*lens*fit

Lance Miller
L.Miller@physics.ox.ac.uk

March 20, 2011

Contents

1 Installing *lens*fit

1.1 64-bit installation

It is recommended that *lens*fit and associated libraries are compiled and run on a 64-bit machine in order to obtain sufficient memory.

1.2 Libraries

64-bit versions required for 64-bit installation.

fftw v.3 from http://www.fftw.org. The code is not compatible with fftw v.2.

cfitsio should be consistent with all recent versions of cfitsio from http://heasarc.nasa.gov/fitsio/fitsio.html (tested on v.3.001).

GSL (Gnu Scientific Library) from http://www.gnu.org/software/gsl/ Version 1.14 or later is required.

In addition, *lens*fit makes use of the astrometric library **swarp** [Bertin et al., 2002] [1]. The appropriate swarp modules from v2.17.1 have been included in the *lens*fit distribution (later versions **swarp** library routines are not compatible with the code bundled here).

The **fftw** and **swarp** libraries have installation options for these libraries to run multi-threaded. *Do not install these libraries with multi-threading enabled. lens*fit carries out its own multi-threaded function calls and does not need the libraries to be installed this way, and in fact setting the libraries to be multi-threaded causes conflicts which interfere with the installation of *lens*fit .

1.3 Compilers/platforms

All code is written in C. It has been tested on: gcc v.4.1.2 (Debian4.1.1-21) operating on a linux(Debian) PC; gcc v.4.0.1 on Apple MacBook Pro OS10.5.5 (Darwin9.5.0); Intel icc v10.1 on Apple MacPro OS 10.5.5 (Darwin9.5.0); and gcc version 4.1.2 on AMD (linuxRedHat4.1.2-14) among others.

1.4 Setting up the code

Before compiling, see the Appendix for discussion of issues such as setting the postage stamp size and ellipticity sampling. You also need to consider how to pass multiple images to *lens*fit - see section 2.3 for discussion of the filename conventions that currently form part of the handling of multiple images. The use of different filename conventions from those assumed in the code would require the code to be edited accordingly.

There are also some global variables at the start of the code. That need to be set for your particular installation, as follows (further details on the function of some of these is given later in the document).

There are four programs to be set up:

[1] http://www.astromatic.net/software/swarp

`makeglobalpsf` fits the position-dependent PSF for a multi-CCD mosaic camera.

`globalshifts` measures astrometric shifts between multiple exposures by cross-correlating the PSF models with the original stars.

`lensfit` carries out the galaxy shape measurement.

`readlensfit` carries out simple analysis of the results from `lensfit` , converting each probability distribution into a single mean shear estimate with an appropriate weight.

1.4.1 `makeglobalpsf`

Creates PSF polynomial coefficients globally for all images in a multi-CCD exposure. Module name: makeglobalpsf.cThe following `#define` statements should be set inside the code.

VERBOSE allows more output (if set to 1) or less output (if set to 0).

SUBTRACT_MEDIAN allows a constant median background to be subtracted from the data (if set to value 1) or not (if set to 0). Since we recommend that a variable background should already have been subtracted from the image data we expect this option would normally be turned off.

FILTER_WEIGHTS when set to 1 applies a median erosion filter to the weight image: this eliminates small "bad pixel" flags in the weight image if they coincide with the central region of a star. This is the normal mode of operation for CFHTLenS, as the CFHTLenS initial analysis could sometimes flag the bright central pixels of stars as being cosmic rays. This must be avoided at all costs, as it preferentially flags as bad those stars with sharp central pixels and thus introduces a systematic bias in the PSF shape.

WEIGHTS_REQUIRED will cause program execution to halt with an error message, when set to 1, if an expected weight image is not found.

NAVERAGE sets the number of stars to be used for averaging in each subregion of each individual CCD image (typical value 20).

WCS specifies that star positions are input as world coordinates (RA,dec) to be interpreted with the FITS header astrometric information.

USE_SWARP specifies that swarp astrometric information is to be used (in which case a scamp/swarp additional .head file must be supplied).

CORRECT_DISTORTION specifies that camera distortion correction is to be carried out. Normally this should be **turned off** (value 0) because the most accurate way of distortion correction is to distort the models inside `lensfit` to match the data, rather than trying to interpolate the data onto an undistorted pixel grid (value 1).

OUTPUT_STARS specifies if additional output FITS images of the stars are to be produced (centred and normalised). These are output to a single 3D FITS file.

GZIPPED specifies if any of the input images or weightfiles are gzipped.

WRITE_INDIVIDUAL_SURFACE when set to 1 will write out a FITS cube for every chip showing how the PSF varies with position.

minexposuretime specifies the minimum exposure time allowed for an image - images less than this time will not be used (needs exposure time keyword to be present in image FITS file headers).

CFHT/SINGLE/SUPRIME specify the camera mosaic format and filename conventions to be assumed. You may need to add an additional switch and a set of format and filename conventions for your particular survey.

1.4.2 `globalshifts`

Reads in the PSF model output by `makeglobalpsf` , then cross-correlates each star in the same star catalogue against the PSF model to find position offsets between the nominal star positions and the model. Depending on your camera and on how good your astrometric solution is, these offsets vary with position. `globalshifts` fits a 2D linear function to the variation and writes the fit coefficients onto the end of the same PSF coefficients files that were input (NB do not run this multiple times as successive fits will be continually added into the files, but only the first set are ever used).

Current module name: globalshifts.c

Set the following definitions at the start of the code (these are the same as `makeglobalpsf` and should be set the same).

VERBOSE allows more output (if set to 1) or less output (if set to 0).

SUBTRACT_MEDIAN allows a constant median background to be subtracted from the data (if set to value 1) or not (if set to 0). Since we recommend that a variable background should already have been subtracted from the image data we expect this option would normally be turned off.

FILTER_WEIGHTS when set to 1 applies a median erosion filter to the weight image: this eliminates small "bad pixel" flags in the weight image if they coincide with the central region of a star. This is the normal mode of operation for CFHTLenS, as the CFHTLenS initial analysis could sometimes flag the bright central pixels of stars as being cosmic rays. This must be avoided at all costs, as it preferentially flags as bad those stars with sharp central pixels and thus introduces a systematic bias in the PSF shape.

WEIGHTS_REQUIRED will cause program execution to halt with an error message, when set to 1, if an expected weight image is not found.

WCS specifies that star positions are input as world coordinates (RA,dec) to be interpreted with the FITS header astrometric information.

USE_SWARP specifies that swarp astrometric information is to be used (in which case a scamp/swarp additional .head file must be supplied)

GZIPPED specifies if any of the input images or weightfiles are gzipped.

minexposuretime specifies the minimum exposure time allowed for an image - images less than this time will not be used (needs exposure time keyword to be present in image FITS file headers).

CFHT/SINGLE/SUPRIME specify the camera mosaic format and filename conventions to be assumed. You may need to add an additional switch and a set of format and filename conventions for your particular survey.

1.4.3 `lensfit`

The main galaxy shape fitting code. At run time, it requires the information from the PSF files created by `makeglobalpsf` and `globalshifts` .

Current module name: tlensfit.c

VERBOSE controls the amount of output sent to STDOUT. Use a value 0 for normal non-verbose use. Use 1 for debugging/testing.

NUM THREAD should be set to the number of shared-memory cores available on your system (e.g. #define NUM THREAD 8 for a quad dual-core Mac Pro). Specifying a larger number than this is not helpful, as there is no speed gain but more memory is used. You may specify a smaller number of threads than there are processors, this will reduce the memory requirement by some amount, but obviously results in longer execution times.

MEMORY LIMIT should be set to the maximum core memory size, in MB, that you want the code to occupy. If this is smaller than would be required to process the entire input dataset, `lensfit` will attempt to split up the processing into smaller units, at the expense of needing to make multiple reads of the data. So this size should be as large as possible. However, you should allow some room for the operating system and other processes to run on your machine. If you allow `lensfit` to occupy too much RAM then your machine could hang or `lensfit` could crash. How close you can go to the machine limit depends on how much swap space is available and how good your machine's virtual memory management is. On large datasets, during run-time `lensfit` dynamically allocates memory and frees it when it is no longer needed: whether this process works efficiently depends on the swap space management. A catalogue of 10^5 galaxies each with 7 exposures, measured with postage stamp sizes of 48×48 pixels, requires approximately $5000 + 1500N$ MB for N threads, to process all data in a single pass. Allocating less memory would result in `lensfit` making multiple reads of the input images and hence having a somewhat slower overall processing time (depending on how fast your i/o is).

GZIPPED specifies whether the input images are gzipped or not. A value zero means it is assumed nothing is gzipped. A value of unity indicates that the weight images only are gzipped. A value of 2 indicates that both weight and data images are gzipped.

WEIGHTS_REQUIRED specifies whether good/bad pixel images are required (1) or optional (0). If weight images are required but are not found at run-time, `lensfit` will exit. If weight images are not required execution will continue but with all pixels being flagged as "good".

minexposuretime sets the minimum accepted exposure time if the appropriate exposure time keyword exists in the image FITS header.

minexposurefraction allows you to specify some minimum fraction of exposures must be used for each galaxy (set to zero for no cut).

CFHT/SINGLE/SUPRIME specify the camera mosaic format and filename conventions to be assumed. You may need to add an additional switch and a set of format and filename conventions for your particular survey.

POSTAGE_STAMP_SIZE sets the postage stamp size (usually set to a value of 48). If this is larger than the PSF postage stamp size the PSF will be padded with zeros.

DEBLENDING_REJECTION rejects blended-looking galaxies and galaxies that are too large for the postage stamp, if set to 1.

WCS should be set to 1 if you want to input galaxy positions in WCS coordinates. This is by far the easiest method of inputting positions and is strongly recommended.

USE_SWARP specifies use of the swarp astrometric information, and must be set consistently with what was used for the PSF files (see CORRECT_DISTORTION below).

CORRECT DISTORTION defines whether you want to correct the images for optical and atmospheric distortion. In normal use this should be set to 2, which instructs `lensfit` to apply the distortion correction to the models (recommended). Other options are 0 (no correction applied) or 1 (data are corrected by interpolation). Swarp header information is required for distortion correction.

GLOBAL_PSF determines whether to expect PSF input that has been created by `makeglobalpsf` or by the old makepsfsurface PSF code (set to value 1 for normal use with `makeglobalpsf`).

PSF_STAR_LIMIT, in the case of makepsfsurface only (GLOBAL_PSF=0), sets a hard lower limit to the number of stars acceptable in each subregion of the image. Subregions with fewer stars will be flagged as bad and no shapes measured. This option is ignored if GLOBAL PSF=1.

psfelimit set an upper limit on the maximum acceptable PSF ellipticity. Set a value 1. to avoid any cut.

MODELS_3D is normally set to 0 for 2D models, experimental 3D models can be selected with value 1 (but need more memory and a lot of time to compute the models).

NITER should be set to 1 for normal operation of `lensfit` . Setting a value of 4 allows some simulations to be created internal to `lensfit` , for testing (NB the simulations option has not been tested since several changes to the code and probably no longer works!).

MODEL_OVERSAMPLING sets whether to create oversampled models (1) or not (0). Oversampled models should be more accurate, although there is an approximation step in converting these to the observed sampling. Should use value 1.

SUBTRACT_MEDIAN allows a constant median background to be subtracted from the data (if set to value 1) or not (if set to 0). Since we recommend that a variable background should already have been subtracted from the image data we expect this option would normally be turned off.

SUBPIXEL_SHIFT controls whether sub-pixel shifts are allowed when coadding likelihoods from multiple input images. For reasonably well-sampled data this does yield a significant improvement and should be set to a value 1. For undersampled data the situation is less clear and experimentation may be required.

PRINT_SHIFTS specifies whether an output ascii file is printed containing the astrometric shifts between multiple exposures that is calculated internally by `lensfit` (1=yes, 0 =no).

WRITE_SUMMED_LIKELIHOODS specifies whether additional output files are created which are needed for iteratively calculating the ellipticity-size prior.

WRITE_CHISQ_SURFACES set to 0 for normal use. When set to 1, writes out a FITS cube showing the chi-squared surfaces as a function of galaxy position for each input galaxy. To avoid ridiculously large files being generated, you must restrict the input object catalogue to a small number (e.g. less than 2000).

WRITE_MODELS set to 0 for normal use. When set to 1, if environment variable MODELFILE is also set, a FITS cube of selected galaxy models is written out for testing.

NTEST_LIMIT specifies whether some number of image postage stamps are to be written out to a file (useful for testing, set to 0 for normal use). Choose a value not too large (< 1000).

systematic set some level of allowed systematic error, expressed as a fraction, when calculating the $\chi-$squared goodness-of-fit (a larger value of systematic allows more galaxies to be classed as acceptably-fit). Not currently enabled.

1.4.4 `readlensfit`

Post-processing code, to read the `lensfit` output FITS table and calculate shear estimates and errors.

Current code module: readlensfits.c

WRITE_POSTERIOR_SURFACES allows the user to write out FITS cubes of shear likelihood surfaces for a subset of galaxies. Modify the code to make the selection you want, don't try to write out too many or the file will be too large.

1.5 Building the *lens*fit installation

The build steps are summarised in the INSTALL file. First, install 64-bit versions of the libraries as described in section 1.2. Then, make any changes to the code that are required for your particular survey as described in section 1.4. The code is written in C, you can specify your choice of C compiler using an environment variable or you can allow it to default to your system's compiler (usually gcc). Compiler option -O3 (speed optimisation in gcc) is invoked. You will need to set environment variables CFLAGS and LDFLAGS as described in INSTALL. Depending on your machine you may need to set explicitly flags such as `-m64` in CFLAGS to achieve 64-bit compilation. If you do not have a 64-bit machine, then the model grid resolution will need to be reduced so that the entire model and data set can fit within the available memory (using a 32-bit machine in this way is not recommended).

Then, you should simply need to run

```
./configure --prefix=⟨installation directory⟩
make
make install
```

`make distclean` should return your installation to its initial clean state.

The autotools configuration was done by Joerg Dietrich.

2 Introduction to *lens*fit

2.1 What *lens*fit does

2.1.1 Principles

*lens*fit is a bayesian model-fitting code aimed at measuring the shapes of faint galaxies in optical galaxy surveys. Its principles were first developed by Lance Miller and Catherine Heymans in 2001-3. Current weak lensing surveys such as CFHTLenS comprise about 10^7 galaxies, so one design aim of the algorithms is that they should be fast. However, as survey sizes increase, the requirements on acceptable levels of non-cosmological systematics signals become more stringent, meaning that the new generation of measurement algorithms must be simultaneously faster and more accurate than previous methods. The principles that *lens*fit adopts for measuring galaxy ellipticity are described by Miller et al. [2007] and an early implementation and test of the code is described by Kitching et al. [2008].

Key points to emphasise about the approach of bayesian model-fitting are

1. It enables a rigorous statistical framework to be established for the measurement, and thus avoids the need for *ad hoc* "corrections" that other methods require, as well as producing correctly-weighted use of the available data.
2. Quantities other than ellipticity that form part of the problem may be marginalised over, and provided an accurate prior is available those uninteresting quantities may be eliminated from the problem without bias. The prime example is that of the unknown galaxy position: all other methods published to date require a fixed centroid position for a galaxy, and centroid errors introduce spurious ellipticity, with

the possibility of bias. In model-fitting we can treat the unknown galaxy position as a free parameter and marginalise.

3. The weak lensing signal is carried by the faintest galaxies, often the integrated signal-to-noise may be as low as 10. Allowing a model-fitting approach adds extra information to the problem, such as prior knowledge of possible surface brightness distributions. In principle this leads to more precise measurements than a method that does not use such information, thereby allowing measurements to lower signal-to-noise ratio.

4. The models must be unbiased, however, in the sense that they should span the range of surface brightness profiles of real galaxies, and the priors adopted should be an accurate reflection of the true distribution. If these conditions are not satisfied, the model-fitting may be systematically biased. However, such considerations apply also to other methods, even when specific models are not assumed, because "PSF-correction" methods require the assumptions of a finite set of basis functions to be fitted to the data, or else a finite set of moments to be measured. Truncation of those sets to avoid excessive noise results in bias [e.g. Bernstein, 2010].

5. The approach automatically makes an optimum combination of data of varying quality.

The models fitted comprise two-component bulge plus disk galaxies. There are seven free parameters: galaxy ellipticity, e_1, e_2; galaxy position x, y; galaxy size r; galaxy flux; and bulge fraction. In the current implementation, the bulge fraction is analytically marginalised over as part of the likelihood calculation. Galaxy flux and size are numerically marginalised over by sampling multiple values, fitting a function and integrating. Galaxy position can be numerically marginalised over very rapidly using Fourier methods, as described by Miller et al. [2007], because the process of model-fitting is equivalent to cross-correlation of the model with the data, and marginalisation over unknown position becomes equivalent to integrating under the peak of the cross-correlation function. *lens*fit thus calculates the likelihood surface as a function of the apparent galaxy two-component ellipticity. Finally, that likelihood surface may be converted to weighted estimates of ellipticity that are suitably unbiased for shear measurement.

2.1.2 Priors

The marginalisation processes require priors. *lens*fit incorporates priors on galaxy position, scale-length, flux, bulge fraction and intrinsic ellipticity. These priors are assumed to be independent. The scale-length prior depends on galaxy magnitude, but the ellipticity prior is assumed magnitude-invariant (this assumption could be modified if warranted). The disk and bulge scalelength priors are derived from the fits to HST observations of galaxies in the Groth strip by Simard et al. [2002]. Those measurements do not strongly constrain the functional form of the scalelength prior: we adopt a prior of the form

$$p(r) \propto r \exp(-(r/a)^{\alpha})$$

where the value of a is adjusted to match the magnitude-dependent median of the Simard et al fits, and α has a value 4/3 (other values may be set in the code). The scalelength prior is coded in function `rfunc` in the `lensfit` program.

Those faint galaxy fits do not strongly constrain the distributions of galaxy ellipticity, so we make the assumption that faint galaxies have the same distribution of disk ellipticity as low-redshift galaxies of the same luminosity in SDSS: disk ellipticities are determined largely by the disk's inclination, thickness and optical depth at the waveband of observation, the latter being luminosity dependent [Unterborn and Ryden, 2008].

Accordingly we fit the ellipticity distribution of SDSS galaxies selected to have g-band absolute magnitude $M_g \simeq -19$, to correspond in luminosity and waveband to galaxies selected at $i \simeq 24$ in CFHTLenS, for which the median redshift is estimated as $z \simeq 0.68$. A prior for the ellipticity of bulge-dominated galaxies is not readily obtained: in the absence of better information, we adopt a fit to the ellipticity distribution of bulge-dominated galaxies in Simard et al. [2002]. These priors are coded in the functions `efunc` and `bulgefunc` that appear in the `lensfit` and `readlensfit` programs.

Likewise, the bulge fraction is not well-constrained by current data, and even in the nearby universe the bulge fractions deduced for bright galaxies are dependent on the models used to fit them [Graham and Worley, 2008]. A reasonable distribution to assume, and which is also consistent with fits to faint galaxy samples such as Schade et al. [1996], is that there are two populations. Bulge-dominated galaxies are fit by purely de Vaucouleurs profiles, and account for around 10 percent of the galaxy population (set by variable `bulgepopfrac` in `lensfit`). Disk-dominated galaxies are assumed to make up the remainder, and we assume the bulge fraction $f = B/T$ to have a truncated normal distribution in f with its maximum at $f = 0$ (`lensfit` variable `bfpeakval`) and with $\sigma = 0.1$ (`lensfit` variable `bfsigma`), truncated so that $0 \leq f \leq 1$. The bulge fraction is marginalised-over inside the likelihood function (`lensfit` function `likelihood`): this can be done by obtaining erf solutions to the marginalisation which are evaluated numerically using the GSL library. The bulge fraction prior is independent of other parameters.

The total galaxy flux is also marginalised, and the prior adopted is a power-law in flux, with index 2 as appropriate for faint galaxy surveys (`lensfit` variable `fluxpriorslope`).

2.1.3 Galaxy models

The two galaxy model components are de Vaucouleurs profile "bulge" and exponential "disk". Both components are cut off at 4.5 scalelengths (both because this matches reasonably well observations, also for computational expediency). In the standard code version these are calculated as 2D distributions with some ellipticity, and for computational expediency both components have the same ellipticity. Although real galaxies have ellipticity gradients caused by different intrinsic 3D ellipticities of these components, handling the distortion under shear of such a model becomes problematic. *lens*fit also has an experimental module for calculating 3D galaxy profiles by integrating through a 3D model galaxy at a variety of inclinations to the line of sight. *lens*fit can instead optionally apply empirical disk extinction corrections to 2D models, derived by Graham and Worley [2008], although in practice we find that such corrections are degenerate with other model parameters. The Sersic index of the two components may be varied if desired, but those indices cannot themselves be fit parameters, as additional dimensions would then be added to the search space.

The two-component fitting can be done computationally cheaply provided we do not introduce any further free parameters in ellipticity or size - hence we fix the bulge half-light radius to equal the disk exponential scale-length (as typically found for brighter galaxies) and we also assume the bulge and disk ellipticities are equal (we might expect disk ellipticity to be larger than bulge ellipticity on average for inclined galaxies). The two components are co-centred. These assumptions could be varied but at present we have little grounds for doing so. The key advantage of this decomposition is that it allows a greater range of surface brightness profile, which is particularly important for cuspy galaxies when the PSF is asymmetric, and it does result in a significant reduction in residuals in the CFHTLenS analysis. Naturally the deduced bulge fractions for faint galaxies are extremely noisy and shouldn't be over-interpreted!

2.1.4 Combining multiple exposures and distortion-correction

All modern surveys comprise multiple dithered exposures. Early in the development of `lensfit` it was realised that the model-fitting should be carried out jointly on these multiple exposures, rather than the exposures being co-added and shear measurement being made on those co-added images. There are several compelling reasons for this.

1. If each exposure has a different PSF, the optimum way to measure shape is to preserve that information - a likelihood-based approach allows exposures with broad PSFs to be correctly weighted with respect to exposures with "better" PSFs, information that is lost if exposures are simply co-added.
2. The process of co-adding requires data to be interpolated onto a new pixel grid. The interpolation kernel necessarily varies with position: this is equivalent to a position-dependent convolution of the PSF. The variation tends to be cyclical (as the native and co-added grids go in and out of phase with respect to each other) and typically on a scale smaller than can be measured using stars. In principle this effect can be reconstructed given knowledge of the interpolation, but in practice this would cause significant complications.
3. The position-dependent convolution results in non-stationary correlated noise. If the noise can be considered to be quasi-stationary everywhere near a faint galaxy, the noise correlation can be included in the likelihood calculation, but varies between galaxies: again such a requirement is avoided if the data are never interpolated.

In `lensfit` the multiple exposures are jointly fitted by each galaxy model. Accurate sub-pixel image registration is crucial: any image registration mismatch results in a systematic error in ellipticity. The problem is worse in this case than if exposures are co-added, because in that case any image registration errors affect both the galaxies and the measured PSF, and hence at some level are corrected for (although again there may be an unhealthy spatial variation that cannot be measured). In `lensfit` the problem is dealt with by cross-correlating the stars used to make the PSF with the PSF model for each exposure, and finding an accurate position offset. A smooth polynomial is fitted to those offsets across each mosaic-camera sub-image, and when the galaxy models are jointly fitted, the models are offset by those amounts appropriately for each exposure. Thus, when a unique WCS position is specified for each galaxy, even if that position has some uncertainty, the *relative* registration between exposures is extremely accurate. Furthermore, because the image registration has been determined using the actual PSF model, that process is as accurate as it can be, and should be free of concerns about bias in image registration arising in the case of asymmetric PSFs.

In the joint fitting, the actual galaxy position is assumed unknown, and can be marginalised over by integrating under the cross-correlation function of the galaxy model and the data (see Miller et al. [2007]). In the current version of `lensfit` we adopt a uniform prior to avoid bias (i.e. we assume we have no prior information on the galaxy's position). As pointed out by Miller et al. [2007], it is not possible for such a prior to be unbounded, as we do not obtain a finite prior or posterior probability in that case. However, this problem arises because we have not asked quite the right question. A uniform prior extending to infinity would allow some galaxy a long way from the nominal galaxy of interest to be included in the likelihood estimation. Such a galaxy may exist, but it is not the galaxy that we are trying to measure. So we should restrict the prior to just the location of one specific galaxy. We probably initially detected the presence of the galaxy of interest from the same data that we are now considering, by running a convolving filter over the image and looking for a significant maximum in the cross-correlation. This is precisely what we do in the weak lensing measurement: we cross-correlate the data with a PSF-convolved model. So the limits of the prior should be the radius outside which the cross-correlation amplitude drops below some detection

threshold: any location outside that radius corresponds to the possible location of another random galaxy and not the one we have detected. Inclusion of this hard prior (into `lensfit` function `likelfunc`) may be done analytically, resulting in solutions involving the exponential integral, which may be evaluated using GSL routines.

The precise registration of the sub-images requires more than a simply rectilinear offset between frames. In general each image is rotated and distorted with respect to any simple projection of the sky. For bias-free measurement, we should ensure that this information is included in the joint fitting process. The way it is done in `lensfit` is, for each galaxy being measured, evaluate a linearised affine distortion matrix that effectively allows shear, rotation and scale variations between images, and then to apply the distortion matrix to the model for each individual image (see section 2.4.2).

2.1.5 PSFs

Ultimately this (and other) shear measurement methods will be limited by uncertain knowledge of the PSF. PSF errors produce systematic effect on the shear measurement as they correlate between galaxies. It would be desirable to marginalise over uncertainties in the PSF, but at the time of writing this is not possible. Instead, we determine the PSF to the best of our ability on each exposure and assume that the PSF determinations are exact.

In *lens*fit the PSFs are determined as postage stamps of pixel values, normalised to unity. Thus we do not risk biasing the PSF by making some choice of basis function truncated to fewer than the number of pixels. As noted above, the PSF should be determined on each exposure individually. Furthermore, on the Megacam mosaic camera, there are discontinuities between mosaic sub-images ("chips"). In *lens*fit we deal with this by fitting 2D polynomial functions to the PSF pixel values, where the polynomials span the entire mosaic, but where low order coefficients are allowed to vary between chips (effectively allowing low-order offsets in pixel values between chips). Fig. 1 shows the stacked mean PSF obtained by this method in one CFHTLenS field. The stacking has been done purely as an illustration here to show the discontinuously-varying PSF at overlapping chip boundaries, when using dithered exposures. Any method that relied on measurement from stacked images would need to be able to deal with these overlap regions.

2.1.6 Shear estimation

The measurement of each galaxy returns either the likelihood or posterior values as a function of observed galaxy ellipticity. To measure cosmological shear, we must calculate some n-point statistic averaged over many galaxies. For the sake of simplicity in the following, let us assume that the statistic we wish to use is the mean shear in some region: the considerations below may be generalised to more useful higher-order statistics.

To form the probability distribution of the average of a set of probability distributions (here, the likelihood surfaces), we should convolve those probability distributions (with some appropriate weighting). The central limit theorem tells us that the mean shear may be evaluated as the mean of the individual shear estimates, and that the probability distribution tends towards a normal distribution with variance a factor n smaller than the variance of an individual measurement, for combination of n galaxies. Thus we expect that we do not need to preserve the full likelihood surfaces and convolve them, we can instead form a shear estimate for each galaxy and then combine those estimates.

However, Miller et al. [2007] argued that a bayesian estimate of galaxy ellipticity is a biased estimator of

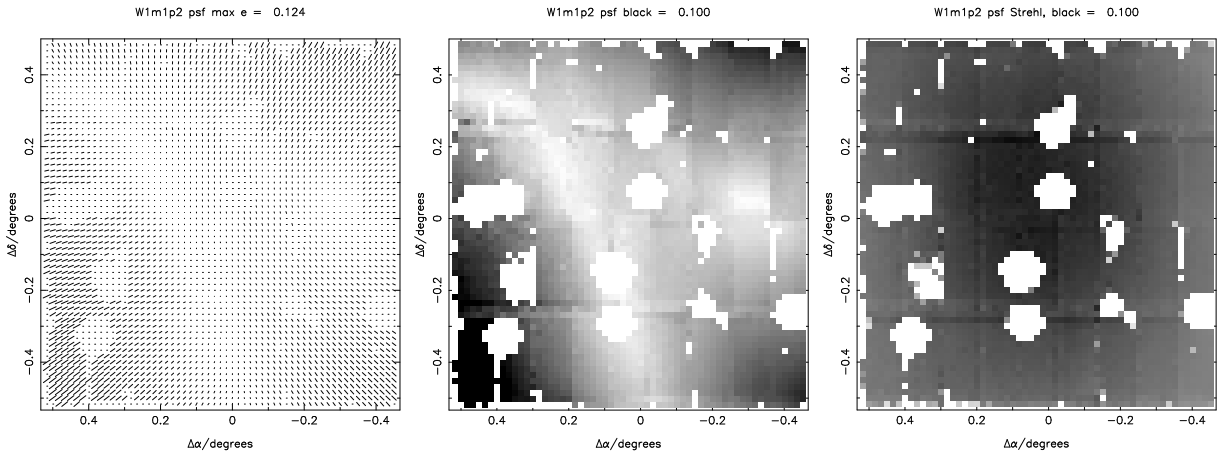

Figure 1: The variation of measured PSF over a 1 deg Megacam field, from the CFHTLenS survey. The plot has been made by combining information from seven exposures, which are somewhat offset with respect to each other. PSF ellipticities have been simply averaged. (left) stick plot showing the variation of measured PSF ellipticity across the field. Holes indicate regions where bright star have been masked out. The maximum stick length on the plot corresponds to ellipticity of 0.124. (centre) greyscale showing the ellipticity amplitude, with black having an ellipticity of 0.1. Note the discontinuous regions where there are dithered chip boundaries. (right) greyscale showing the fraction of the light in the central pixel of the PSF, with black indicating a fraction 0.1.

shear when applied to an ensemble. One explanation of this is to note that, if we multiply the likelihood surface of each galaxy by some prior, find the mean posterior ellipticity for each galaxy and then average, we can make both the mean and its variance arbitrarily small for large samples, even in the case where the likelihood surface is flat (no information). The fault lies in the multiple application of the prior - we have imposed our prior information too enthusiastically, and what we should have done is to form estimates of the mean likelihood ellipticity for each galaxy, average, and only then apply the prior. In the case where the likelihood surfaces are flat (or at least, very broad), the final distribution of the mean after application of the prior simply looks like the prior as expected, no matter how large the sample of galaxies.

The reason the posterior estimate differs from the likelihood estimate for an individual galaxy is because the measurements are noisy. At high signal-to-noise the bias is small, at low signal-to-noise the bias is high, and may be evaluated following Miller et al. [2007]. The (linearised) bias correction factor was referred to as the "sensitivity" by Miller et al. and is a reflection of the information content of each likelihood surface. The *lens*fit code calculates these sensitivity estimates, although those sensitivity values have now been superceded by an improved method.

Application of the sensitivity values to the posterior mean values is problematic for a number of reasons. First, in principle the sensitivity is a rank 2 tensor. Even if we only consider its diagonal elements, as advocated by Miller et al. [2007], it has orientation dependence. Because the likelihood surface in ellipticity is correlated with the PSF orientation, use of the full orientation-dependent sensitivity would introduce correlation between the resulting galaxy weights and the PSF. A possible solution is to take the geometric mean of the two diagonal components (or perhaps the determinant) as a weight: this is somewhat *ad hoc*. A second practical limitation is that to form an estimator of, say, the mean shear we need to evaluate $\sum_i < e_i > / \sum s_i$ where $< e_i >$ is the estimated expectation value of ellipticity for the i^{th} galaxy and s_i is its

orientation-averaged sensitivity. This does not lend itself well to standard algorithms that take a weighted mean.

An improved method that leads to demonstrably lower PSF-galaxy cross-correlation systematics on CFHTLenS data is not to use the sensitivity, but instead to revert to a likelihood estimator of the galaxy's ellipticity, which should then be unbiased but noisy. The noise is then taken into account by calculating an inverse-variance weight, where the variance is given by the convolution of the prior 2D ellipticity distribution with the likelihood distribution. This is the procedure now adopted in `readlensfit`

2.2 Overview of *lens*fit operation

As explained above, *lens*fit operates on images that have already had bias and dark current removed and have been flat-fielded and fringe-corrected.

If using the mode where likelihoods from multiple images are to be combined, it is currently assumed that pixel values have been calibrated onto a single scale such that a galaxy has the same measured flux on each image (as produced by Elixir on CFHT Megacam images, for example). A calibration scalar may be present in the FITS headers. The keyword name is specified in the scamp/swarp configuration file. The signal from each image is weighted inside `lensfit` by the inverse noise variance.

Basic measurement then proceeds in four phases: first, a position-dependent PSF is created by program `makeglobalpsf` ; second, residual astrometric shifts are measured using `globalshifts` (only necessary if there are multiple exposures); third, the galaxy shapes are measured by program `lensfit` which also reads the output from `makeglobalpsf` ; finally any shear analysis is done by analysing the output catalogue of individual galaxy shapes with `readlensfit` .

*lens*fit requires two input catalogues to have been generated with software such as SExtractor. The first should be a list of stars on the image, to be used for constructing the PSF, the second should be the galaxies whose shapes are to be measured. In order to probe as faint as possible, the stars and galaxies should be selected from the deepest available data, which usually will be a coadded stack of all images. `lensfit` will not use that coadded stack, only the catalogues of stars and galaxies created from it. The PSF stars should be distributed over the entire image, and should be reliably classified as single stars. The PSF routine allows the input stars catalogue to be filtered by star brightness and by integrated signal-to-noise, and also tries to eliminate stars that have close neighbours, but the more reliable the input catalogue the better the results. The PSF routine iteratively removes stars whose profile does not match well the local PSF that is being created, with the quality of the match being measured by cross-correlation. The method seems quite effective at removing galaxies and blends, so it should not be necessary to select individual stars by eye. In CFHTLenS we also applied a prior rejection of stars whose colours did not lie on the normal colour-colour locus. Some tuning of the star selection parameters may be required nonetheless. We particularly strongly recommend setting a minimum threshold of 20 on the signal-to-noise ratio of input stars: this may be achieved using one of the `makeglobalpsf` command-line inputs(see below).

`makeglobalpsf` and `lensfit` operate on "postage stamps". These are sub-images extracted from the main image centred on the stars and galaxies that are analysed. Each galaxy is measured one at a time. The postage stamp size must be the same for all objects (at present, it would be a substantial modification to the code to change this). Its value is set at 32 pixels square inside `makeglobalpsf` (see Appendix for how to change this). `lensfit` reads the output from the PSF coefficients files and sets its postage-stamp size to be the same, although this can be overridden (see Appendix). In fact in the present version of `lensfit` , the postage stamp size is automatically reset to a minimum value of 48 pixels square if what is supplied

in the PSF coefficients files is less than that, and the PSF postage stamps are padded with zero outside the measured region. The galaxy postage stamps are not saved as output, although a small sample may be written out if specified, for testing.

`lensfit` assumes that multiple images will be analysed, it may easily be run on a single image as a special case of this (see section 2.3).

The output from `lensfit` is a FITS table giving the posterior probability surface in ellipticity for each galaxy, marginalised over the other fit parameters, as measured on the combined images in the multiple-image case (i.e. there is always a single ellipticity distribution created no matter how many images are used for each galaxy). Conversion of these surfaces to ellipticity values is done by program `readlensfit`

2.3 Mosaics and exposures

*lens*fit assumes that each exposure it is given to analyse may contain a number of *non-overlapping* CCD images, for example the Megacam camera has 36 such CCDs. Each individual image should be provided to *lens*fit as a separate FITS file (i.e. do not give a single mosaiced file to *lens*fit , although you might already have made one for the purposes of creating star and galaxy catalogues). The reason is that the PSF can then be modelled for each CCD image with discontinuities between the CCDs being accounted for. The negative side of this is that the code needs to know the arrangement of the mosaic CCD images in order to construct a global PSF, and it needs to be told how many CCDs there are (e.g. 36), what their arrangement is (e.g. 4×9 with some gaps between them) and how to identify them from information in the FITS headers or filenames. At present this is the most unsatisfactory aspect of *lens*fit as it requires some changes in the code, made by hand, to set it up for a new camera. If you only have a single-CCD camera this is a trivial special case of the multi-CCD camera. The non-overlapping assumption is important however: if your CCD images overlap then they must be treated as separate exposures as in the next paragraph. Some standard camera setups are already selectable using the SUPRIME/CFHT/SINGLE switches, but even for these cameras a particular filename format is assumed, that you may need to either conform to or edit in the code.

You may also have repeat exposures of the same piece of sky, perhaps only partially-overlapping with earlier exposures. You should first create PSF models for each exposure separately. These multiple exposures can then be given to `lensfit` which will then carry out an optimal coaddition of all data. The only information you need to provide is the *maximum* number of times any one galaxy has been observed. `lensfit` will then allocate enough memory to deal with the data, and keep track of which exposures are useful for each galaxy. The camera geometry is assumed to be the same for all exposures, however individual CCD images are allowed to be missing from some exposures. Again, the case of a single exposure is a trivial special case.

2.4 Preparation of input images

2.4.1 Basic preparation

Before running either the PSF creation code or `lensfit` itself, the images must be prepared by having a spatially-varying background level subtracted from them, so that the background value of the data becomes zero with no background gradients. In the CFHTLS analysis we use the swarp background subtraction, which fits splines to median-filtered samples across each image. `lensfit` does not do this step.

Second, it is important to identify saturated stars or rows with saturated readout trails. To do this the images must have a keyword SATLEV in the header which defines the saturation level (after all calibration

steps have been carried out) and which is correctly set. The swarp configuration file will be searched for the name of the saturation level keyword (e.g. SATLEV).

Fringing patterns could cause contamination of the lensing signal and these should also be removed.

The provision of additional mask images identifying bad regions of an image are optional but are strongly recommended, and should be used to flag out hot pixels, bad columns, ghosts from bright stars or other regions of invalid data.

2.4.2 Preparation for correction of image distortion

Images are usually distorted (sheared) by a combination of atmospheric refraction and optical distortion in the camera. The latter should generally be stable with time in long exposures, but may be a function of temperature or camera orientation. The former is primarily a function of wavelength, observatory altitude, and observation zenith distance. A robust way of empirically correcting for these effects is astrometrically, measuring the positions of stars on the images to be corrected, comparing with the positions in accurate surveys, and fitting distortion correction functions. It is of course essential that the input catalogue astrometry is accurate. The fitting may be done by the program scamp[2] which produces additional ascii header files that in turn may be read by the companion program, swarp, which interpolates the original image onto a new set of pixels that are free from the original distortion. However, the swarp process is undesirable for weak lensing measurement, for two reasons. First, the interpolation introduces covariance in the noise between pixels. Second, the interpolation kernel acts as a convolution, and the convolution kernel varies systematically over the image on arcminute scales. Thus, in principle, the PSF is modified as a function of position, however the angular scales are sufficiently small that it is difficult to recover the effect when determining the PSF, and yet the scales affected are where we wish to measure weak lensing signal.

The solution adopted here is to use the scamp distortion correction, but for `lensfit` to use the correction information internally in a way that avoids the above two problems. It does this by extracting postage stamps for each galaxy centred on the nearest pixel to the nominal galaxy position, and applying the swarp correction to galaxy models being fitted.

Hence, to correct for distortion, the astrometric correction functions and scamp header files need to have been produced before running `lensfit` . Because we don't want to interpolate the stars CORRECT_DISTORTION should be set to 0 when making the PSF. But in `lensfit` it should be set to a value 2. You must ensure consistency between these two pieces of software, although some checks are made and processing is halted if an inconsistency is detected. A number of pieces of input information are also needed from scamp and so a scamp/swarp configuration file is also expected (this could be tidied up in future versions).

The code will also accept images assumed to have no distortion, for which the scamp headers are therefore not required, but for real data, astrometric correction is recommended.

2.5 Improvements to *lens*fit

This guide aims to explain what *lens*fit does and how to use it. It is not a code developers' guide. However, it is useful to point out possible areas for improvement, some of which are currently being developed, others that are not. Some of the possible improvements fall into the category of improving the generality of the code to differing datasets, others are more concerned with the core algorithms.

[2] http://www.astromatic.net/software/scamp

2.5.1 Generalising to arbitrary datasets

It would be convenient not to have to reconfigure `lensfit` for new camera geometries. Joerg Dietrich has been working on this.

2.5.2 PSF modelling

The main limitations of the current modelling are: it assumes the PSF is fully sampled; it fits piecewise polynomials which may not be the best choice of interpolating function; the order of those fitted polynomials has to be specified by hand rather than by a rigorous statistical evaluation; it does not calculate uncertainties on PSF pixel values and does not marginalise over those when fitting. There is some evidence in the CFHTLenS analysis that the sampling assumption is inadequate and this is currently the highest priority area to improve. In the long term an entirely new PSF algorithm would be warranted to solve these issues.

2.5.3 Model improvement

Galaxy models are necessarily simple at present. Experimental 3D models are available but have not yet resulted in significant performance improvement.

2.5.4 Speed-up

The code would benefit from speed-up, and more complex models and sampling could be adopted if it were faster. Kris Zarb-Adami is experimenting with porting the code to a GPU system.

3 Running the *lens*fit programs

3.1 Environment variables

Environment variables that identify the location of the various input and output files should be defined.

DATA_DIR defines the path to the input FITS images (if not set, defaults to current directory).

HEAD_DIR defines the path to the additional ascii .head files for the input FITS images in the case where CORRECT_DISTORTION is specified (not needed otherwise). If not set, defaults to DATA_DIR.

SWARP_CONFIG defines the path and name of the scamp/swarp configuration file (must be set if CORRECT_DISTORTION is specified).

CATALOGUE_STARS defines the full path and name of the input star catalogue read by `makeglobalpsf` and `globalshifts` (must be set).

CATALOGUE_GALAXIES defines the full path and name of the input galaxies catalogue read by `lensfit` (must be set).

PSF_DIR defines the path where the PSF FITS files are written and where `lensfit` will look to read these files (if not set, defaults to current directory).

BADPIX_DIR defines the path to any pre-existing "weight" masks (if not set, defaults to current directory).

FITS table output from `lensfit` is written into the current directory.

3.2 `makeglobalpsf`

`makeglobalpsf` creates a single coefficients outputs file for all the images in a multi-CCD mosaic exposure, provided each image is input as a separate FITS file. The first stage of the program is the same as the older code `makepsfsurface` but once all the star postage stamps have been extracted, stars centred and bad stars rejected, the fitting then proceeds as a global polynomial fit to the entire dataset, with an in-built assumption about the geometrical arrangement of the input images. A potentially important sophistication is that some or all of the polynomial coefficients may be set independently for each chip (this is possible because each chip image is read in separately rather than being combined into a single giant FITS image). Thus, for example, you could fit a global third-order polynomial spanning the entire mosaic exposure, but specify that the 0th-order coefficient be allowed to vary between chips (or any other combination of orders. If the chip-dependent order is higher than the global order the global order is chosen for the chip-dependent solution and higher order terms are not calculated).

Inputs to `makeglobalpsf` are provided as command-line arguments:

`makeglobalpsf` ⟨image list⟩ ⟨global order⟩ ⟨chip-dependent order⟩ ⟨snratio⟩ [⟨bright mag⟩ ⟨faint mag⟩]

image list is an ascii list of the input FITS images to be measured. There should be one FITS image for each chip in the mosaic. The FITS images given in the list must all exist, they should also have associated weight images with a filename convention that is fixed inside `makeglobalpsf` and if necessary the scamp/swarp .head files must also exist with a filename convention that is fixed inside `makeglobalpsf`

global order specifies the polynomial order used for the global fit across all chips. Order 0 is a constant term, order 1 is linear etc. Negative values of order allow higher-order 2D cross-terms to be included (i.e. -2 is a quadratic fit that includes an x^2y term and similar: positive order does not include such terms).

chip-dependent order specifies the order of terms that are to be allowed to vary between chips: e.g. chip-order 0 means that a constant term is allowed to be different for each chip. Negative order has the same meaning as above. If no chip-dependent terms are required a value of "none" should be entered.

S/N ratio is a lower limit required for the integrated signal-to-noise ratio of each star to be used for the PSF. Low S/N stars are downweighted, but nonetheless allowing low signal-to-noise stars in the selection both significantly increases the computation time needed by `makeglobalpsf` and also risks creating a biased PSF. The centroids of noisy stars cannot be accurately determined and may be dominated by individual noisy pixels, so it is a bad idea to allow low S/N stars in the selection. We recommend a minimum peak S/N of at least 20 be specified (or else stars in the input list be pre-selected with a similar criterion), although this criterion should be tuned for a particular survey to compromise between using stars with high S/N and having enough stars to create a good PSF in each sub-field.

brightmag and *faintmag* are optional bright and faint limits on the magnitude of the PSF stars, only stars with brightmag < magnitude < faintmag are used to make the PSF. If used, both limits must be specified. If these values are not specified they default to a wide range of magnitude. They are largely redundant if a S/N threshold is specified. Stars brighter than a nominal saturation limit are also rejected inside `makeglobalpsf` (see below).

In addition, `makeglobalpsf` tries to read some keywords from the image FITS header. The most important of these is the saturation level keyword (assumed to be named either SATLEV or SATURATE or specified in the swarp configuration file). This is needed to reject saturated stars from the PSF creation. If the keyword is missing the value will default to the standard value for 16-bit AD conversion, 65535. Stars are rejected if their peak pixel values exceeds 50 percent of the saturation level (on many CCDs the response becomes non-linear as the saturation level is approached).

Filenames are evaluated as follows inside makeglobalpsf. The input images have a suffix ".fits" added to the input name and the value of DATA_DIR added as a prefix. Weight image names are created by stripping off the last part of the input image name and adding a suffix ".weight.fits" with prefix given by the value of PSF_DIR. The scamp/swarp head files have suffix given as specified in the scamp/swarp configuration file (usually ".head") and this is also added on to each input image name with the last part stripped off and the value of HEAD_DIR, or DATA_DIR if HEAD_DIR is not set, added as a prefix. Which parts of the input string are stripped off is specified inside the code by the variable named `delimiter` which is input to the strtok function: you can change the delimiter to match your filename structure.

To be able to make a global fit the code has to make an assumption about the geometrical arrangement of the chip images. It does this by deconstructing the input image filename. It assumes the first six characters are an exposure name or number that is common to all the chip images for that exposure. Then a numerical chip identification number follows, and this is used to place the chip in its correct relative location. `lensfit` also attempts to read a "chip number" keyword, IMAGEID, from the FITS header, hopefully these agree. At present there are three cameras allowed in the code: CFHT Megacam, SUPRIMEcam or an assumed SINGLE CCD camera. For CFHT the code assumes there are 36 chips (specified by static variable NCHIPS) and that these are arranged as 9 chips on each of 4 rows. The horizontal and vertical spacing between chips, including gaps, are specified with the program variables `xchipspacing, ychipspacing`. For Megacam an additional gap between the 2nd and 3rd rows is specified with the program variable `big gap`. Hence to use this code on other datasets some modifications both to the filename structures and the assumed geometry are required inside the `makeglobalpsf` and `lensfit` codes. Ask LM for advice.

Note that, in the case of multiple exposures, you should specify the same global and chip-dependent order for each exposure. Otherwise `lensfit` will take the fit order of the first exposure it is given and assume the other exposures have the same fit order.

3.3 `globalshifts`

`globalshifts` reads the PSF coefficients files created by `makeglobalpsf` and writes the fit coefficients to the residual astrometric shifts into the same coefficients files. The code itself is largely cloned from `makeglobalpsf` . This could all be done faster if the two codes were combined but this has not yet been done.

This step is only necessary if there are multiple exposures, but in that case it is essential.

Command line inputs are the same as `makeglobalpsf` except that the two fit-order inputs are absent (the `globalshifts` astrometric fit order is fixed at 2D linear). The catalogue inputs and signal-to-noise inputs should be the same as specified for `makeglobalpsf` , with the same input list of input images.

`globalshifts` ⟨image list⟩ ⟨snratio⟩ [⟨bright mag⟩ ⟨faint mag⟩]

3.4 `lensfit`

`lensfit` operates on all the available input images simultaneously in order to optimally combine measurements. Inputs to `lensfit` are provided as command-line arguments:

`lensfit` ⟨image list name⟩ ⟨output FITS table name⟩ ⟨exposure 1⟩ ⟨exposure n ⟩ [⟨bright magnitude limit⟩ ⟨faint magnitude limit⟩]

image list name Input ascii list containing the names of all the input FITS images required to be analysed. The names should not have the .fits extension specified (see example image list). `lensfit` will then construct an image name ⟨name⟩.fits and a PSF name for each input image of the form ⟨name⟩_psfcoeffs.fits and look for this file in directory PSF_DIR. It will exit if this PSF file is not found. `lensfit` will also look for an image pixel weight file of the form ⟨name⟩.weight.fits for each individual image specified in the input list, but in the path defined by BADPIX DIR. If this file does not exist `lensfit` either issues a warning but continues, or halts, depending on the value of WEIGHTS_REQUIRED set inside the `lensfit` source code. For swarp images it is essential that these weight files is available. The weight file must be a FITS image of the same dimensions as each input image, with pixel values of 0 for "bad" pixels and values > 0 for "good" pixels. The input ASCII catalogue of galaxies to be measured is specified by the environment variable CATALOGUE_GALAXIES, which must be set. This must specify the positions of each galaxy on each of the input images on which it appears, see Appendix for further details.

output FITS table name Outputname of your choice. This FITS table can be read by program `readlensfit`

exposure 1, exposure n specifies which image exposures in the catalogue you wish to be analysed (see Appendix for more details).

bright & faint magnitude limit This optional pair of parameters allows faster processing time by skipping over input galaxies that are outside the specified magnitude limits, where the limits are compared with magnitude values given in the input catalogue. Skipped galaxies are preserved in the output file but are given flags indicative of "no data".

The input catalogue of galaxies is specified by the environment variable CATALOGUE_GALAXIES and whether WCS or pixel coordinates are specified must match the choice of setting of the global variable WCS inside the `lensfit` code.

`lensfit` writes a FITS file to the output filename in the current directory. The file contains two binary FITS tables. The first lists the ellipticity and galaxy scale-length values that have been sampled. The keyword DELTA-E also gives the highest sampling in ellipticity (currently set to $\Delta E = 0.02$ within `lensfit`). The second table gives the posterior probability surface for each galaxy provided in the input list, with the following information:

column	quantity
1	catalogue ID number from the input catalogue
2	world R.A. coordinate from the input catalogue
3	world dec coordinate from the input catalogue
4	catalogue magnitude from the input catalogue
5	vector of names of input images used for this galaxy
6	vector of ID numbers of chips used for this galaxy
7	vector of values of PSF e1 ellipticity component for each individual image at this galaxy
8	vector of values of PSF e2 ellipticity component for each individual image at this galaxy
9	mean PSF ellipticity at this galaxy (two-component quantity)
10	mean PSF "Strehl ratio" at this galaxy (defined as fraction of PSF light contained in the central pixel)
11	fit class
12	fitted galaxy scale-length (marginalised over ellipticity)
13	fitted bulge fraction (marginalised)
14	fitted total flux
15	data signal-to-noise ratio integrated over isophotes of galaxy
16	fit probability of best-fit model
17	sampling factor of the ellipticity grid (1=highest)
18	sensitivity (two-component quantity)
19	list of ellipticity values reported (i.e. the number ID of the ellipticity grid point for the ellipticity grid given in the first output table)
20	output log(likelihood) values at each of the specified ellipticity grid points, marginalised over the other fit parameters.

The ellipticity is defined as $(a-b)/(a+b)\exp 2i\theta$ with PA=0 in the EW direction, where a, b, θ are the major, minor axes and position angle respectively. Note that, when calculating the shear correlation function between points that are widely separated on the sky, the rotation of the position-angle coordinate system around the celestial sphere must be taken into account using spherical trigonometry.

fitclass takes the following values:

-4	fit exceeds reduced-χ^2 limit of 1.4
-3	bad likelihood surface
-2	galaxy rejected by blending criteria
-1	no data found, or galaxy outside magnitude limits
0	galaxy fit found
1	star fit found

sensitivity is a two-component vector for each galaxy, being $\partial\langle e_\mu\rangle/\partial g_\mu$ for $\mu = 1, 2$, as defined by Miller et al. [2007]. Use of this quantity has now been superceded by the improved shear estimation in `readlensfit` .

log likelihood values is a vector with number of elements given by the number of sampled ellipticity values (as in the first table). The values are $\log_e$ probability. The surface is normalised per unit ellipticity, so that $\sum_i p_i \Delta E^2 = 1$. Posterior probability values, rather than likelihood, may be recovered by adding the log(prior) values to the log(likelihood) values, although renormalisation will then be required. Note that older versions of `lensfit` created a table of posterior values: `readlensfit` will read correctly either old or new format files according to the setting of a version keyword in the header.

`lensfit` also optionally writes out FITS files of summed posterior probability distributions in magnitude slices (files containing names *magbin*) which could be used to help iteratively improve the prior [Kitching et al., 2008].

3.5 `readlensfit`

Program `readlensfit` reads the `lensfit` output table and carries out estimation of the shear likelihood distribution for each galaxy, writing output to an ascii file which contains for each galaxy the WCS coordinates, the $\langle g_1 \rangle$, $\langle g_2 \rangle$ and inverse-variance weight values, plus additional information out of the FITS table. The command-line syntax is

`readlensfit` ⟨nmin number of exposures⟩ ⟨input fits file⟩ ⟨output ascii file⟩

The code reads the `lensfit` output fits file to get the fit likelihood surface. If the output file already exists, the program exits, it will not overwrite. The output file is a simple ascii list, one row per galaxy, with some initial header rows prefixed by "#".

column	quantity
1	wcsx
2	wcsy
3	⟨g1⟩
4	⟨g2⟩
5	inverse-variance weight
6	fitclass
7	size
8	bulge-fraction
9	model-flux
10	SNratio
11	fit-probability
12	⟨PSF⟩-e1
13	⟨PSF⟩-e2
14	⟨PSF⟩-Strehl-ratio
15	s1
16	s2
17	bayesian-e1
18	bayesian-e2
19	likelihood-emod
20	bayesian-emod
21	catmag
22	n-exposures-used
23	cat-ID

4 Fair use of *lens*fit , acknowledging *lens*fit and feedback

*lens*fit has involved substantial development work by its authors, working in collaboration with the CFHTLenS consortium. Use and development of *lens*fit is permitted on condition that the code remains owned by the primary authors, and that *lens*fit and derivatives based on *lens*fit are not used to compete on the science goals of the authors and, with respect to CFHTLS data, the CFHTLenS team. If in doubt, please discuss with the author.

If your use of *lens*fit results in published work, you should acknowledge its use with the following form of words, "We acknowledge use of lensfit [Miller et al., 2007, Kitching et al., 2008] for shear measurement.".

Thanks! Obviously, we would welcome any constructive feedback.

References

G. M. Bernstein. Shape measurement biases from underfitting and ellipticity gradients. *MNRAS*, 406: 2793–2804, August 2010. doi: 10.1111/j.1365-2966.2010.16883.x.

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

A. W. Graham and C. C. Worley. Inclination- and dust-corrected galaxy parameters: bulge-to-disc ratios and size-luminosity relations. *MNRAS*, 388:1708–1728, August 2008. doi: 10.1111/j.1365-2966.2008.13506.x.

T. D. Kitching, L. Miller, C. E. Heymans, L. van Waerbeke, and A. F. Heavens. Bayesian galaxy shape measurement for weak lensing surveys - II. Application to simulations. *MNRAS*, 390:149–167, October 2008. doi: 10.1111/j.1365-2966.2008.13628.x.

L. Miller, T. D. Kitching, C. Heymans, A. F. Heavens, and L. van Waerbeke. Bayesian galaxy shape measurement for weak lensing surveys - I. Methodology and a fast-fitting algorithm. *MNRAS*, 382:315–324, November 2007. doi: 10.1111/j.1365-2966.2007.12363.x.

D. Schade, S. J. Lilly, O. Le Fevre, F. Hammer, and D. Crampton. Canada-France Redshift Survey. XI. Morphology of High-Redshift Field Galaxies from High-Resolution Ground-based Imaging. *ApJ*, 464:79–+, June 1996. doi: 10.1086/177301.

L. Simard, C. N. A. Willmer, N. P. Vogt, V. L. Sarajedini, A. C. Phillips, B. J. Weiner, D. C. Koo, M. Im, G. D. Illingworth, and S. M. Faber. The DEEP Groth Strip Survey. II. Hubble Space Telescope Structural Parameters of Galaxies in the Groth Strip. *ApJS*, 142:1–33, September 2002. doi: 10.1086/341399.

C. T. Unterborn and B. S. Ryden. Inclination-Dependent Extinction Effects in Disk Galaxies in the Sloan Digital Sky Survey. *ApJ*, 687:976–985, November 2008. doi: 10.1086/591898.

A Setting up *lens*fit

This appendix contains additional detail relevant to setting up the code.

A.1 Background subtraction

It is essential that any varying background is first subtracted from the input images. In CFHTLenS we used the SExtractor filter, which fits polynomials to median-filtered samples across the image.

A.2 Saturation level

The saturation level must be set as a FITS header keyword SATLEV or keyword as specified in the swarp configuration file.

A.3 Global variables

Set the values of global variables inside the `makeglobalpsf` and `lensfit` codes as described above, to control the modes of operation of *lens*fit .

A.4 Postage stamp size

The postage stamp size should be chosen to match the survey according to the pixel scale. The size should be large enough that the galaxies being measured fit inside without having significant isophotes at the edge. For faint surveys such as CFHTLenS, angular sizes of ~ 10 arcsec seem adequate, and we used postage stamps of size 48×48 pixels. For a postage stamp of size $n \times n$ the compute time scales roughly as $n^2 \log n$ and the memory usage scales as n^2, so there is a strong incentive to use the smallest possible size. There is less likelihood of confusion from neighbouring objects with smaller postage stamps, also. It might be worth experimenting on a subset of the data with a range of sizes. Fourier transforms are done with the *fftw* library, which chooses the fastest algorithm appropriate to the specified postage stamp size (the size does not need to be a power of two, see the *fftw* manual for a discussion of sizes).

The size is initially set as a fixed number inside the `makeglobalpsf` code, and to change it you would need to change the line in the code: `pwidth = 32;` to a new value and recompile.

The size is passed in the `makeglobalpsf` output FITS files to `lensfit` . However, you can override the default inside `lensfit` by setting POSTAGE_STAMP_SIZE to whatever value you wish. At the moment `lensfit` will automatically force a minimum postage stamp size of 48 pixels square. In CFHTLenS we evaluated PSFs with sizes 32 square for speed and allowed `lensfit` to pad these to 48.

Constraints: Although pwidth and pheight may be specified separately, the code has only ever been tested on square postage stamps, and any rectangular shape is liable to introduce a bias in the shape measurement, so make sure the two variables are equal. Also, **the size must be an even number**: the Fourier transform arrays will be incorrectly filled otherwise.

A.5 Star catalogues for PSF generation

`makeglobalpsf` works only on individual CCD images, so for the PSF generation we do not need a composite catalogue of stars. It might however be convenient to specify a single star catalogue that covers your whole mosaic survey: that way you don't need to worry about working out which stars cover which image, `makeglobalpsf` will simply use any of the stars that you have specified that lie on each image. The input catalogue should consist of a list of star positions in either WCS or pixel coordinates, plus optionally a magnitude value which may be used to select ranges of star brightnesses. More critical however is the command-line input parameter "S/N ratio", which cuts out stars that have peak S/N less than the specified value. Although any value for this parameter is allowed as input, it is recommended that a minimum value of 20 is adopted, otherwise the resulting PSFs are just too noisy and badly fitted by the polynomials. A "clean" input catalogue of stars (i.e. excluding blends and non-stellar objects) is helpful but `makeglobalpsf` does do its own cleaning of badly-fitting stars. In CFHTLenS we also rejected stars whose colours did not lie on the colour-colour stellar locus, using some separate code - this should help eliminate compact galaxies and blended close pairs of stars or binaries (unless they have similar colours). When `makeglobalpsf` runs, check the output to make sure that not too many stars have been rejected (likely a sign of a bad input catalogue, bad positions, or too low a cut in S/N).

A.6 Input galaxy catalogues and images

`lensfit` is designed to work with multiple input images, including both the case where the same field has many exposures, or the case where there are multiple exposures with position shifts ("dithers") between them.

The general principle is that any one galaxy can appear on some number of multiple exposures, but that there are also exposures that do not contain that galaxy. To optimally combine the data `lensfit` needs to have access to all the exposures for each galaxy, so the most obvious way of achieving this is to give all input images and a catalogue of all galaxies to `lensfit` . The simplest way of then proceeding is to set the global variable WCS to a value 1 in `lensfit` and to give `lensfit` a catalogue of WCS positions, one for each galaxy. For this to work your WCS positions and the WCS coordinate transformation in the FITS headers must be accurate (ideally, as accurate as you would wish if you were going to stack multiple images).

If you wish to work with simulations or with images that don't contain WCS header information, the procedure is a little more complex for multiple input images, as described below. The catalogue should comprise the x, y positions in pixel coordinates of each galaxy on each CCD image. If a galaxy does not appear on any one CCD image then it can be given a default position that is "off-image", e.g. $x = -999, y = -999$ and `lensfit` will then know that image does not contain data for that particular galaxy.

For a large survey the total image set may exceed the available memory, in which case `lensfit` divides the input catalogue into sections such that the data may be held in memory, processing all galaxies in a subset before re-reading the images for the next. `lensfit` needs to be told how much memory it can use, via the MEMORY_LIMIT definition.

The code and files provided give an example for CFHTLenS which you can adapt to suit your survey. Megacam has 36 non-overlapping CCDs, and the CFHTLS survey was made with 7 exposures of Megacam with slightly offset positions for each. Thus any one galaxy may appear on a maximum of 7 images, and needs storage for up to 7 postage stamps. This is a parameter that is needed to define the amount of memory required and its value is set to be ⟨endexp⟩-⟨startexp⟩+1. If the number of input images that one galaxy appears on exceeds this value then an error message will be generated and program execution will halt.

The input catalogue comprises some comment lines starting with # which are ignored. Then, each galaxy has data on one line of the catalogue. The first four variables on each line are the world ra, dec coordinates, the catalogue galaxy magnitude and an optional input ID number. If only WCS values are present the code will still work, a default magnitude is assigned to each galaxy, but this is not recommended as the magnitude information is needed for the magnitude-dependent prior on scalelength. The input catalogue ID is simply passed through to the output catalogue. and if absent is generated in the output from the catalogue row number. If WCS=1 only the WCS coordinates are used by `lensfit` and no further information is required in the input catalogue. If WCS=0 then the code expects to read further x, y coordinate information from the catalogue file, and the world coordinates are not used but are reproduced in the output so that each galaxy may be identified (this might be useful for simulations without astrometry information).

For the case of x, y catalogue input only, following these four variables are $n_{exposure}$ sets of three numbers. The three numbers are the chip numerical ID (e.g.. covering the range 1-36 to specify which of the 36 CCDs in Megacam the galaxy appears on) and the x, y position of that galaxy on that chip. There are $n_{exposure}$ sets of such numbers to match each of the exposures. The number of exposures must match the number `nmaximages` specified in the code, and the total 7×36 (for example) must match the total number of input images that are given to `lensfit` . Then, `lensfit` needs to know how to match each chip/position in the catalogue to the list of input images, and it does this simply by *assuming* an input order for the images that matches the exposures given in the catalogue - i.e. there is no check that what is specified in the catalogue

matches the input images, this is up to you to get right. If you get it wrong, `lensfit` will run but you will probably find a high number of galaxies with "bad likelihood surfaces", fitclass −3. This catalogue input option has not been recently tested.

It is useful to be able to run `lensfit` on individual exposures extracted out of the above catalogue. There are therefore two arguments on the command-line, "exposure$_1$" and "exposure$_n$" that will tell `lensfit` to only use exposures in that range. For example specifying "1 7" will take all seven exposures from the catalogue. Specifying "3 3" will only analyse the third exposure in the catalogue. Again there is no checking done that an x, y catalogue matches the input image list - you must make sure this is correct. If you only specify a subset of exposures to be analysed, then you must supply only the images associated with that exposure in the input image list.

Otherwise, if WCS input is specified, `lensfit` will simply look on every image for every galaxy. The actual values used for ⟨startexp⟩ and ⟨endexp⟩ are not important but the total number of images on which a galaxy is found must not exceed ⟨endexp⟩-⟨startexp⟩+1.

A.7 Filenames

The PSF codes and `lensfit` make some assumptions about the filenames expected, and in some cases use those filenames to work out the association between an image and its position within the camera field. The supplied `makeglobalpsf` algorithm assumes the Megacam geometry (4×9 chips) with some specified gaps between each - for a different camera the geometry assumption and chip gaps will need to be modified. Filenames are also assumed to be of the form xxxxxxp_NC.sub.fits where the files are assumed to exist in directory DATA_DIR, and where xxxxxx is a 6-character identifying string, N is a number indicating the chip position (i.e. for Megacam N is in the range 1–36). If `makeglobalpsf` files are supplied to `lensfit` this code also needs to understand the geometry of the camera of course - some variables are passed through the PSF FITS files.

Also, the PSF code and `lensfit` will look for weight files called xxxxxxp_NC.weight.fits(.gz) in directory BADPIX_DIR. If CORRECT_DISTORTION is specified, the programs will also look for scamp header files in directory HEAD_DIR with name xxxxxxp_N.head (i.e. parts of the filename are stripped off). The ".head" suffix is specified in the scamp/swarp configuration file. If your filename conventions are different all the above will need modifying. Hopefully future releases will improve this aspect of the code.

Input FITS files may be gzipped if the GZIPPED global variable is set appropriately.

A.8 An important note on galaxy positions for multiple image input

For the purposes of worrying about how to specify galaxy positions, the process of optimum image measurement can be thought of somewhat as a similar process to co-adding images to make a deeper stack. To do that one has to make sure that the registration of the images is as good as it can be. Thus it would be wrong to simply measure the apparent position of each galaxy independently on each input image and supply that to `lensfit` , because for faint galaxies random noise introduces a random position error which would seriously degrade the image combination. *It is essential that each galaxy be first assigned a single unique world coordinate*. Then, either that coordinate can be supplied directly to `lensfit` , or you can yourself calculate the pixel coordinates on each image from a global astrometric transformation onto each image. It does not matter if that global position is inaccurate at the level of a few pixels, the important point is that there should not be random shifts between the positions of a galaxy on its multiple postage stamps. The

safest and easiest way of proceeding is to give the WCS coordinates directly to `lensfit` and let it do the astrometric conversions needed, provided you have accurate WCS information in every FITS image header.

A.9 Swarp input

If you want to correct for astrometric distortion you must specify CORRECT_DISTORTION in both the PSF code and `lensfit` and you must supply a swarp configuration file. An example is provided in the fileset, named `input_files/create_coadd_swarp.swarp`.

A.10 Flux calibration

If multiple images are to be combined, `lensfit` must know the relative flux scales of those images so that a unique model may be jointly fitted to the full dataset for each galaxy. `lensfit` therefore assumes that the pixel values have been flux calibrated, i.e. that any one galaxy has the same total image counts on each of the images on which it appears. (This assumption implicitly assumes that each image is in the same waveband - this is not fundamental to the method but if images in differing wavebands are to be combined some changes to `lensfit` would be required). If a swarp configuration file is provided, it reads the keyword name specified by FSCALE_KEYWORD in the swarp configuration file and applies the value of that keyword in each individual fits file header.

A.11 Ellipticity sampling

`lensfit` uses an adaptive grid to optimally sample the ellipticity probability surface for each galaxy. However, to achieve a multiplex advantage in compute time it pre-computes all the galaxy models on a fine grid in ellipticity. Ideally this grid could be arbitrarily fine, but as the models have to be stored, it rapidly becomes prohibitive to make the ellipticity sampling too fine. For CFHTLS we adopt a sampling of 0.02 which is fine enough to avoid obvious digitisation of output ellipticity but maintains a manageable amount of memory (typically 2 GB for the models). The ellipticity sampling is set inside `lensfit` with the variable `e_interval`. For use on a 32-bit machine you will need to run `lensfit` with coarser sampling.

A.12 Overall memory use, compute time and threads

The models tend to dominate the memory use, as indicated above. The memory required depends on $\texttt{e\_interval}^{-2}$ and on the postage stamp size as $\texttt{pwidth}^2$.

The data postage stamps also require memory, depending on $\mathbf{N}.\texttt{pwidth}^2.\texttt{nmaximages}$ where `nmaximages` is the maximum number of exposures needed for each galaxy (see above).

The required compute time per galaxy primarily depends on the postage stamp size as $(\texttt{pwidth})^2.\mathbf{log}(\texttt{pwidth})$ and only depends on the number of images very slowly, provided the memory required for the setup does not exceed the available system memory. *lens*fit manages the allocated memory by checking the allocated amount against the maximum allowed as specified by the `#define MEMORY_LIMIT` statement in the `lensfit` code. If too much memory would be required to process the entire catalogue, `lensfit` attempts to process it in chunks small enough to fit into memory. As multiple data reads are then required, this slows up the processing time, but not by too much.

To make optimum use of your machine's multiple cpus and available shared memory, `lensfit` uses threads. The number of threads should be set to the number of cores (e.g. NUM_THREAD 8 for a quad dual-core system).