

$$X_c(\boldsymbol{p}) = \prod_{g\le N_g} \left(\prod_{(\sum_{l\in\Lambda_g} d_l)\le D_g} p_l^{d_l} \right) , \tag{1.8}$$

where Λ_g is the set of l's that belongs to the distortion group g, and $D_g \in \mathbb{N}$ is the polynomial degree of group g. The polynomial engine of **PSFEx** is the same as the one implemented in the **SCAMP**[40] software *[18]* and can use

[40] http://astromatic.net/software/scamp

any set of **SExtractor** and/or FITS header parameters as components of p. Although PSF variations are more likely to depend essentially on source position on the focal plane, it is thus possible to include explicit dependency on parameters such as telescope position, time, source flux (Fig. 1.8) or instrument temperature.

The p_l components are selected using the `PSFVAR_KEYS` configuration parameter. The arguments can be names of **SExtractor** measurements, or keywords from the image FITS header representing numerical values. FITS header keywords must be preceded with a colon (`:`), like in `:AIRMASS`. The default `PSFVAR_KEYS` are `X_IMAGE,Y_IMAGE`.

The `PSFVAR_GROUPS` configuration parameters must be filled in in combination with the `PSFVAR_KEYS` to indicate to which PSF variability group each component of $\boldsymbol{p}$ belongs. The default for `PSFVAR_GROUPS` is `1,1`, meaning that both `PSFVAR_KEYS` belong to the same unique PSF variability group. The polynomial degrees D_g are set with `PSFVAR_DEGREES`. The default `PSFVAR_DEGREES` is `2`. In practice, a third-degree polynomial on pixel coordinates (represented by 20 PSF vectors) should be able to map PSF variations with good accuracy on most exposures (Fig. 1.7).

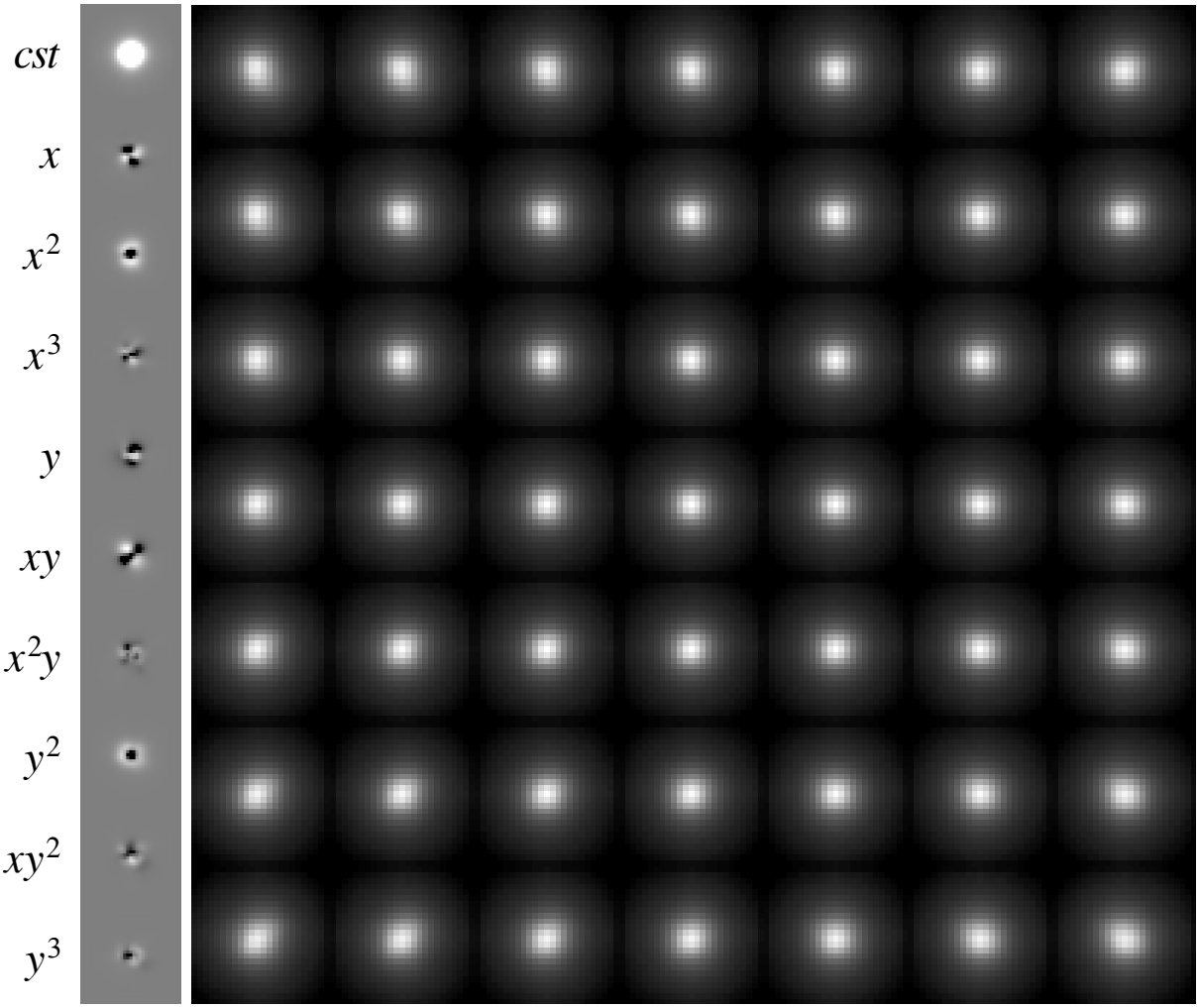

Fig. 1.7: Example of PSF mapping as a function of pixel coordinates in **PSFEx**. *Left*: PSF component vectors for each polynomial term derived from the CFHTLS-deep "D4" r-band stack observed with the MEGACAM camera. A third-degree polynomial was chosen for this example. Note the prominent variation of PSF width with the square of the distance to the field centre. *Right*: reconstruction of the PSF over the 1° field of view (the grey scale has been slightly compressed to improve clarity).

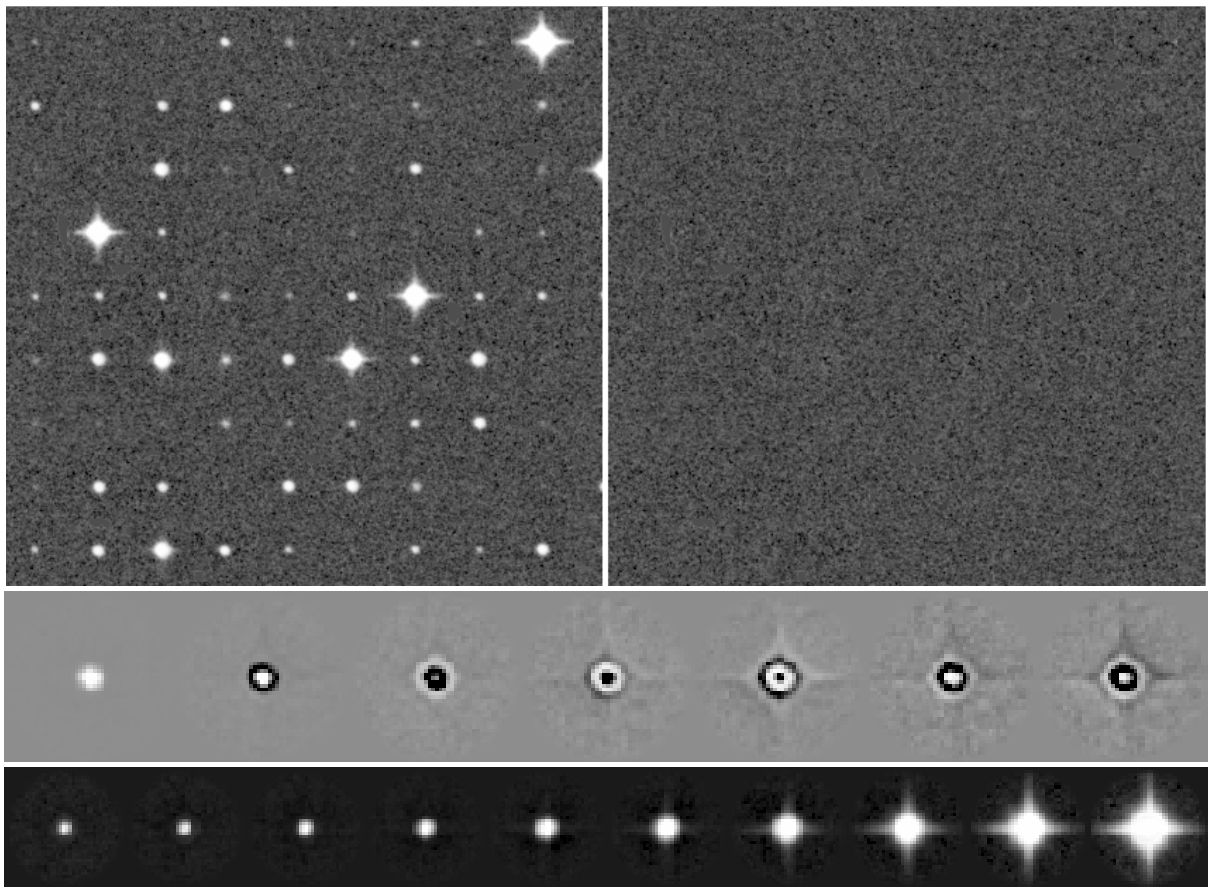

Fig. 1.8: Example of PSF mapping on images from a non-linear imaging device. 1670 point sources from the central 4096×4096 pixels of a photographic scan (SERC J #418 survey plate, courtesy of J. Guibert, CAI, Paris observatory) were extracted using **SExtractor**, and their images run through **PSFEx**. A sample is shown at the *top-left*. The PSF model was given a $6^{\rm th}$ degree polynomial dependency on the instrumental magnitude measured by **SExtractor** (MAG_AUTO). *Middle*: PSF components derived by **PSFEx**. *Bottom*: reconstructed PSF images as a function of decreasing magnitude. *Top-right*: sample residuals after subtraction of the PSF-model.

1.5.5 Quality assessment

Maintaining a certain level of image quality, and especially PSF quality, by identifying and rejecting "bad" exposures, is a critical issue in large imaging surveys. Image control must be automated, not only because of the sheer quantity of data in modern digital surveys, but also to ensure an adequate level of consistency. Automated PSF quality assessment is traditionally based upon point source FWHM and ellipticity measurements. Although this is certainly efficient for finding fuzzy or elongated images, it cannot make the distinction between e.g. a defocused image and a moderately bad seeing.

PSFEx can trace out the apparition of specific patterns using customized basis functions. Moreover, **PSFEx** implements a series of generic quality measurements performed on the PSF model as it varies across the field of view. The main set of measurements is done in PSF pixel space (with oversampling factor η) by comparing the actual PSF model vector $\boldsymbol{\phi}$ with a reference PSF model $\rho(\boldsymbol{x}')$. We adopt as a reference model the elliptical Moffat function *[2]* that fits best (in the chi-square sense) the model (1.9):

$$\rho(\boldsymbol{x}') = I_0 \left(1 + ||\mathrm{A}(\boldsymbol{x}' - \boldsymbol{x}'_c)||^2\right)^{-\beta}, \tag{1.9}$$

with

$$\mathrm{A} = \frac{4}{\eta}\left(2^{-\frac{1}{\beta}} - 1\right) \begin{pmatrix} \cos\theta/W_{\max} & \sin\theta/W_{\max} \\ -\sin\theta/W_{\min} & \cos\theta/W_{\min} \end{pmatrix},$$

where I_0 is the central intensity of the PSF, $\boldsymbol{x}'_c$ the central coordinates (in PSF pixels), $W_{\max}$, the PSF FWHM along the major axis, $W_{\min}$ the FWHM along the minor axis, and θ the position angle (6 free parameters). As a matter of fact, the Moffat function provides a good fit to seeing-limited images of point-sources, and to a lesser degree, to the core of diffraction-limited images for instruments with circular apertures *[19]*: in most imaging surveys, the "correct" instrumental PSF will be very similar to a Moffat function with low ellipticity.

Since **PSFEx** is meant to deal with significantly undersampled PSFs, another fit — which we call "pixel-free" — is also performed, where the Moffat model is convolved with a square top-hat function the width of a physical pixel, as an approximation to the real intra-pixel response function. The width of the pixel is set to 1 in image sampling step units by default, which corresponds to a 100% fill-factor. It can be changed using the `PSF_PIXELSIZE` configuration parameter. Future versions of **PSFEx** might propose more sophisticated models of the intra-pixel response function.

The (non-linear) fits are performed using the LevMar implementation of the Levenberg-Marquardt algorithm *[20]*. They are repeated at regular intervals on a grid of PSF parameter vectors $\boldsymbol{p}$, generally composed of the image coordinates $\boldsymbol{x}$), but with possible additional parameters such as time, observing conditions, etc. The density of the grid may be adjusted using the PSFVAR_NSNAP configuration parameter. The default value for `PSFVAR_NSNAP` is `9` (snapshots per component of $\boldsymbol{p}$). Larger numbers can be useful to track PSF variations on large images with greater accuracy; but beware of the computing time, which increases as the total number of PSF snapshots (grid points).

The average FWHM $(W_{\max} + W_{\min})/2$, ellipticity $(W_{\max} - W_{\min})/(W_{\max} + W_{\min})$ and β parameters derived from the fits provide a first set of local IQ estimators (Fig. 1.9). The second set is composed of the so-called *residuals* index

$$r = 2\frac{\sum_i \left(\phi_i + \rho'(\boldsymbol{x}'_i)\right)\left|\phi_i - \rho'(\boldsymbol{x}'_i)\right|}{\sum_i \left(\phi_i + \rho'(\boldsymbol{x}'_i)\right)^2} \tag{1.10}$$

and the *asymmetry* index

$$\alpha = 2\frac{\sum_i \left(\phi_i + \phi_{N-i}\right)\left|\phi_i - \phi_{N-i}\right|}{\sum_i \left(\phi_i + \phi_{N-i}\right)^2}, \tag{1.11}$$

where the ϕ_{N-i}'s are the point-symmetric counterparts to the ϕ_i components.

1.6 Examples

In the following, examples of use of PSFEx are given, together with commented command lines.

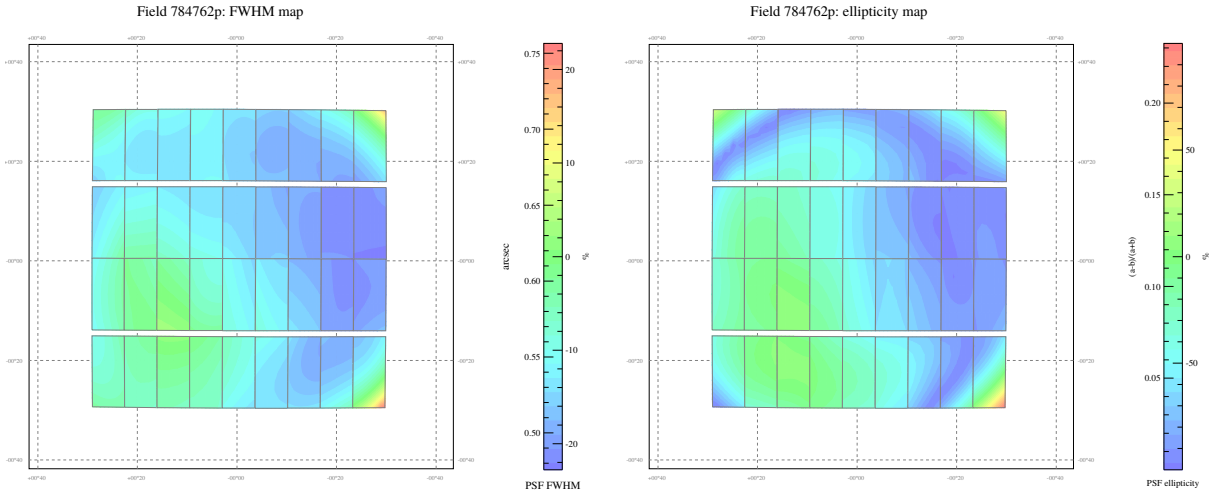

Fig. 1.9: FWHM map (*left*) and ellipticity map (*right*) generated by **PSFEx** from a CFHTLS-Wide exposure. The maps and the individual Megaprime CCD footprints on the sky are presented in gnomonic projection (north on top, east at left). PSF variations are modelled independently on each CCD using a 3$^{\rm rd}$ degree polynomial (see text).

1.6.1 Hands-on example 1

Let us consider a V band FITS image RX_J2202-19_V.fits and its weight map RX_J2202-19_V.weight.fits. We wish to fit all the galaxies of the image with a galaxy model using SExtractor, which requires computing a model of the PSF first.

We must first run SExtractor on this image to obtain a temporary catalogue in FITS_LDAC format that contains small sub-images from which the PSF model will be extracted. For this, we define in a SExtractor parameter file — let us call it prepsfex.param — the parameters required for the use of PSFEx:

```
X_IMAGE
Y_IMAGE
FLUX_RADIUS
FLUX_APER(1)
FLUXERR_APER(1)
ELONGATION
FLAGS
SNR_WIN
VIGNET(35,35)
```

TBW

1.6.2 Example 2: very wide photographic plate

TBW

1.6.3 Example 3: unfocused instrument

TBW

1.7 Frequently Asked Questions

Skeptical Sam doesn't have time to test software extensively but is always keen on asking aggressive questions to the author to find out if a program could fit his needs.

PSFEx represents PSFs as an array of tabulated values! Can it really deal with undersampled images? Isn't it too noisy?

PSFEx was designed from the ground up to deal with undersampled images and arbitrary PSFs. Although the PSF "model" in PSFEx is actually a small image, it is sampled at a different step than the original pixels: more finely for undersampled observations, and more coarsely for oversampled observations, to avoid any loss and redundancy of information. Despite built-in regularisation, PSF models reconstructed on the pixel basis can indeed be noisy if the number of selected stars is small. This can be circumvented to some extent by using *ad hoc* basis to solve for the PSF model coefficients.

I heard that PSFEx has been developed almost 12 years ago, and has been used for production at TERAPIX for many years. Why have you waited until 2010 for releasing it to the general community?

PSFEx was originally developed for doing PSF-fitting crowded-field photometry with SExtractor. However I was not very happy with the way it worked, as SExtractor's detection and deblending engine is not meant to deal with crowded star fields. The current release of PSFEx is made in the framework of the EFIGI [2]_ and DES [3]_ projects, as a support tool for galaxy model-fitting.

I would like to use PSFEx to generate PSF models for weak-lensing analyses. Is it the right tool for that?

Simulations of 1h exposures with a 4m optical telescope and sub-arcsecond seeing show that ellipticities of galaxies with a Signal-to-Noise Ratio SNR>20 can be recovered with a level of systematics below 10^{-3} using PSFEx models, even in the presence of significant amounts of coma and astigmatism. This is for constant PSFs. Tests with variable PSFs are ongoing.

1.8 Troubleshooting

TBW

1.9 Acknowledging PSFEx

Please use the following reference *[1]*:

> Bertin 2011: Automated Morphometry with SExtractor and PSFEx, ASP Conference Series, Vol. 442, 2011, Ian N. Evans, Alberto Accomazzi, Douglas J. Mink, and Arnold H. Rots, eds., p. 435

1.10 Acknowledgements

The authors would like to thank Mireille Dantel, Frédéric Magnard, Chiara Marmo, Gregory Sémah, and the TERAPIX team at IAP for testing and support on image quality indices, Shantanu Desai, Tony Darnell, Greg Daues, Joe Mohr and the Dark Energy Survey Management team at University of Illinois and NCSA, for testing and support on PSF homogenisation, Philippe Delorme for his contributions to PSF-fitting in SExtractor, Valérie de Lapparent , Pascal Fouqué, and Jason Kalirai for extensive testing and suggestions, Mark Calabretta for his great astrometric library, Manolis Lourakis for making his LevMar library public, Akim Demaille for his help with the autotools, and Gary Mamon for his careful reading and corrections to the manuscript.

1.11 Appendices

1.11.1 `.psf` file format description

PSF models are FITS binary tables[61] with a single row, containing PSF image components. PSF files derived from MEF (Multi-Extension FITS)[62] images are themselves MEF file, with one `PSF_DATA` extension per input image

[61] https://archive.stsci.edu/fits/fits_standard/node67.html
[62] http://www.stsci.edu/hst/HST_overview/documents/datahandbook/intro_ch23.html

extension.

A sample of a PSF binary table extension header is given below:

```
XTENSION= 'BINTABLE'           / THIS IS A BINARY TABLE (FROM THE LDACTOOLS)
BITPIX  =                     8 /
NAXIS   =                     2 /
NAXIS1  =                 25000 / BYTES PER ROW
NAXIS2  =                     1 / NUMBER OF ROWS
PCOUNT  =                     0 / RANDOM PARAMETER COUNT
GCOUNT  =                     1 / GROUP COUNT
TFIELDS =                     1 / FIELDS PER ROWS
EXTNAME = 'PSF_DATA'           / TABLE NAME
LOADED  =                  7256 / Number of loaded sources
ACCEPTED=                  6637 / Number of accepted sources
CHI2    =           1.17986252 / Final reduced Chi2
POLNAXIS=                     2 / Number of context parameters
POLGRP1 =                     1 / Polynom group for this context parameter
POLNAME1= 'X_IMAGE '           / Name of this context parameter
POLZERO1=   2.046996443272E+03 / Offset value for this context parameter
POLSCAL1=   4.074312777519E+03 / Scale value for this context parameter
POLGRP2 =                     1 / Polynom group for this context parameter
POLNAME2= 'Y_IMAGE '           / Name of this context parameter
POLZERO2=   2.047786695004E+03 / Offset value for this context parameter
POLSCAL2=   4.077873875618E+03 / Scale value for this context parameter
POLNGRP =                     1 / Number of context groups
POLDEG1 =                     3 / Polynom degree for this context group
PSF_FWHM=           2.23724842 / PSF FWHM in image pixels
PSF_SAMP=           0.47601029 / Sampling step of the PSF data
PSFNAXIS=                     3 / Dimensionality of the PSF data
PSFAXIS1=                    25 / Number of element along this axis
PSFAXIS2=                    25 / Number of element along this axis
PSFAXIS3=                    10 / Number of element along this axis
TTYPE1  = 'PSF_MASK'           / Tabulated PSF data
TFORM1  = '6250E   '
TDIM1   = '(25, 25, 10)'
END
```

The file content is largely self-describing. Please note that

- The `TDIM1` keyword in the extension header contains the total number of components (conditioned by the `PSFVAR_DEGREES` configuration parameter) and the dimensions of the tabulated PSF (see the `PSF_SIZE` configuration parameter).
- Every context parameter (e.g., X_j) is rescaled before being used as a polynomial variable (x_j):

$$x_j = \frac{X_j - \mathrm{POLZERO}j}{\mathrm{POLSCAL}j} \tag{1.12}$$

- The PSF FWHM reported by the `PSF_FWHM` keyword is actually derived from the mode of the half-light diameter[63] distribution of the input source image sample.

[63] https://en.wikipedia.org/wiki/Effective_radius

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Bibliography

[1] E. Bertin. Automated Morphometry with SExtractor and PSFEx.[43] In I. N. Evans, A. Accomazzi, D. J. Mink, and A. H. Rots, editors, *Astronomical Data Analysis Software and Systems XX*, volume 442 of Astronomical Society of the Pacific Conference Series, 435. July 2011.

[2] A. F. J. Moffat. A Theoretical Investigation of Focal Stellar Images in the Photographic Emulsion and Application to Photographic Photometry.[44] *A&A*, 3:455, 1969.

[3] T. Darnell, E. Bertin, M. Gower, C. Ngeow, S. Desai, J. J. Mohr, D. Adams, G. E. Daues, M. Gower, C. Ngeow, S. Desai, C. Beldica, M. Freemon, H. Lin, E. H. Neilsen, D. Tucker, L. A. N. da Costa, L. Martelli, R. L. C. Ogando, M. Jarvis, and E. Sheldon. The Dark Energy Survey Data Management System: The Coaddition Pipeline and PSF Homogenization.[45] In D. A. Bohlender, D. Durand, and P. Dowler, editors, *Astronomical Data Analysis Software and Systems XVIII*, volume 411 of Astronomical Society of the Pacific Conference Series, 18. September 2009.

[4] R. Massey and A. Refregier. Polar shapelets.[46] *MNRAS*, 363:197–210, 2005.

[5] M. Jarvis, P. Schechter, and B. Jain. Telescope Optics and Weak Lensing: PSF Patterns due to Low Order Aberrations.[47] *ArXiv e-prints*, 2008.

[6] P. B. Stetson. DAOPHOT - A computer program for crowded-field stellar photometry.[48] *PASP*, 99:191–222, 1987.

[7] P. Magain, F. Courbin, M. Gillon, S. Sohy, G. Letawe, V. Chantry, and Y. Letawe. A deconvolution-based algorithm for crowded field photometry with unknown point spread function.[49] *A&A*, 461:373–379, 2007.

[8] R. H. Lupton and J. E. Gunn. M13 - Main sequence photometry and the mass function.[50] *AJ*, 91:317–325, 1986.

[9] J. Anderson and I. R. King. Toward High-Precision Astrometry with WFPC2. I. Deriving an Accurate Point-Spread Function.[51] *PASP*, 112:1360–1382, 2000.

[10] K. J. Mighell. Stellar photometry and astrometry with discrete point spread functions.[52] *MNRAS*, 361:861–878, 2005.

[43] http://adsabs.harvard.edu/abs/2011ASPC..442..435B
[44] http://adsabs.harvard.edu/abs/1969A&A.....3..455M
[45] http://adsabs.harvard.edu/abs/2009ASPC..411...18D
[46] http://adsabs.harvard.edu/abs/2005MNRAS.363..197M
[47] http://adsabs.harvard.edu/abs/2008arXiv0810.0027J
[48] http://adsabs.harvard.edu/abs/1987PASP...99..191S
[49] http://adsabs.harvard.edu/abs/2007A&A...461..373M
[50] http://adsabs.harvard.edu/abs/1986AJ.....91..317L
[51] http://adsabs.harvard.edu/abs/2000PASP..112.1360A
[52] http://adsabs.harvard.edu/abs/2005MNRAS.361..861M

[11] RY Tsai and Thomas S Huang. Multiframe image restoration and registration.[53] *Advances in computer vision and Image Processing*, 1(2):317–339, 1984.

[12] T. R. Lauer. The Photometry of Undersampled Point-Spread Functions.[54] *PASP*, 111:1434–1443, 1999.

[13] George Wolberg. *Digital Image Warping*. IEEE Computer Society Press, Los Alamitos, CA, USA, 1st edition, 1994. ISBN 0818689447. URL: http://eu.wiley.com/WileyCDA/WileyTitle/productCd-0818689447,miniSiteCd-IEEE_CS2.html.

[14] E. Bertin, Y. Mellier, M. Radovich, G. Missonnier, P. Didelon, and B. Morin. The TERAPIX Pipeline.[55] In D. A. Bohlender, D. Durand, and T. H. Handley, editors, *Astronomical Data Analysis Software and Systems XI*, volume 281 of Astronomical Society of the Pacific Conference Series, 228. 2002.

[15] G. Bernstein. Advanced Exposure-Time Calculations: Undersampling, Dithering, Cosmic Rays, Astrometry, and Ellipticities.[56] *PASP*, 114:98–111, 2002.

[16] L. Pinheiro da Silva, M. Auvergne, D. Toublanc, J. Rowe, R. Kuschnig, and J. Matthews. Estimation of a super-resolved PSF for the data reduction of undersampled stellar observations. Deriving an accurate model for fitting photometry with Corot space telescope.[57] *A&A*, 452:363–369, 2006.

[17] A. Refregier. Shapelets - I. A method for image analysis.[58] *MNRAS*, 338:35–47, 2003.

[18] E. Bertin. Automatic Astrometric and Photometric Calibration with SCAMP.[59] In C. Gabriel, C. Arviset, D. Ponz, and S. Enrique, editors, *Astronomical Data Analysis Software and Systems XV*, volume 351 of Astronomical Society of the Pacific Conference Series, 112. July 2006.

[19] I. Trujillo, J. A. L. Aguerri, J. Cepa, and C. M. Gutiérrez. The effects of seeing on Sèrsic profiles.[60] *MNRAS*, 321:269–276, 2001.

[20] M.I.A. Lourakis. Levmar: levenberg-marquardt nonlinear least squares algorithms in C/C++. http://www.ics.forth.gr/~lourakis/levmar/, 2004. [Accessed on 31 Mar. 2015.]. URL: http://www.ics.forth.gr/~lourakis/levmar/.

[53] http://arxiv.org/abs/
[54] http://adsabs.harvard.edu/abs/1999PASP..111.1434L
[55] http://adsabs.harvard.edu/abs/2002ASPC..281..228B
[56] http://adsabs.harvard.edu/abs/2002PASP..114...98B
[57] http://adsabs.harvard.edu/abs/2006A&A...452..363P
[58] http://adsabs.harvard.edu/abs/2003MNRAS.338...35R
[59] http://adsabs.harvard.edu/abs/2006ASPC..351..112B
[60] http://adsabs.harvard.edu/abs/2001MNRAS.321..269T