glafic: gravitational lens adaptive-mesh fitting code

(version 2.1.0)

Masamune Oguri

June 23, 2021

Contents

Acknowledgement

The development of this gravitational lens software owes a great deal to existing public softwares, *gravlens* by Chuck Keeton and *glamroc* by Ted Baltz. Actually I "borrow" a lot from these codes' concepts, designs and user interfaces. I thank Ted Baltz also for stimulating discussions about algorithms for solving lens equation. I'm grateful to many people including Issha Kayo, Eduard Rusu, Anupreeta More, Keiichi Umetsu, Fabian Koehlinger, Gabriel Caminha, Christoph Becker, Akiyoshi Tsujita, Tsuyoshi Ishida, Jeremy Lim, and Mandy Chen for feedbacks and comments. I also thank members of Hyper Suprime-cam weak lensing working group for many useful discussions on imaging simulations.

Chapter 1

Overview

1.1 General descriptions

`glafic` [1] is a software for studying strong gravitational lenses (Oguri 2010). Its main functions are to solve lens equation, to calculate lensed images for both point and extended sources, and to model observed strong lensing. Adaptive-meshing with increased resolution near the critical curves allows a fast and efficient computation of lensed images. It is designed to model strong lensing by a cluster of galaxies, which often involves multiple lensed sources at several different redshifts, but it should be useful to model galaxy-scale lenses too.

I think the code works fine. That said, I basically make no guarantee of it. Please use it at your own risk, though I'm more than happy to hear what you feel on the code. If you have any questions, suggestions, and comments, please contact me.

To help you get a sense of how the code works, here I give some examples. Figure 1.1 basically shows mesh structure, critical curves and caustics, and example source and corresponding image positions. The figure demonstrates how the adaptive-meshing scheme successfully resolves critical curves, including those associated with small clumps. Also shown in Figure 1.2 are example lensed images of extended sources. The code can handle various brightness profiles for extended sources. One can also generates lensed mock-images of many galaxies using the code (Figure 1.3).

1.2 Features

- Basically `glafic` handles dimensional units. In particular, the lens and source planes are always in units of arcsecond. This makes it easy and straightforward to compare the results with observations.

- `glafic` supports a wide range of current standard lens mass models (e.g., point mass, SIE, NFW, Einasto, various perturbations, ...).

- It also supports a wide range of sources, including point sources and extended sources, multiple sources at different redshifts, etc.

[1] https://www.slac.stanford.edu/~oguri/glafic/

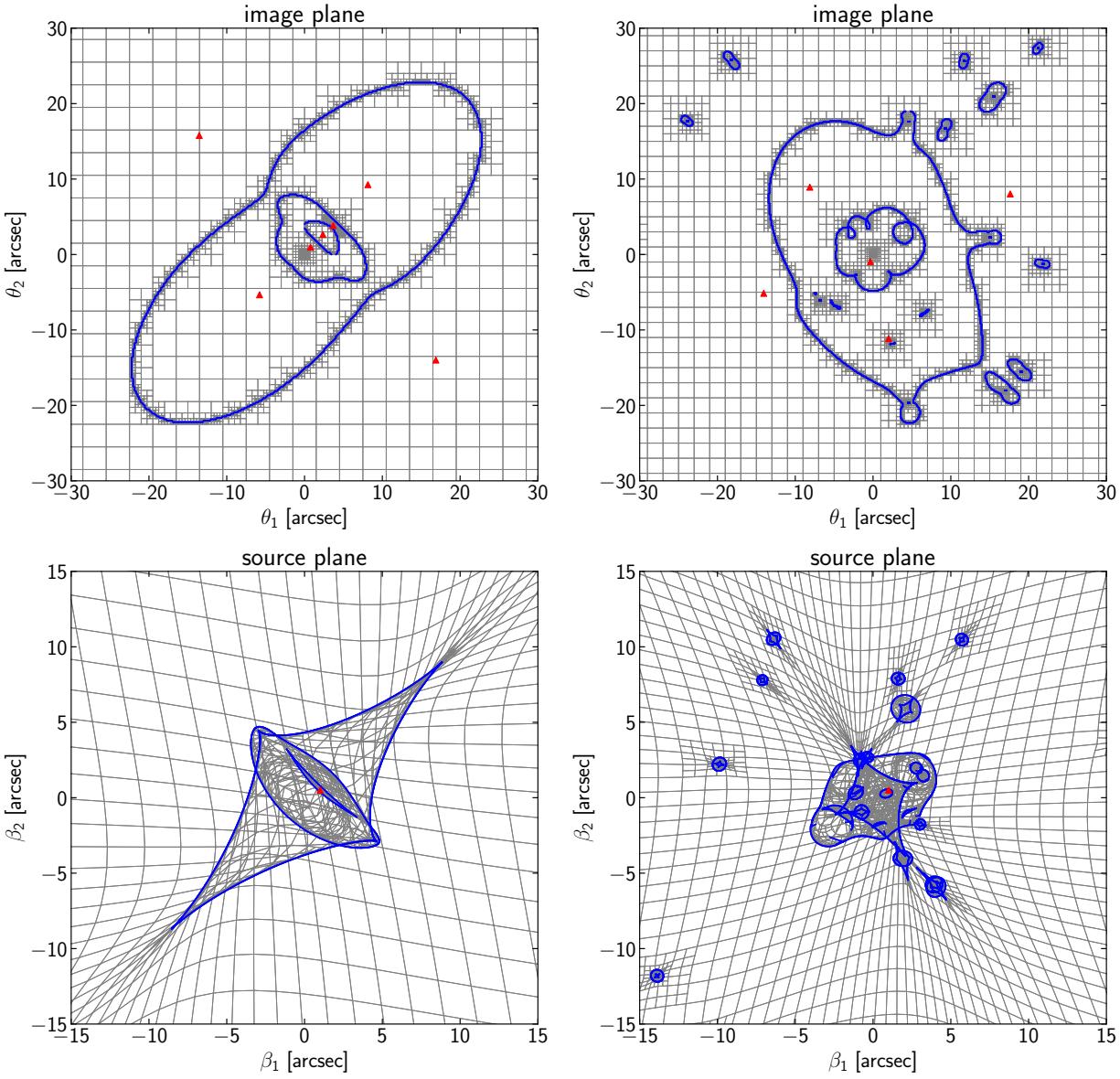

Figure 1.1: Example of lens equation solving for point sources. I use square grids (*thin black lines*) that are adaptively refined near critical curves to derive image positions for a given source. Upper panels show image planes, and lower panels are corresponding source planes. Critical curves and caustics are drawn by blue lines. Positions of sources and images are indicated by red triangles. Left panels show an example from a simple mass model that consist of NFW and SIE profiles at different redshifts. A source near the center is producing 7 lensed images. In right panels, I add small galaxies to the primary NFW lens potential. This time 5 lensed images are produced.

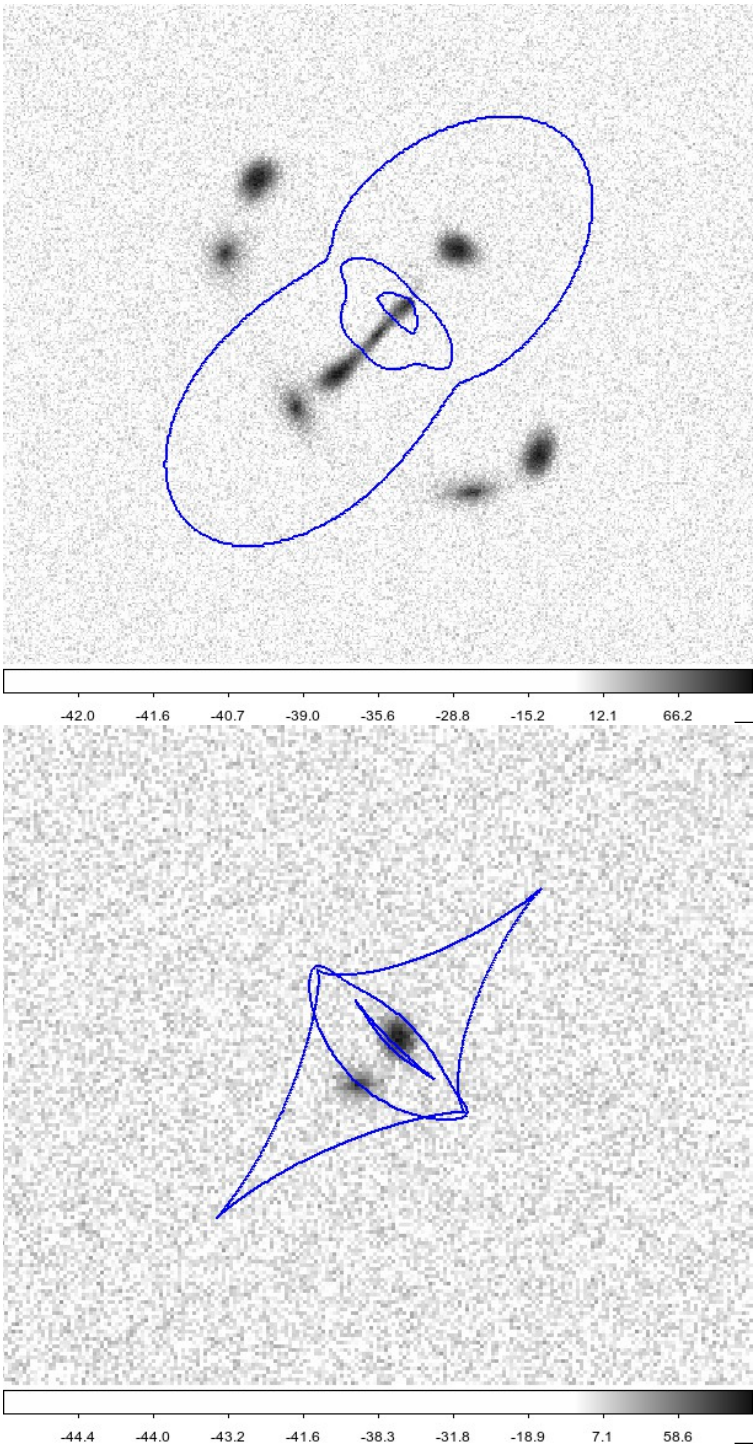

Figure 1.2: Example of lens mapping for extended sources. The lens model is same as the one that was used in the left panels of Figure 1.1. I put two sources with different orientations and sizes in the source plane (*lower panel*). Lensed images are shown in the image plane (*upper panel*).

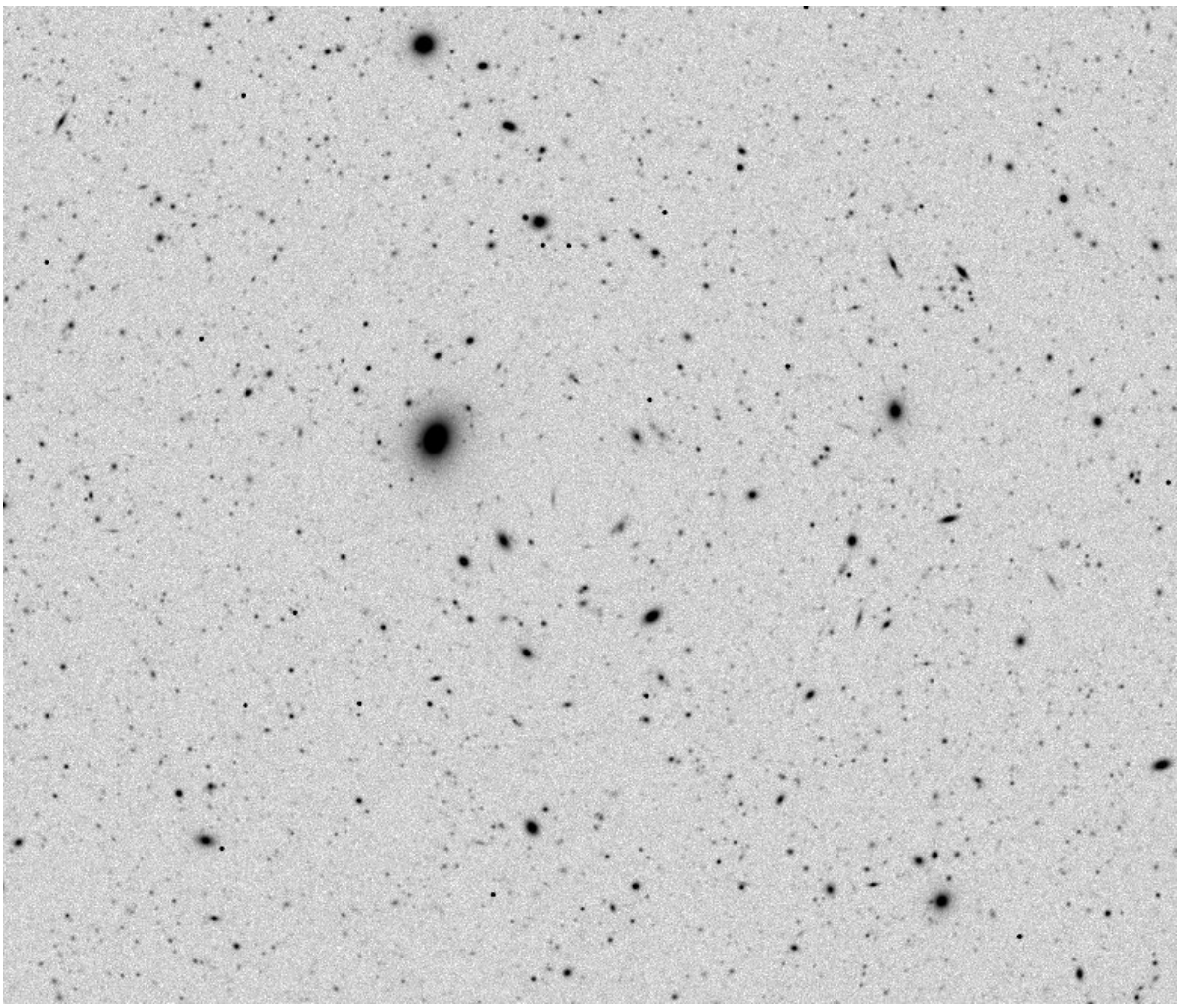

Figure 1.3: Simulated wide-field image. Galaxy images of the central region are distorted due to the foreground cluster.

- Functional lens model optimizations with many options including source-plane χ^2 minimizations, randomization/reoptimizations, and Markov-chain Monte-Carlo.

1.3 Updates from version 1

Several features are newly implemented in `glafic` version 2. As a result, input files for version 2 are not compatible with those for version 1. Here is a quick summary of important updates.

- Multiple lens planes (gravitational lensing by multiple structures at different redshifts) are now fully supported.
- Because of this, lens redshifts are now specified as one of model parameters of each lens, rather than as a primary parameter. Hence, 8 parameters are used to characterize each lens component (cf. 7 in version 1). Input files, and most likely prior files as well, in version 1 need to be updated to reflect this change to get them work in version 2.
- Default values of cosmological parameters are changed.
- The source code is now free from Numerical Recipes and is publicly available in github[2].
- Because of changes of some routines such as numerical integrations and interpolations, as well as changes of some constants (such as speed of light) to include more digits, output values can be slightly different between version 1 and 2, even for the same input parameters. Because of changes in the optimization (χ^2 minimization) routine, `optimize` can also behave differently. The differences should be small enough and therefore should not be crucial in most cases.
- The fast analytic calculation of `pow` lens model based on Tessore & Metcalf (2015) is implemented. Fast approximated NFW (`anfw`) and and Hernquist (`ahern`) models based on Oguri (2021) are also implemented from version 2.1.0.
- Although it is not covered in this manual, the python interface is now available.

[2]https://github.com/oguri/glafic2

Chapter 2

Running glafic

2.1 Getting Started

glafic is run from the command-line in unix/linux/mac. After setting the path to the binary, you can start glafic simply by

```
foo% glafic file.input
```

where file.input is an input text file that specifies the parameters, mass model, and commands executed. Please note that glafic *requires* this input file to be run. The format of the file will be explained in § 2.2. Unless the input file contains a command quit, it automatically enters an interactive mode — you'll see a prompt like this

```
glafic>
```

You can execute commands here to check results, to do modeling further, etc. To end glafic please use the quit command.

```
glafic> quit
```

2.2 Format of Input File

The input file consists of parameter setting, the definition of mass models and sources used in the run, and a list of commands. The basic structure of the input file is as follows:

```
 [parameters setting]

startup
 [define mass models and sources]
end_startup

start_setopt
 [define flags for optimizations]
end_setopt
```

```
start_command
 [your commands here]
```

The file should start with setting of parameters, both primary and secondary parameters, followed by the definition of mass models and sources. All the commands after `start_command` will be executed in order. Any lines that start with `#` will be ignored — so you can insert any comments in the input file with this.

In the beginning of the input file, parameters can be defined by simply listing parameter names and corresponding parameter values. Examples of these are

```
omega     0.3
lambda    0.7
maxlev    5
 ....
```

Next you should define mass models used in the code between `startup` and `end_startup`. In the first line containing `startup`, the numbers of lenses, extended sources, and point sources have to be specified. It is followed by sentences specifying individual components (the order matters). Here is an example:

```
startup 1 2 3
lens  sie  0.5 300.0  0.0 0.0 0.2 45.0 0.02 0.0
extend sersic 1.5 150.0  0.0  0.5 0.3 90.0 0.8 1.0
extend gauss  1.5 150.0 -0.2  0.3 0.2 10.0 0.6 0.0
point 1.5 -0.2 0.1
point 2.5  0.6 0.4
point 2.5  0.8 0.3
#psf 0.8 0.02 60.0 5.0 1.2 0.02 -30.0 3.0 0.7
end_startup
```

The `lens` component consists of the name of the mass model and 8 parameters characterizing the model. An extended source is described by the name of the source model and 8 model parameters. A point source requires just 3 parameters, the redshift and xy-position. The last one, `psf`, is optional; it is used only for extended sources and specifies the shape of PSF. If not included (note that you may also include PSF by supplying a fits file, as shown later), the extended source images are created without convolving with PSF. You can use `#` to comment out any of these components (however note that number of components specified at the beginning and the number of subsequent component entries must agree).

If you are doing χ^2 minimizations, you need to indicate which parameters you want to vary. In the input files, this is done between `start_setopt` and `end_setopt`. An example is

```
start_setopt
0 1 0 0 1 1 0 0
0 1 1 1 1 1 0 0
0 1 1 1 1 1 0 0
0 1 1
```

```
1 1 1
1 1 1
#0 0 0 0 0 0 0 0 0
end_setopt
```

The structure above must be exactly same as the one indicated by `startup`. 1 indicates that the parameter is varied, whereas 0 means the parameter is fixed.

2.3 Example of Input File

`glafic` provides an example input file. To see it, please just run `glafic` with `-d` option:

```
foo% glafic -d
```

You can also save the example to a file and run it for a test:

```
foo% glafic -d > example.input
foo% glafic example.input
```

It will help you to learn about the code, e.g., how it runs commands, what are outputs, how to write input files, etc.

Chapter 3

Parameter Reference

3.1 Primary Parameters

Primary parameters consist of most important parameters whose values I strongly recommend to specify explicitly at the beginning of the input file. These parameters, however, can be changed afterward by using the command `reset_par` (see §5.3). Table 3.1 summarizes the primary parameters. Below are brief descriptions of these parameters.

- `omega`, `lambda`, `weos`, `hubble`
  These are standard cosmological parameters, Ω_M, $\Omega_{\rm DE}$, w_0, and h, respectively.

- `prefix`
  The prefix for any output files. For example, the result of lens mass modeling is written in a file named *prefix*`_optresult.dat`. Note that results are overwritten.

- `xmin`, `ymin`, `xmax`, `ymax`
  These parameters specify the rectangular region of the lens plane in which lens equation is solved (in units of arcsecond). Any images outside the region will not be identified by the code, so you should select the region large enough to cover all possible images of interest (but small enough to speed up calculations).

- `pix_ext`
  The pixel (grid) size for extended sources, in units of arcsecond. No subgrid is made for extended sources, and thus this pixel size corresponds to the spatial resolution for extended sources.

- `pix_poi`, `maxlev`
  `pix_poi` denotes the largest pixel (grid) size for point sources, in units of arcsecond. `maxlev` gives the maximum recursion level of adaptive meshing. If `maxlev` is greater than 1, the subgrid recursion is performed to enhance the resolution near critical curves, and thus the resulting minimum pixel size is $\texttt{pix\_poi}\times 2^{1-\texttt{maxlev}}$.

Name	Description	Type	Default
`omega`	matter density Ω_M	double	0.3
`lambda`	dark energy density $\Omega_{\rm DE}$	double	0.7
`weos`	dark energy EOS w_0	double	-1.0
`hubble`	Hubble constant h	double	0.7
`prefix`	output file name prefix	char	out
`xmin`	x-coord minimum [arcsec]	double	-60.0
`ymin`	y-coord minimum [arcsec]	double	-60.0
`xmax`	x-coord maximum [arcsec]	double	60.0
`ymax`	y-coord maximum [arcsec]	double	60.0
`pix_ext`	pixel size for extended sources [arcsec]	double	0.2
`pix_poi`	largest pixel size for point sources [arcsec]	double	3.0
`maxlev`	maximum level of adaptive meshing	int	5

Table 3.1: Primary parameters.

3.2 Secondary Parameters

Secondary parameters are less important parameters which control various commands. One can define these parameters at the beginning of the input file, or afterward as well by using the command `reset_par` (see §5.3). Table 3.2 summarizes some of secondary parameters that are frequently used in the code.[1] The brief explanations of these parameters are shown below.

- `galfile`
  The name of the data file used for the mass model `gals`. The format of the file should be

  ```
  x[1] y[1] L[1] e[1] PA[1]
  x[2] y[2] L[2] e[2] PA[2]
  .....
  ```

  where `x[i]` and `y[i]` are xy-coordinates of i-th galaxy, `L[i]` is the luminosity normalized by a fiducial luminosity L_*, `e[i]` is an ellipticity, and `PA[i]` is a position angle measured East of North (assuming North is up).

- `srcfile`
  The name of the data file used for the extended source model `srcs`. The format of the file should be

  ```
  f[1] x[1] y[1] e[1] PA[1] re[1] n[1]
  f[2] x[2] y[2] e[2] PA[2] re[2] n[2]
  .....
  ```

[1] There are some additional secondary parameters, which are used for minor tweak of some commands.

i.e., parameters `p[2]`–`p[8]` for the `sersic` model should be indicated in each line. If `re`<= 0 or `n`<= 0, PSF is drawn at the position (apparently PSF must be define for this).

- `ran_seed`
  Seed for the random number generator, which has to be a negative integer.

- `outformat_exp`
  This is a flag whether to use the exponential form for some result outputs. If = 0 result values are shown in normal fixed-point notation, whereas for = 0 values are shown in the exponential form. This flag affects only outputs related to lensed point source images (e.g., `findimg` command), the Einstein radii (`calcein` command), and lensing properties (`calcimage` command).

- `flag_hodensity`, `hodensity`
  The flag `flag_hodensity` specifies which nonlinear overdensity is used, for mass models `nfw`, `gnfw`, `tnfw`, and `ein`. In the case of `flag_hodensity`=0, the overdensity computed from the spherical collapse model is adopted (see Appendix C). On the other hand, for `flag_hodensity`=1 the code uses a fixed mean overdensity (i.e., $\bar{\rho}(< r_{\rm vir}) = \Delta \times \bar{\rho}_{\rm M}(z)$), and for `flag_hodensity`=2 a fixed critical overdensity (i.e., $\bar{\rho}(< r_{\rm vir}) = \Delta \times \rho_{\rm crit}(z)$) is adopted. The value of the fixed overdensity Δ can be specified by `hodensity`.

- `gnfw_usetab`, `ein_usetab`
  Computing lensing properties of generalized NFW profiles and Einasto profiles are time-consuming (see Appendix B), and thus I prepared tabulated models for these profiles. If the parameter is set to =1, the code makes tables of lensing properties when the model is first called (it can take a few minutes), and estimates all the deflection angles etc. by interpolating the table values. If you do not want to use the tabulated model, just set the parameter to =0.

- `nfw_users`
  If the parameter is set to =1, the code uses r_s in arcsec for the model parameter `p[7]`, instead of c or c_{-2}, for mass models `nfw`, `gnfw`, `tnfw`, and `ein`.

- `flag_extnorm`
  This indicates normalizations of extended sources. If =0, the peak surface brightness per pixel, Σ_0, is used as the normalization of the model, i.e., the parameter `p[2]` of the extended source. If =1, the total counts $f_{\rm tot}$ is adopted for the definition of `p[2]`. See also §4.2.

- `chi2_checknimg`
  This is a flag used for point source optimizations. If =1 the code checks the number of images, and discard any models that predict different number of images from the observation. If =0 no check is made during the optimization. This option does not work for the source-plane χ^2.

- `chi2_splane`
  Another flag used for point source optimizations. While the standard image-plane χ^2

is used for =0, the code uses the source-plane χ^2, which is much faster to compute but less robust and accurate, if =1.

- `chi2_usemag`
  This is for the command `readobs_point`. If =1, magnitude differences has to be specified rather than the (default) relative fluxes. Some discussions are also given in §5.2.

- `chi2_restart`
  The is a parameter for optimizations of both point and extended sources. It indicates the number of restarts — repeated optimizations from the previous best-fit models to check whether it reaches a true χ^2 minimum or not. Thus the code performs optimizations (`chi2_restart`+1) times in total. If a negative number is indicated, the code repeats optimizations until χ^2 converges.

- `obs_gain`, `obs_ncomb`, `obs_readnoise`
  These parameters are for observed extended images ready by the `readobs_extend`. They are nothing but gain (i.e., the conversion factor between ADU and electron counts) and number of frames combined for the input fits image. Then a noise for each pixel is computed by

$$\sigma^2 = \sigma^2_{\rm sky} + (flux - \langle sky \rangle)/(gain \times ncomb). \tag{3.1}$$

  When adding the noise to the image, the Gaussian distribution is assumed for this case. If the sky noise of < 0 is indicated, the code compute the noise from the total count from object and sky, and add `obs_readnoise` to the noise. Specifically,

$$\sigma^2 = flux/(gain \times ncomb) + readnoise \times readnoise/(gain^2 \times ncomb). \tag{3.2}$$

  This time, the Poisson distribution is used to assign a noise to each pixel.

- `skyfix`, `skyfix_value`
  The flag `skyfix` specifies whether to vary a sky level when optimizing extended sources (sky level is fixed if =1, varied and optimized if =0). `skyfix_value` indicates the fixed value of sky counts; if `skyfix_value`$> 10^9$ the fixed sky value is estimated directly from the input fits image. When `skyfix`= 0, `skyfix_value` specifies the initial sky value for optimization.

- `psfconv_size`
  `psfconv_size` specifies half the box size of the PSF image used for the convolution, in units of arcsec. The software crashes if the convolution size is too big compared with the image size for extended sources, in that case please choose the smaller `psfconv_size`.

- `seeing_sub`
  If `seeing_sub`> 1, oversampled PSF by a factor of `seeing_sub` is used for the PSF convolution. Note that various interpolations are employed to derive values in subpixels.

- `flag_srcsbin`, `srcsbinsize`
  The extended source model `srcs` sometimes contains a large number of sources, and for this case the calculation gets quite time-consuming. One can get around this by dividing sources into bins. `flag_srcsbin` specifies whether to use this binning. The bin size `srcsbinsize` should be chosen such that it is much larger compared with typical source sizes.

- `flag_mcmcall`
  If `flag_mcmcall` is non-zero, model parameters for rejected points are also written to the MCMC output file. Commands that use a MCMC output file (e.g., `mcmc_kaprad`) support both the `flag_mcmcall`=0 and !=0 output formats.

- `addwcs`, `wcs_ra0`, `wcs_dec0`
  If `addecs` is non-zero, WCS info is added to fits output files. The coordinates corresponding to (x, y)=(0, 0) must be specified by `wcs_ra0` and `wcs_dec0`. Currently $+y$ is always North and $+x$ is always West.

- `ovary`, `lvary`, `wvary`, `hvary`
  Flags whether to vary corresponding cosmological parameters when optimizations. The cosmological parameters will be optimized if =1, and fixed if =0.

Name	Description	Type	Default
galfile	filename for the mass model gals	char	galfile.dat
srcfile	filename for the extended source model srcs	char	srcfile.dat
ran_seed	seed for the random number generator	int	−1234
outformat_exp	flag for whether to use the exp format	int	0
flag_hodensity	flag for the halo overdensity	int	0
hodensity	fixed halo overdensity Δ	double	200.0
gnfw_usetab	flag for whether to use a tabulated model for gnfw	int	1
ein_usetab	flag for whether to use a tabulated model for ein	int	1
nfw_users	flag for whether to use r_s [arcsec] instead of c or c_{-2}	int	0
flag_extnorm	flag for the normalization of extended sources	int	0
chi2_checknimg	flag for whether to check the number of images	int	1
chi2_splane	flag for whether to use the source-plane χ^2	int	0
chi2_usemag	use magnitude differences instead of flux ratios	int	0
chi2_restart	number of restarts in χ^2 minimizations	int	0
obs_gain	gain [e-/ADU] for extended image	double	3.0
obs_ncomb	number of combined frames for extended images	int	1
obs_readnoise	readout noise [e-] for extended images	double	10.0
skyfix	flag for whether to fix the sky value	int	0
skyfix_value	fixed sky value	double	1.0e10
psfconv_size	half the box size for the PSF convolution [arcsec]	double	4.0
seeing_sub	oversampling factor for PSF convolution	int	1
flag_srcsbin	flag for whether to use bins for srcs	int	1
srcsbinsize	bin size for binned srcs calculation [arcsec]	double	20.0
flag_mcmcall	flag for whether to dump rejected points in MCMC	int	0
addwcs	flag for whether to add WCS info in fits outputs	int	0
wcs_ra0	RA value corresponding to $x = 0$	double	150.0
wcs_dec0	Dec value corresponding to $y = 0$	double	30.0
ovary	flag for whether to vary omega	int	0
lvary	flag for whether to vary lambda	int	0
wvary	flag for whether to vary weos	int	0
hvary	flag for whether to vary hubble	int	0

Table 3.2: Secondary parameters.

Chapter 4

Model Reference

4.1 Lens

As described in §2.2, lens models are specified at the beginning of the input file, between `startup` and `end_startup`. The required format for lens models is

```
lens name p[1] p[2] p[3] p[4] p[5] p[6] p[7] p[8]
```

For all lens models, the parameters `p[1]` refers to the lens redshift z_l. The meaning of the other parameters is different for different mass models, but in general the parameter `p[2]` indicates a mass scale, `p[3]` and `p[4]` are xy-coordinates, `p[5]` and `p[6]` are an ellipticity and position angle, `p[7]` and `p[8]` are miscellaneous but often assigned to scale radii or power-law index.

Table 4.1 summarizes all available lens models. Note that calculations of various lensing properties are summarized in Appendix A, and detailed descriptions of each mass model in Appendix B.

For most mass models, the ellipticity is introduced either in isodensity contours or in isopotential contours. Mass models for the latter are indicated by the suffix `pot`. Note that the ellipticities for these two cases are different; e is the ellipticity of the isodensity contour, whereas e_p is the ellipticity of the isopotential contour. In general lens potentials are more circular than the corresponding surface mass densities, and thus for models with similar degree of ellipticities we have $e_p < e$. The position angle θ_e is defined East of North, assuming $+y$ direction is North. In other words, θ_e is the angle measured counterclockwise from $+y$-axis.

4.2 Extended Source

Sources should also be defined at the beginning of the input file, between `startup` and `end_startup`. The format for extended sources is

```
extend name p[1] p[2] p[3] p[4] p[5] p[6] p[7] p[8]
```

Models are summarized in Table 4.2, and the definition of each model in Appendix D. The definition of the parameter `p[2]` can be switched by changing the parameter `flag_extnorm`,

Name	Description	p[2]	p[5]	p[6]	p[7]	p[8]
pert	external convergence/shear	$z_{s,\mathrm{fid}}$	γ	θ_γ	—	κ
clus3	3rd external perturbation	$z_{s,\mathrm{fid}}$	δ	θ_δ	—	—
mpole	multipole perturbation	$z_{s,\mathrm{fid}}$	ϵ	θ_m	m	n
point	point mass	M_{tot}	—	—	—	—
sie	isothermal elliptical density	σ	e	θ_e	r_{core}	—
jaffe	pseudo-Jaffe elliptical density	σ	e	θ_e	r_{trun}	r_{core}
pow	power-law elliptical density	$z_{s,\mathrm{fid}}$	e	θ_e	r_{Ein}	γ
powpot	power-law elliptical potential	$z_{s,\mathrm{fid}}$	e_p	θ_e	r_{Ein}	γ
sers	Sérsic (elliptical density)	M_{tot}	e	θ_e	r_e	n
serspot	Sérsic (elliptical potential)	M_{tot}	e_p	θ_e	r_e	n
hern	Hernquist (elliptical density)	M_{tot}	e	θ_e	r_b	—
ahern	Hernquist (approx. elliptical density)	M_{tot}	e	θ_e	r_b	—
hernpot	Hernquist (elliptical potential)	M_{tot}	e_p	θ_e	r_b	—
nfw	NFW (elliptical density)	M	e	θ_e	c or r_s	—
anfw	NFW (approx. elliptical density)	M	e	θ_e	c or r_s	—
nfwpot	NFW (elliptical potential)	M	e_p	θ_e	c or r_s	—
gnfw	generalized NFW (elliptical density)	M	e	θ_e	c_{-2} or r_s	α
gnfwpot	generalized NFW (elliptical potential)	M	e_p	θ_e	c_{-2} or r_s	α
tnfw	truncated NFW (elliptical density)	M	e	θ_e	c or r_s	t
tnfwpot	truncated NFW (elliptical potential)	M	e_p	θ_e	c or r_s	t
ein	Einasto (elliptical density)	M	e	θ_e	c_{E} or r_s	α_{E}
einpot	Einasto (elliptical potential)	M	e_p	θ_e	c_{E} or r_s	α_{E}
gals	many galaxies (sum of jaffe)	σ_*	—	—	$r_{\mathrm{trun},*}$	η

Table 4.1: Lens models. See Appendix B for detailed descriptions and definitions of model parameters for each mass model. For all source models, the parameters p[1] refers to the lens redshift z_l, and in most cases the parameters p[2] and p[3] denote the xy-coordinates of the model in units of arcsecond. All mass scales M is in units of $h^{-1}M_\odot$, the velocity dispersion σ in units of $\mathrm{km\,s^{-1}}$, angle scales θ in units of degree (measured "East of North", i.e., counterclockwise from $+y$-axis), and length scales r in units of arcsecond.

except for the point model for which the total flux is always used. Obviously, the PSF must be defined to use the point model.

4.3 Point Source

Point sources are defined simply by

```
point p[1] p[2] p[3]
```

where p[1] is the source redshift z_s, and p[2] and p[3] are xy-coordinates of the point source in units of arcsecond.

Name	Description	p[2]	p[5]	p[6]	p[7]	p[8]
gauss	Gaussian profile	Σ_0 or $f_{\rm tot}$	e	θ_e	σ	—
sersic	Sérsic profile	Σ_0 or $f_{\rm tot}$	e	θ_e	r_e	n
tophat	tophat source	Σ_0 or $f_{\rm tot}$	e	θ_e	r_0	—
moffat	Moffat profile	Σ_0 or $f_{\rm tot}$	e	θ_e	α	β
jaffe	pseudo Jaffe profile	Σ_0 or $f_{\rm tot}$	e	θ_e	$r_{\rm trun}$	$r_{\rm core}$
point	point source	$f_{\rm tot}$	—	—	—	—
srcs	many sources (sum of sersic)	$f_{\rm norm}$	—	—	—	—

Table 4.2: Models of extended sources. See Appendix D for the definition of each model. For all source models, the parameters p[1] refers to the source redshift z_s, and p[3] and p[4] indicates xy-coordinates of the source. Σ_0 is in units of counts/pixel^2, and θ_e is in units of degree (measured "East of North").

4.4 PSF

In glafic, an analytic PSF model based on the sum of two Moffat profiles is supported. It is specified by

```
psf p[1] p[2] p[3] p[4] p[5] p[6] p[7] p[8] p[9]
```

where p[1], p[2], p[3], p[4] are FWHM [arcsec], e, θ_e, β of the first Moffat component, respectively, and p[5], p[6], p[7], p[8] are those for the second Moffat component. p[9] is the fraction of the first component to the total, $f_1/(f_1+f_2)$, where f_1 and f_2 are" total fluxes" for the first and second Moffat components.

Chapter 5

Command Reference

5.1 Basic calculations

5.1.1 Lens properties

`calcimage` $\langle z_s \rangle$ $\langle x \rangle$ $\langle y \rangle$
: Compute various lensing properties for an image at (x, y) for the source redshift z_s.

`calcein` $\langle z_s \rangle$
: Compute the Einstein radius of each lens component for the source redshift z_s. Ellipticities are ignored in computing the Einstein radii. Also shown is the mass enclosed by the Einstein radius, i.e., $M(< \theta_{\rm Ein})$. The result is written to *prefix*`_ein.dat`.

`calcein2` $\langle z_s \rangle$ $\langle x \rangle$ $\langle y \rangle$ $\langle i \rangle$
: Compute the Einstein radius for the source redshift z_s from the definition $\bar{\kappa}(< r_{\rm Ein}) = 1$ with the center (x, y). The parameter i indicates the lens ID, i.e., κ is computed only from the corresponding lens model. If $i = 0$ (or not indicated), the sum of κ of all the lens models is used to compute the Einstein radius, taking account of multiple lens planes. The mass enclosed by the Einstein radius is also computed. The result is written to *prefix*`_ein2.dat`.

`calcmr`
: Compute the total mass, virial mass, and virial radius (both in arcsec and h^{-1}Mpc physical) for each lens model. If `flag_hodensity`$\neq$0, the virial mass and radius are replaced by the mass and radius for the given overdensity.

`writelens` $\langle z_s \rangle$
: Write various lensing properties to *prefix*`_lens.fits`, assuming the source redshift z_s. The output file contains tables of values in a regular grid with a pixel size defined by `pix_ext`. A list of lensing properties written in the file is (ϕ_x, ϕ_y, ϕ, κ, γ_1, γ_2, μ^{-1}, ω), in order of appearance.

`kapparad` $\langle z_s \rangle$ $\langle x \rangle$ $\langle y \rangle$ $\langle r_1 \rangle$ $\langle r_2 \rangle$ $\langle n \rangle$ $\langle i \rangle$
: Compute circular-average convergences κ with the center (x, y) for the source redshift z_s. They are computed in a range of radii between r_1 and r_2, with an interval of

$(r_2 - r_1)/n$ (thus κ is computed at $(n+1)$ different radii). If $n < 0$, a logarithmic spacing between r_1 and r_2 is adopted instead of a linear spacing. The parameter i indicates the lens ID (see `calcein2`). The result is written to *prefix*`_kaprad.dat`.

`kappacum` $\langle z_s \rangle$ $\langle x \rangle$ $\langle y \rangle$ $\langle r_1 \rangle$ $\langle r_2 \rangle$ $\langle n \rangle$ $\langle i \rangle$
Similar to `kapparad`, but the average κ within the radius, $\bar{\kappa}(< r) = 2\int_0^r \kappa(r)dr/r^2$, is computed. The result is written to *prefix*`_kapcum.dat`.

`lenscenter` $\langle z_s \rangle$
Derive observed centers (xy-positions) of all lens components. While in the case of a single lens plane, these are simply input positions, in the case of multiple lens planes, observed lens positions can be different from input lens positions, and this command allows one to check observed centers for given input centers. The result is written to *prefix*`_lenscenter.dat`. The format of the file is

```
lens_plane_ID lens_plane_z center_x center_y
....
```

5.1.2 Extended sources

`writeimage` $\langle sky \rangle$ $\langle noise \rangle$
Write images of lensed extended sources to *prefix*`_image.fits`. A mean sky level and 1σ noise can be specified by *sky* and *noise*, respectively. The fits data cube includes lensed images of individual extended sources first, and finally show images of all extended sources. The pixel size of the images is `pix_ext`. No noise is added if $noise = 0$. If $noise < 0$, the code derives noises for individual pixels from the sum of the counts from the objects and sky.

`writeimage_ori` $\langle sky \rangle$ $\langle noise \rangle$
Same as `writeimage`, but *unlensed* (i.e., original) images are shown. The output file is *prefix*`_source.fits`.

`calcextend` $\langle i \rangle$ $\langle f_{\rm th} \rangle$ $\langle r_{\rm lim} \rangle$
Calculate a total flux, peak count, and peak location for i-th extended image. If $i = 0$ (or not indicated) they are computed for all extended sources. $f_{\rm th}$ is the count threshold; the area and summed flux of pixels with values larger than $f_{\rm th}$ are also computed. For $-1 < f_{\rm th} < 0$, the threshold is set to $|f_{\rm th}|$ times the global peak value. If $f_{\rm th}$ is ignored (or $= 0$ is indicated), $f_{\rm th}$ is assumed to -0.5 (i.e., threshold equal to half the peak value). The code also compute the length-to-width ratio r of each arc, and derive the maximum ratio $r_{\rm arc}$. Also the number of arcs having $r > r_{\rm lim}$ is counted. The result is written to *prefix*`_extend.dat`. In addition, properties of arcs defined by $r > r_{\rm lim}$ are stored in *prefix*`_extend_arc.dat`. The format of the file is

```
x_arc y_arc r area_th ftot_th
....
```

`calcextend_ori` $\langle i \rangle$ $\langle f_{\rm th} \rangle$ $\langle r_{\rm lim} \rangle$
: Same as `calcextend`, but for *unlensed* (i.e., original) images.

`writetd_extend`
: Write time delay surfaces (Fermat potentials) for all extended sources to *prefix*`_td_extend.fits`. This command works only in the case of a single lens plane. The pixel size of the output fits is determined by `pix_ext`.

`mockext1` $\langle n \rangle$ $\langle i \rangle$ $\langle x_1 \rangle$ $\langle x_2 \rangle$ $\langle y_1 \rangle$ $\langle y_2 \rangle$ $\langle e_{\rm min} \rangle$ $\langle e_{\rm max} \rangle$ $\langle f_{\rm th} \rangle$ $\langle r_{\rm lim} \rangle$
: Randomly redistribute i-th extended sources for n times, in a rectangular region defined by $x_1 < x < x_2$ and $y_1 < y < y_2$. The ellipticity is also redistributed in a range $e_{\rm min} < e < e_{\rm max}$ with a random position angle. For each source, image properties computed by `calcextend` are obtained. The result is written to *prefix*`_mockext.dat`. The format of the file is

```
# n area model p[1] p[2] p[7] p[8] r_lim
xs ys e PA ftot pflux px py fth area_th ftot_th r_max n_arc
....
```

  Also properties of arcs are written to *prefix*`_mockext_arc.dat`.

`mockext2` $\langle n \rangle$ $\langle i \rangle$ $\langle r \rangle$ $\langle x_0 \rangle$ $\langle y_0 \rangle$ $\langle e_{\rm min} \rangle$ $\langle e_{\rm max} \rangle$ $\langle f_{\rm th} \rangle$ $\langle r_{\rm lim} \rangle$
: Same as `mockext1`, but this time the region is defined by a circle with a center (x_0, y_0) and a radius r.

`mockext3` $\langle n \rangle$ $\langle i \rangle$ $\langle f_{\rm rec} \rangle$ $\langle e_{\rm min} \rangle$ $\langle e_{\rm max} \rangle$ $\langle f_{\rm th} \rangle$ $\langle r_{\rm lim} \rangle$
: Same as `mockext1`, but the rectangular region is defined by $f_{\rm rec}$ times the extent of the caustics.

`writepsf`
: Write the PSF image to *prefix*`_psf.fits`.

`readpsf` $\langle psffile.fits \rangle$
: Read the PSF fits file and use it for calculations of extended source images. The input PSF file has to be similar to that used in `galfit`, i.e., it must be a square, centered and background-subtracted. The input PSF image overrides PSF parameters.

5.1.3 Point sources

`findimg` $\langle i \rangle$
: Find all lensed images for point sources. Magnifications and time delays for these images are also calculated. The parameter i specifies the point source ID, i.e., the lens equation is solved for only i-th point source. If not specified (or $i = 0$), the code will find lensed images for all point sources. In the display output, the magnitude without zero-point for each image, i.e., $-2.5 \log |\mu|$, is shown in the square bracket following the magnification. The result is written to *prefix*`_point.fits`.

`findsrcimg` $\langle i \rangle$ $\langle x_i \rangle$ $\langle y_i \rangle$
: Move the source position of the i-th image to the position calculated from an image position (x_i, y_i), and then perform `findimg` for the new source position.

`writecrit` $\langle z_s \rangle$
: Compute critical curves and caustics for the source redshift z_s. The output filename is *prefix*`_crit.dat`. Each line of the file contains following entry:

```
xi[1] yi[1] xs[1] ys[1] xi[2] yi[2] xs[2] ys[2]
```

A set of segments connecting (`xi[1]`, `yi[1]`) and (`xi[2]`, `yi[2]`) represents critical curves of the lens system. Similarly, a set of segments defined by `xs` and `ys` represents caustics. The resolution of these curves is determined by `pix_poi` and `maxlev`.

`writemesh` $\langle z_s \rangle$
: Write the mesh pattern used to find lensed point images for the source redshift z_s to *prefix*`_mesh.dat`. The structure of the output file is same as that of the `writecrit` output (see above).

`writetd_point`
: Write time delay surfaces (Fermat potentials) for all point sources to *prefix*`_td_point.fits`. This command works only in the case of a single lens plane. The pixel size of the output fits is determined by `pix_ext`.

`writelens_splane` $\langle z_s \rangle$ $\langle x_1 \rangle$ $\langle x_2 \rangle$ $\langle y_1 \rangle$ $\langle y_2 \rangle$ $\langle pix \rangle$
: Write lens properties in the source plane to *prefix*`_lens_splane.fits`, in the rectangular region of the source plane specified by x_1, x_2, y_1, and y_2, with a pixel size of pix. A list of lensing properties written in the file is the total magnification, the maximum magnification among multiple images, and the number of multiple images.

`mock1` $\langle n \rangle$ $\langle z_s \rangle$ $\langle x_1 \rangle$ $\langle x_2 \rangle$ $\langle y_1 \rangle$ $\langle y_2 \rangle$
: Randomly distribute n point sources at the source redshift z_s in a rectangular region defined by $x_1 < x < x_2$ and $y_1 < y < y_2$, and compute their lensed images. The result is written to *prefix*`_mock.dat`. The format of the file is

```
# n zs area
n_img[1] zs[1] xs[1] ys[1]
x1[1] y1[1] mu1[1] dt1[1]
....
xn[1] yn[1] mun[1] dtn[1]
n_img[2] zs[2] xs[2] ys[2]
x1[2] y1[2] mu1[2] dt1[2]
....
```

`mock2` $\langle n \rangle$ $\langle z_s \rangle$ $\langle r \rangle$ $\langle x_0 \rangle$ $\langle y_0 \rangle$
: Same as `mock1`, but this time the region is defined by a circle with a center (x_0, y_0) and a radius r.

`mock3` ⟨n⟩ ⟨z_s⟩ ⟨$f_{\rm rec}$⟩
: Same as `mock1`, but the rectangular region is defined by $f_{\rm rec}$ times the extent of the caustics.

`mockline` ⟨n⟩ ⟨z_s⟩ ⟨x_1⟩ ⟨x_2⟩ ⟨y_1⟩ ⟨y_2⟩ ⟨$flag\_out$⟩
: Compute lensing properties for equally spaced n point sources (redshift z_s) on a line segment between (x_1, y_1) and (x_2, y_2). The result is written to *prefix*`_mockline.dat`. If $flag\_out$ is set to 0 only the number of images and the total magnification are recorded. Otherwise for each source all the image properties are recorded with the format similar to `mock1`.

5.1.4 Composite sources

`point_flux` ⟨$fluxfile.dat$⟩
: Read a file that defines fluxes for point sources. The format of the file should be, the first line is an integer indicating the number of point sources, and second to last lines should list flux values for individual point sources. The value of the flux is either the peak count Σ_0 or the total flux $f_{\rm tot}$, depending on the parameter `flag_extnorm`.

`writeimageall` ⟨sky⟩ ⟨$noise$⟩
: Write images of both lensed point and extended sources to *prefix*`_image.fits`. See `writeimage` for meaning of *sky* and *noise*. Fluxes of point sources should be defined by `point_flux` before running this command. Images of point sources are defined by the PSF, so `flag_seeing` must be =1 and the seeing parameters should be defined appropriately.

`writeimageall_ori` ⟨sky⟩ ⟨$noise$⟩
: Same as `writeimageall`, but *unlensed* (i.e., original) images are shown. The output file is *prefix*`_source.fits`.

5.2 Model optimization

5.2.1 Preparation

`readobs_extend` ⟨$obsfile.fits$⟩ ⟨$maskfile.fits$⟩
: Read an image of lensed arcs with the filename $obsfile.fits$. The size of the image has to be consistent with that determined from `pix_ext` and `xmin`/`xmax`/`ymin`/`ymax`. You may also include a mask file ($maskfile.fits$); pixels with values in the mask file = 0 are used for optimization, and those with the mask value > 0 are ignored.

`readnoise_extend` ⟨$noisefile.fits$⟩
: Read a fits fine with the filename $noisefile.fits$ that defines noise values for each pixel. The size of the image has to be consistent with that determined from `pix_ext` and `xmin`/`xmax`/`ymin`/`ymax`.

`writenoise`
Calculate noises from the observed image read by `readobs_extend`, and write them to *prefix*`_obsnoise.fits`.

`addnoise` ⟨*sky*⟩ ⟨*noise*⟩
Add mean sky and 1σ sky noise indicated by *sky* and *noise*, respectively, to the observed image. If *noise* < 0, noises for individual pixels are computed from the sum of the counts from the objects and sky. This command may be useful for simulating real images. The output will be written to *prefix*`_addnoise.fits`.

`readobs_point` ⟨*obsfile.dat*⟩
Read a data file *obsfile.dat* for point source optimization. The format of the file is

```
 ID n_img zs err_zs
x[1] y[1] f[1] err_xy[1] err_f[1] dt[1] err_dt[1] parity[1]
....
x[n] y[n] f[n] err_xy[n] err_f[n] dt[n] err_dt[n] parity[n]
```

Here `ID` is the point source ID, `n_img` is the total number of images for this source. `zs` is the source redshift z_s and `err_zs` is 1σ error on z_s, both of which are optional (i.e., you do not need to specify any values here if you do not need any prior on z_s). `x`, `y`, and `f` are xy-coordinates and relative flux of each image (`f` should be replaced with the magnitude difference if `chi2_usemag`=1), `dt` is time delay. The code assumes Gaussian errors for all these parameters with the standard deviation indicated by `err`. If `err`=0.0, the corresponding parameter is not used for model optimizations (e.g., if you do not have any time delay measurements, just set `err_dt`=0.0 for all images). `parity` is an integer, if it is set to a positive (negative) integer, the image is forced to have a positive (negative) parity during optimization. If you do not want any parity constraint, just set `parity`=0.

`parprior` ⟨*priorfile.dat*⟩
Read a text file *priorfile.dat* which lists priors on parameters. In the file you can specify various types of priors, which I will explain in turn.

— `range`: This indicates the desired range of a parameter. Optimization will be performed only in this range. Examples are

```
range lens 2 5 0.1 0.5
range omega 0.2 0.4
```

The first line means that `p[5]` of the 2-nd lens model is restricted in the range between 0.1 and 0.5. The second line simply restricts the range of Ω_M between 0.2 and 0.4.

— `gauss`: Add a Gaussian prior to a parameter when optimizing the model. Again, examples are

```
gauss extend 1 6 -10.0 20.0
gauss hubble 0.72 0.04
```

In the first line `p[6]` of 1-st extended source is constrained to -10.0 ± 20.0. In the second example the Hubble constant is assumed to $h = 0.72 \pm 0.04$. If negative values are indicated for errors, the code interprets their absolute values as errors on $\log_{10}$ of the parameters (i.e., uses a long-normal distribution for priors).

— `match`: This is used to relate one parameter to another, and is currently available for `lens`, `point` (redshifts only), `extend`, and `psf`. The cross-match between `point` and `lens` is supported by `poilens`, and `extend` and `lens` is supported by `extlens`. Here are some examples:

```
match lens 2 4 1 4 1.0 0.0
match lens 2 1 1 4 0.7 5.0
match extlens 2 3 1 2 1.0 0.0
```

In the first example, `p[4]` of 2-nd lens model is forced to match `p[4]` of 1-st lens model (with a ratio of 1.0 and zero tolerance). In the second example, a combination something like `p[1]`(2-nd lens)$-0.7\times$`p[3]`(1-st lens) is assumed to have a Gaussian prior with an error of 5.0. In the third example, `p[3]` of 2-nd extended source is forced to match `p[2]` of 1-st lens model. For PSF parameters, it should be something like

```
match psf 1 5 1.0 0.0
```

In this example, the FWHMs of two Moffat components are forced to match.

— `obsext`: This is for optimization of extended sources. Since it is hard to know from observed images which part of arcs correspond to which sources, `obsext` tells the (approximated) correspondence between points in the image plane and extended source IDs. An example is

```
obsext 1 -5.1 20.1
```

This tells the code that a arc at around $(-5.1, 20.1)$ is associated with 1-st extended source.

`mapprior` ⟨*mappriorfile.dat*⟩

Read a text file *mappriorfile.dat* which lists priors on map values. The format of the file is

```
name[1] zs[1] x[1] y[1] ave[1] sig[1]
name[2] zs[2] x[2] y[2] ave[2] sig[2]
....
```

Each line contains one constraint. `name` indicates which map value you want to constrain: Available names include `kappa` (κ), `mag` ($|\mu|$, i.e., no sign), `gamma1` (γ_1), `gamma2` (γ_2), `g1` ($\gamma_1/(1-\kappa)$), and `g2` ($\gamma_2/(1-\kappa)$). Map values are computed at the position (`x[i]`, `y[i]`) with the source redshift `zs[i]`, and are constrained by Gaussian priors with `ave[i]`$\pm$`sig[i]`.

5.2.2 Optimization

`optimize` ⟨*extend/point*⟩

Perform model optimization, using a downhill simplex method (Press et al. 1992) to look for minimum χ^2. By default (without any argument) both extended and point sources are considered. If the argument is specified, the code considers extended/point sources only. By setting `chi2_restart` you can automatically restart optimization from the best-fit model, which provides an easy (but not so robust) check of the convergence of the result. The result will be written to *prefix*`_optresult.dat`.

`optextend`

Perform optimization only for parameters of extended sources (except for source redshifts which can only be fitted with `optimize`). The result will be written to *prefix*`_optresult_extend.dat`.

`optpoint`

Perform optimization only for parameters of point sources (except for source redshifts which can only be fitted with `optimize`). The result will be written to *prefix*`_optresult_point.dat`.

`c2calc`

Calculate χ^2 value for the current parameter set without any optimization and display the value.

`randomize`

All the parameters varied during optimization are randomized in the range defined by `parprior`. This can also be used to check the convergence of optimization. Please remember to set reasonable parameter ranges for all parameters by `parprior` before running this command!

`opt_explore` ⟨n⟩ ⟨χ^2 *limit*⟩ ⟨*extend/point*⟩

Repeat `optimize` and `randomize` n times for finding global χ^2 minimum. Fitted models will be written to *prefix*`_explore.dat` if χ^2 is smaller than χ^2 *limit* which can be indicated in the argument.

`varyone` ⟨i⟩ ⟨j⟩ ⟨*pmin*⟩ ⟨*pmax*⟩ ⟨n⟩ ⟨*extend/point*⟩

Calculate one-dimensional χ^2 slice. The parameter `p[j]` of i-th lens model is change from *pmin* to *pmax* with an interval of $((pmax - pmin)/n)$. A logarithmic spacing is adopted if $n < 0$. For each point the other parameters are optimized, and the resulting χ^2 is written to *prefix*`_vary.dat`.

`varytwo` ⟨$i1$⟩ ⟨$j1$⟩ ⟨$pmin1$⟩ ⟨$pmax1$⟩ ⟨$n1$⟩ ⟨$i2$⟩ ⟨$j2$⟩ ⟨$pmin2$⟩ ⟨$pmax2$⟩ ⟨$n2$⟩ ⟨*extend/point*⟩

Same as `varyone`, but two-dimensional χ^2 surface is explored.

`varyzs_extend` ⟨i⟩ ⟨*pmin*⟩ ⟨*pmax*⟩ ⟨n⟩ ⟨*extend/point*⟩

Same as `varyone`, but the source redshift of i-th extended source is varied.

`varyzs_point` ⟨*i*⟩ ⟨*pmin*⟩ ⟨*pmax*⟩ ⟨*n*⟩ ⟨*extend/point*⟩
Same as `varyone`, but the source redshift of *i*-th point source is varied.

`varycosmo` ⟨*name*⟩ ⟨*pmin*⟩ ⟨*pmax*⟩ ⟨*n*⟩ ⟨*extend/point*⟩
Same as `varyone`, but a cosmological parameter is varied. The name of parameter (`omega`, `lambda`, `weos`, or `hubble`) should be specified at *name*.

5.2.3 Markov-Chain Monte-Carlo (MCMC)

`mcmc_sigma` ⟨*sigfile.dat*⟩
Read a file that lists σ values for model parameters. The format of the file should be, the first line is an integer indicating the number of parameters, and second to last lines should list Gaussian σ of proposal distributions for individual parameters. The order matters; the correct order of MCMC parameters is, lens model parameters, cosmological parameters, parameters for extended sources, the sky value for extended source fitting, source redshifts for point sources, and PSF parameters for extended source fitting. If a negative value is indicated, a log-normal distribution (with $|\sigma| = \sigma(\log_{10} par)$) is used as a proposal distribution.

`mcmc` ⟨*n*⟩ ⟨*extend/point*⟩
Perform MCMC calculations. Before running this command σ values should be defined by `mcmc_sigma`. χ^2 is computed n times, and result is written to *prefix*`_mcmc.dat`.

`mcmc_kaprad` ⟨*file_mcmc.dat*⟩ ⟨*npar*⟩ ⟨z_s⟩ ⟨*x*⟩ ⟨*y*⟩ ⟨r_1⟩ ⟨r_2⟩ ⟨*n*⟩ ⟨*i*⟩
Read a resulting chain file of `mcmc` (with *npar* MCMC parameters), and for each chain circular-average convergences κ are computed. See `kapparad` for meaning of the other arguments. The output file is *prefix*`_mcmc_kaprad.dat`.

`mcmc_ein` ⟨*file_mcmc.dat*⟩ ⟨*npar*⟩ ⟨z_s⟩ ⟨*i*⟩
Read a rusting chain file of `mcmc` (with *npar* MCMC parameters), and for each chain the Einstein radius (for *i*-th lens model only, if specified) for the source redshift z_s is calculated. The output file is *prefix*`_mcmc_ein.dat`.

`mcmc_ein2` ⟨*file_mcmc.dat*⟩ ⟨*npar*⟩ ⟨z_s⟩ ⟨*x*⟩ ⟨*y*⟩ ⟨*i*⟩
Similar to `mcmc_ein`, but the Einstein radius is computed from the definition $\bar{\kappa}(< r_{\rm Ein}) = 1$ with the center (x, y).

`mcmc_calcim` ⟨*file_mcmc.dat*⟩ ⟨*npar*⟩ ⟨z_s⟩ ⟨*x*⟩ ⟨*y*⟩
Read a resulting chain file of `mcmc` (with *npar* MCMC parameters), and for each chain various image properties for the source at redshift z_s located at (x, y) are derived. The output file is *prefix*`_mcmc_calcim.dat`. Specifically for each chain μ, κ, γ_1, γ_2, γ, ϕ_x, ϕ_y, ω are computed and written to the file.

5.3 Utilities

`reset_par` ⟨*parname*⟩ ⟨*value*⟩
Change a value of any primary or secondary parameter *parname* (see Chapter 3) to

value.

`reset_lens` ⟨*i*⟩ ⟨*j*⟩ ⟨*value*⟩
Change a parameter value of *i*-th lens model, i.e., `p[j]`=*value*.

`reset_extend` ⟨*i*⟩ ⟨*j*⟩ ⟨*value*⟩
Change a parameter value of *i*-th extended source model, i.e., `p[j]`=*value*.

`reset_point` ⟨*i*⟩ ⟨*j*⟩ ⟨*value*⟩
Change a parameter value of *i*-th point source model, i.e., `p[j]`=*value*.

`reset_psf` ⟨*j*⟩ ⟨*value*⟩
Change a parameter value of the PSF model, i.e., `p[j]`=*value*.

`resetopt_lens` ⟨*i*⟩ ⟨*j*⟩ ⟨*value*⟩
Change a optimization flag for *j*-th parameter of *i*-th lens model. *value* should be 0 or 1.

`resetopt_extend` ⟨*i*⟩ ⟨*j*⟩ ⟨*value*⟩
Change a optimization flag for *j*-th parameter of *i*-th extended source model. *value* should be 0 or 1.

`resetopt_point` ⟨*i*⟩ ⟨*j*⟩ ⟨*value*⟩
Change a optimization flag for *j*-th parameter of *i*-th point source model. *value* should be 0 or 1.

`resetopt_psf` ⟨*j*⟩ ⟨*value*⟩
Change a optimization flag for *j*-th parameter of the PSF model. *value* should be 0 or 1.

`mv_lens` ⟨*i*⟩ ⟨*x*⟩ ⟨*y*⟩
Move the position of *i*-th lens model to (x, y).

`mv_extend` ⟨*i*⟩ ⟨*x*⟩ ⟨*y*⟩
Move the position of *i*-th extended source model to (x, y).

`mv_point` ⟨*i*⟩ ⟨*x*⟩ ⟨*y*⟩
Move the position of *i*-th point source model to (x, y).

`reset_obs_point` ⟨*i*⟩ ⟨*j*⟩ ⟨*k*⟩ ⟨*value*⟩
Change *k*-th parameter value of *j*-th multiple images of *i*-th point source that is originally specified by *readobs_point* command.

`printmodel`
Print current model parameters to display.

`printopt`
Print current optimization flags to display.

`printlensplane`
: Print lens planes to display.

`arcsec2mpc` $\langle z \rangle$ $\langle theta \rangle$
: Convert *theta* arcsec to h^{-1}Mpc (physical) for a redshift z.

`mpc2arcsec` $\langle z \rangle$ $\langle x \rangle$
: Convert x h^{-1}Mpc (physical) to arcsec for a redshift z.

`critdens` $\langle z_l \rangle$ $\langle z_s \rangle$
: Compute a physical (i.e., not comoving) critical surface mass density $\Sigma_{\rm crit}$ for the lens redshift z_l and the source redshift z_s.

`dismod` $\langle z \rangle$
: Compute a distance modulus for a redshift z.

`quit`
: End the program.

Chapter 6

More information

I'm showing some examples on the web.

Appendix A

Computing Lensing Properties

A.1 Circular Mass Distribution

When a mass profile has circular symmetry, the lens equation reduces to the one-dimensional form, written as a function of a radius $r = \sqrt{x^2 + y^2}$ from the center of the mass model. In general we define a mass model by the surface density profile $\kappa(r)$:

$$\kappa(r) = \frac{\Sigma(r)}{\Sigma_{\rm crit}}, \tag{A.1}$$

$$\Sigma_{\rm crit} = \frac{c^2}{4\pi G}\frac{D_{\rm os}}{D_{\rm ol}D_{\rm ls}}, \tag{A.2}$$

where $\Sigma(r)$ is projected surface mass density. We can then compute the deflection angle α and the lens potential $\phi(r)$ as follows:

$$\alpha(r) = r\bar{\kappa}(<r) = \frac{2}{r}\int_0^r r'\kappa(r')dr', \tag{A.3}$$

$$\phi(r) = \int_0^r \alpha(r')dr' = 2\int_0^r \kappa(r')r'\ln\left(\frac{r}{r'}\right)dr'. \tag{A.4}$$

The derivative of the defection angle, which is important for calculating shear and magnification, is obtained as

$$\alpha'(r) = 2\kappa(r) - \frac{\alpha(r)}{r} = 2\kappa(r) - \bar{\kappa}(<r). \tag{A.5}$$

For circular symmetric lenses, shear γ is always tangentially-aligned. Its size is given by

$$\gamma(r) = \bar{\kappa}(<r) - \kappa(r) = \kappa(r) - \alpha'(r). \tag{A.6}$$

These are related to the first derivatives of the lens potential as

$$\phi_x = \alpha(r)\frac{x}{r}, \tag{A.7}$$

$$\phi_y = \alpha(r)\frac{y}{r}, \tag{A.8}$$

and second derivatives as

$$\phi_{xx} = \alpha'(r)\frac{x^2}{r^2} + \alpha(r)\frac{y^2}{r^3} = 2\kappa(r)\frac{x^2}{r^2} - \alpha(r)\frac{x^2-y^2}{r^3}, \quad (A.9)$$

$$\phi_{yy} = \alpha'(r)\frac{y^2}{r^2} + \alpha(r)\frac{x^2}{r^3} = 2\kappa(r)\frac{y^2}{r^2} + \alpha(r)\frac{x^2-y^2}{r^3}, \quad (A.10)$$

$$\phi_{xy} = \alpha'(r)\frac{xy}{r^2} - \alpha(r)\frac{xy}{r^3} = 2\left[\kappa(r) - \frac{\alpha(r)}{r}\right]\frac{xy}{r^2}. \quad (A.11)$$

Therefore, we have

$$\frac{1}{2}(\phi_{xx} + \phi_{yy}) = \kappa(r), \quad (A.12)$$

$$\frac{1}{2}(\phi_{xx} - \phi_{yy}) = -\gamma(r)\cos 2\theta = \gamma_1, \quad (A.13)$$

$$\phi_{xy} = -\gamma(r)\sin 2\theta = \gamma_2. \quad (A.14)$$

It is sometimes useful to rewrite these lensing properties in a dimensionless form using a new variable $u \equiv r/r_0$:

$$\kappa(r) = b_{\rm norm}\kappa_{\rm dl}(u), \quad (A.15)$$

$$\alpha(r) = r_0 b_{\rm norm}\alpha_{\rm dl}(u), \quad (A.16)$$

$$\phi(r) = r_0^2 b_{\rm norm}\phi_{\rm dl}(u), \quad (A.17)$$

where $b_{\rm norm}$ is a dimensionless normalization factor and r_0 is a characteristic radius for the mass profile.

A.2 Elliptical Mass Distribution

The most natural (and observation-motivated) way to extend them to non-circular ones is to consider the elliptical surface mass density. This can be done by simply replacing r in $\kappa(r)$ by v which is defined as

$$\kappa(r): \quad r \to v \equiv \sqrt{\frac{\tilde{x}^2}{(1-e)} + (1-e)\tilde{y}^2}, \quad (A.18)$$

where e is an ellipticity (the axis ratio is $1-e$), and $\tilde{x}$ and $\tilde{y}$ are defined by

$$\tilde{x} = x\cos\theta_e + y\sin\theta_e, \quad (A.19)$$

$$\tilde{y} = -x\sin\theta_e + y\cos\theta_e. \quad (A.20)$$

In some case the lensing properties can be computed analytically, but in most case we have to resort to numerical integrations. I adopt the method of Schramm (1990,see also Keeton 2001b) which allows us to calculate lensing properties as a set of one-dimensional integrals. I

first rewrite the convergence as a function of $\xi \equiv \sqrt{\tilde{x}^2/q^2 + \tilde{y}^2}$ with $q \equiv 1-e$, i.e., $\kappa = \kappa(\xi)$. Then the lens potential and its derivatives are

$$\begin{aligned}
\phi &= \frac{q}{2} I(\tilde{x},\, \tilde{y}), && \text{(A.21)}\\
\phi_{\tilde{x}} &= q\tilde{x} J_1(\tilde{x},\, \tilde{y}), && \text{(A.22)}\\
\phi_{\tilde{y}} &= q\tilde{y} J_0(\tilde{x},\, \tilde{y}), && \text{(A.23)}\\
\phi_{\tilde{x}\tilde{x}} &= 2q\tilde{x}^2 K_2(\tilde{x},\, \tilde{y}) + q J_1(\tilde{x},\, \tilde{y}), && \text{(A.24)}\\
\phi_{\tilde{y}\tilde{y}} &= 2q\tilde{y}^2 K_0(\tilde{x},\, \tilde{y}) + q J_0(\tilde{x},\, \tilde{y}), && \text{(A.25)}\\
\phi_{\tilde{x}\tilde{y}} &= 2q\tilde{x}\tilde{y} K_1(\tilde{x},\, \tilde{y}), && \text{(A.26)}
\end{aligned}$$

where the integrals are

$$\begin{aligned}
I(x,\, y) &= \int_0^1 \frac{\zeta\alpha(\zeta)}{u\left[1-(1-q^2)u\right]^{1/2}} du, && \text{(A.27)}\\
J_n(x,\, y) &= \int_0^1 \frac{\kappa(\zeta)}{\left[1-(1-q^2)u\right]^{n+1/2}} du, && \text{(A.28)}\\
K_n(x,\, y) &= \int_0^1 \frac{u\kappa'(\zeta)}{\left[1-(1-q^2)u\right]^{n+1/2}} \frac{1}{2\zeta} du. && \text{(A.29)}
\end{aligned}$$

Here the variable ζ is defined by

$$\zeta \equiv \sqrt{u\left(\frac{x^2}{1-(1-q^2)u} + y^2\right)}. \qquad \text{(A.30)}$$

From the definitions of $\tilde{x}$ and $\tilde{y}$, one can easily derive the derivatives in the original coordinates:

$$\begin{aligned}
\phi_x &= \phi_{\tilde{x}} \cos\theta_e - \phi_{\tilde{y}} \sin\theta_e, && \text{(A.31)}\\
\phi_y &= \phi_{\tilde{x}} \sin\theta_e + \phi_{\tilde{y}} \cos\theta_e, && \text{(A.32)}\\
\phi_{xx} &= \phi_{\tilde{x}\tilde{x}} \cos^2\theta_e - 2\phi_{\tilde{x}\tilde{y}} \sin\theta_e \cos\theta_e + \phi_{\tilde{y}\tilde{y}} \sin^2\theta_e, && \text{(A.33)}\\
\phi_{yy} &= \phi_{\tilde{x}\tilde{x}} \sin^2\theta_e + 2\phi_{\tilde{x}\tilde{y}} \sin\theta_e \cos\theta_e + \phi_{\tilde{y}\tilde{y}} \cos^2\theta_e, && \text{(A.34)}\\
\phi_{xy} &= \phi_{\tilde{x}\tilde{x}} \sin\theta_e \cos\theta_e + \phi_{\tilde{x}\tilde{y}} (\cos^2\theta_e - \sin^2\theta_e) - \phi_{\tilde{y}\tilde{y}} \sin\theta_e \cos\theta_e. && \text{(A.35)}
\end{aligned}$$

A.3 Elliptical Potential

Another simple way to introduce non-circularity in the projected mass distribution is to make the lens potential (eq. [A.4]) elliptical. An advantage of this is that all the lensing properties can be described by simple arithmetic combinations of circular-symmetric quantities. However, when the ellipticity is large, it could result in "unphysical" surface mass distributions, such as concave isodensity contours and negative mass densities.

Specifically, I replace r in equation (A.4) by the following variable v:

$$\phi(r): \quad r \to v \equiv \sqrt{(1+e_p)\tilde{x}^2 + (1-e_p)\tilde{y}^2}, \qquad \text{(A.36)}$$

where $\tilde{x}$ and $\tilde{y}$ are same as those in equations (A.19) and (A.20). Note that the ellipticity of the potential, e_p, in general differs from the ellipticity of the projected mass distribution. In the standard situations the lens potential is rounder than the corresponding mass distribution. Calculations of the first and second derivatives of the lens potential are then quite straightforward:

$$\begin{aligned}
\phi_x &= \alpha(v)v_x, && \text{(A.37)}\\
\phi_y &= \alpha(v)v_y, && \text{(A.38)}\\
\phi_{xx} &= \alpha'(v)v_x^2 + \alpha(v)v_{xx}, && \text{(A.39)}\\
\phi_{xy} &= \alpha'(v)v_x v_y + \alpha(v)v_{xy}, && \text{(A.40)}\\
\phi_{yy} &= \alpha'(v)v_y^2 + \alpha(v)v_{yy}, && \text{(A.41)}
\end{aligned}$$

where $\alpha(u)$ and $\alpha'(u)$ are circular-symmetric deflection angle and its derivative. The derivatives of v are given by

$$\begin{aligned}
v_x &= \frac{x + e_p(x\cos 2\theta_e + y\sin 2\theta_e)}{v}, && \text{(A.42)}\\
v_y &= \frac{y + e_p(x\sin 2\theta_e - y\cos 2\theta_e)}{v}, && \text{(A.43)}\\
v_{xx} &= \frac{1 + e_p\cos 2\theta_e - v_x^2}{v}, && \text{(A.44)}\\
v_{yy} &= \frac{1 - e_p\cos 2\theta_e - v_y^2}{v}, && \text{(A.45)}\\
v_{xy} &= \frac{e_p\sin 2\theta_e - v_x v_y}{v}. && \text{(A.46)}
\end{aligned}$$

Appendix B

Catalog of Mass Models

B.1 External Perturbation (`pert`)

Objects near the lens sometimes affect the potential of the main lens. In case the perturbation is weak, one usually expands the effect as a Taylor series and keeps only low-order terms (e.g., Kochanek 1991). The (observable) lowest-order term is constant convergence κ and constant tidal shear γ. They are described by the following potential:

$$\begin{aligned} \phi &= \frac{1}{2}r^2\kappa + \frac{1}{2}r^2\gamma\cos 2(\theta - \theta_\gamma) && \text{(B.1)} \\ &= \frac{1}{2}(x^2+y^2)\kappa + \frac{1}{2}(x^2-y^2)\gamma_1 + xy\gamma_2, && \text{(B.2)} \end{aligned}$$

where

$$\begin{aligned} \gamma_1 &= \gamma\cos 2\theta_\gamma, && \text{(B.3)} \\ \gamma_2 &= \gamma\sin 2\theta_\gamma. && \text{(B.4)} \end{aligned}$$

The deflection angles are easily computed as

$$\begin{aligned} \phi_x &= x\kappa + x\gamma_1 + y\gamma_2, && \text{(B.5)} \\ \phi_y &= y\kappa - y\gamma_1 + x\gamma_2. && \text{(B.6)} \end{aligned}$$

The second derivatives,

$$\begin{aligned} \phi_{xx} &= \kappa + \gamma_1, && \text{(B.7)} \\ \phi_{yy} &= \kappa - \gamma_1, && \text{(B.8)} \\ \phi_{xy} &= \gamma_2, && \text{(B.9)} \end{aligned}$$

clearly illustrate that the lens potential induces constant κ and γ. These amplitudes are defined for a fiducial source redshift $z_{s,\mathrm{fid}}$; for sources at different redshifts, I simply scale them by $\propto D_{\mathrm{ls}}/D_{\mathrm{os}}$.

B.2 Third-Order Perturbation (`clus3`)

The third-order term of the Taylor series may be useful when the perturbation is rather strong. To suppress the number of parameters, I consider a "restricted" third-order perturbation defined by (e.g., Bernstein & Fischer 1999)

$$\phi = \frac{\delta}{4} r^3 \left[\sin(\theta - \theta_\delta) + \sin 3(\theta - \theta_\delta) \right]. \tag{B.10}$$

This is an exact third-order term for a singular isothermal sphere perturber, and will be reasonable for other perturbers. The derivatives are

$$\phi_x = \frac{\delta}{4} r \left[3x \left\{ \sin(\theta - \theta_\delta) + \sin 3(\theta - \theta_\delta) \right\} - y \left\{ \cos(\theta - \theta_\delta) + 3\cos 3(\theta - \theta_\delta) \right\} \right], \tag{B.11}$$

$$\phi_y = \frac{\delta}{4} r \left[3y \left\{ \sin(\theta - \theta_\delta) + \sin 3(\theta - \theta_\delta) \right\} + x \left\{ \cos(\theta - \theta_\delta) + 3\cos 3(\theta - \theta_\delta) \right\} \right], \tag{B.12}$$

$$\begin{aligned} \phi_{xx} = \frac{\delta}{4} \Bigg[& 3 \left(r + \frac{x^2}{r} \right) \left\{ \sin(\theta - \theta_\delta) + \sin 3(\theta - \theta_\delta) \right\} - 4 \frac{xy}{r} \left\{ \cos(\theta - \theta_\delta) + 3\cos 3(\theta - \theta_\delta) \right\} \\ & - \frac{y^2}{r} \left\{ \sin(\theta - \theta_\delta) + 9\sin 3(\theta - \theta_\delta) \right\} \Bigg], \end{aligned} \tag{B.13}$$

$$\begin{aligned} \phi_{yy} = \frac{\delta}{4} \Bigg[& 3 \left(r + \frac{y^2}{r} \right) \left\{ \sin(\theta - \theta_\delta) + \sin 3(\theta - \theta_\delta) \right\} + 4 \frac{xy}{r} \left\{ \cos(\theta - \theta_\delta) + 3\cos 3(\theta - \theta_\delta) \right\} \\ & - \frac{x^2}{r} \left\{ \sin(\theta - \theta_\delta) + 9\sin 3(\theta - \theta_\delta) \right\} \Bigg], \end{aligned} \tag{B.14}$$

$$\begin{aligned} \phi_{xy} = \frac{\delta}{4} \Bigg[& 3 \frac{xy}{r} \left\{ \sin(\theta - \theta_\delta) + \sin 3(\theta - \theta_\delta) \right\} + 2 \frac{x^2 - y^2}{r} \left\{ \cos(\theta - \theta_\delta) + 3\cos 3(\theta - \theta_\delta) \right\} \\ & + \frac{xy}{r} \left\{ \sin(\theta - \theta_\delta) + 9\sin 3(\theta - \theta_\delta) \right\} \Bigg]. \end{aligned} \tag{B.15}$$

B.3 Multipole Perturbation (`mpole`)

One can consider more general perturbations defined by

$$\phi = -\frac{\epsilon}{m} r^n \cos m(\theta - \theta_\epsilon - \pi/2). \tag{B.16}$$

For example, $m = 2$ and $n = 2$ describes an external shear. The derivatives are

$$\begin{aligned}
\phi_x &= -\frac{\epsilon}{m} r^{n-2} \left[nx \cos m(\theta - \theta_\epsilon - \pi/2) + my \sin m(\theta - \theta_\epsilon - \pi/2) \right], && \text{(B.17)} \\
\phi_y &= -\frac{\epsilon}{m} r^{n-2} \left[ny \cos m(\theta - \theta_\epsilon - \pi/2) - mx \sin m(\theta - \theta_\epsilon - \pi/2) \right], && \text{(B.18)} \\
\phi_{xx} &= -\frac{\epsilon}{m} r^{n-4} \left[(n-2)x \left\{ nx \cos m(\theta - \theta_\epsilon - \pi/2) + my \sin m(\theta - \theta_\epsilon - \pi/2) \right\} \right. \\
&\quad \left. + (nr^2 - m^2 y^2) \cos m(\theta - \theta_\epsilon - \pi/2) + nmxy \sin m(\theta - \theta_\epsilon - \pi/2) \right], && \text{(B.19)} \\
\phi_{yy} &= -\frac{\epsilon}{m} r^{n-4} \left[(n-2)y \left\{ ny \cos m(\theta - \theta_\epsilon - \pi/2) - mx \sin m(\theta - \theta_\epsilon - \pi/2) \right\} \right. \\
&\quad \left. + (nr^2 - m^2 x^2) \cos m(\theta - \theta_\epsilon - \pi/2) - nmxy \sin m(\theta - \theta_\epsilon - \pi/2) \right], && \text{(B.20)} \\
\phi_{xy} &= -\frac{\epsilon}{m} r^{n-4} \left[\left\{ n(n-2) + m^2 \right\} xy \cos m(\theta - \theta_\epsilon - \pi/2) \right. \\
&\quad \left. + m(n-1)(y^2 - x^2) \sin m(\theta - \theta_\epsilon - \pi/2) \right]. && \text{(B.21)}
\end{aligned}$$

B.4 Point mass (`point`)

The density profile of a point mass lens is described simply by δ-function, $\rho(\mathbf{x}) = M_{\rm tot}\delta(\mathbf{x})$. Obviously, no elliptical model is available for this model. The lens potential is computed as

$$\phi = r_{\rm Ein}^2 \ln r, \tag{B.22}$$

where $r_{\rm Ein}$ is the Einstein radius. In angular units it is give by

$$r_{\rm Ein} = \frac{1}{D_{\rm ol}} \sqrt{\frac{M_{\rm tot}}{\pi \Sigma_{\rm crit}}}. \tag{B.23}$$

The lensing properties can be computed easily (see §A.1).

B.5 Isothermal Ellipsoid (`sie`)

The Singular Isothermal Ellipsoid (SIE; the three-dimensional radial profile of $\rho \propto r^{-2}$) is used most frequently to model lensing galaxies. More generally, isothermal ellipsoids with finite core radii are useful in modeling galaxy-scale lenses. The convergence of the model is given by

$$\kappa = \frac{b_{\rm SIE}(q)}{2\sqrt{s^2(q) + \tilde{x}^2 + \tilde{y}^2/q^2}}, \tag{B.24}$$

where $q = 1 - e$. For a spherical case ($q = 1$) the normalization factor $b_{\rm SIE}$ is related to the galaxy velocity dispersion σ as

$$b_{\rm SIE}(1) = 4\pi \left(\frac{\sigma}{c}\right)^2 \frac{D_{\rm ls}}{D_{\rm os}}. \tag{B.25}$$

It agrees with the Einstein radius in the singular limit $r_{\rm core} \to 0$. Similarly, the parameter $s(q)$ should coincide with the core radius $r_{\rm core}$ for $q = 1$

$$s(1) = r_{\rm core}. \tag{B.26}$$

For non-spherical galaxies the choice of the normalization is not trivial. I adopt the following normalization:[1]

$$f(q) = \frac{b_{\rm SIE}(q)}{b_{\rm SIE}(1)} = \frac{s(q)}{s(1)} = \frac{1}{\sqrt{q}}. \tag{B.27}$$

This corresponds to choosing the dynamical normalization $\lambda(e) = 1$ in the notation of Chae (2003) and Oguri et al. (2008). The coordinates $\tilde{x}$ and $\tilde{y}$ are rotated by an angle θ_e:

$$\tilde{x} = x\sin\theta_e - y\cos\theta_e, \tag{B.28}$$

$$\tilde{y} = x\cos\theta_e + y\sin\theta_e. \tag{B.29}$$

An advantage of this model is that lensing properties can be described analytically (Kassiola & Kovner 1993; Kormann et al. 1994). The lens potential and its first derivative (in the rotated frame) become

$$\phi = \tilde{x}\phi_{\tilde{x}} + \tilde{y}\phi_{\tilde{y}} + b_{\rm SIE}(q)qs(q)\ln\left[\frac{(1+q)s(q)}{\sqrt{(\psi+s(q))^2+(1-q^2)\tilde{x}^2}}\right], \tag{B.30}$$

$$\phi_{\tilde{x}} = \frac{b_{\rm SIE}(q)q}{\sqrt{1-q^2}}\arctan\left[\frac{\sqrt{1-q^2}\tilde{x}}{\psi+s(q)}\right], \tag{B.31}$$

$$\phi_{\tilde{y}} = \frac{b_{\rm SIE}(q)q}{\sqrt{1-q^2}}\,\mathrm{arctanh}\left[\frac{\sqrt{1-q^2}\tilde{y}}{\psi+q^2s(q)}\right], \tag{B.32}$$

where ψ is defined by

$$\psi = \sqrt{q^2(s^2(q)+\tilde{x}^2)+\tilde{y}^2}. \tag{B.33}$$

Note that in the limit $q \to 1$ the deflection angle reduces to

$$\phi_{\tilde{x}} = \frac{b_{\rm SIE}(q)\tilde{x}}{\psi+s(q)}, \tag{B.34}$$

$$\phi_{\tilde{y}} = \frac{b_{\rm SIE}(q)\tilde{y}}{\psi+s(q)}. \tag{B.35}$$

The second derivatives are

$$\phi_{\tilde{x}\tilde{x}} = \frac{b_{\rm SIE}(q)q}{\psi}\frac{q^2s^2(q)+\tilde{y}^2+s(q)\psi}{(1+q^2)s^2(q)+2\psi s(q)+\tilde{x}^2+\tilde{y}^2}, \tag{B.36}$$

$$\phi_{\tilde{y}\tilde{y}} = \frac{b_{\rm SIE}(q)q}{\psi}\frac{s^2(q)+\tilde{x}^2+s(q)\psi}{(1+q^2)s^2(q)+2\psi s(q)+\tilde{x}^2+\tilde{y}^2}, \tag{B.37}$$

$$\phi_{\tilde{x}\tilde{y}} = -\frac{b_{\rm SIE}(q)q}{\psi}\frac{\tilde{x}\tilde{y}}{(1+q^2)s^2(q)+2\psi s(q)+\tilde{x}^2+\tilde{y}^2}. \tag{B.38}$$

Then the first and second derivative in the original frame can be derived as follows:

$$\phi_x = \phi_{\tilde{x}}\sin\theta_e + \phi_{\tilde{y}}\cos\theta_e, \tag{B.39}$$

$$\phi_y = -\phi_{\tilde{x}}\cos\theta_e + \phi_{\tilde{y}}\sin\theta_e, \tag{B.40}$$

$$\phi_{xx} = \phi_{\tilde{x}\tilde{x}}\sin^2\theta_e + 2\phi_{\tilde{x}\tilde{y}}\sin\theta_e\cos\theta_e + \cos^2\theta_e\phi_{\tilde{y}\tilde{y}}, \tag{B.41}$$

$$\phi_{yy} = \phi_{\tilde{x}\tilde{x}}\cos^2\theta_e - 2\phi_{\tilde{x}\tilde{y}}\sin\theta_e\cos\theta_e + \sin^2\theta_e\phi_{\tilde{y}\tilde{y}}, \tag{B.42}$$

$$\phi_{xy} = -\phi_{\tilde{x}\tilde{x}}\sin\theta_e\cos\theta_e + \phi_{\tilde{x}\tilde{y}}(\sin^2\theta_e - \cos^2\theta_e) + \phi_{\tilde{y}\tilde{y}}\sin\theta_e\cos\theta_e. \tag{B.43}$$

[1]Note that my normalization differs from the one used in *gravlens*. It adopts $f(q) = \sqrt{(1+q^2)/(2q^2)}$.

B.6 Pseudo-Jaffe Ellipsoid (`jaffe`)

The original Jaffe profile has the three-dimensional radial profile of $\rho \propto r^{-2}(r+r_{\rm trun})^{-2}$ (Jaffe 1983). For lensing purposes, it is useful to modify the profile to $\rho \propto (r^2+r_{\rm core}^2)^{-1}(r^2+r_{\rm trun}^2)^{-1}$ (Keeton 2001b), because the lensing properties can be expressed by a combination of those of isothermal ellipsoids (see §B.5). Specifically, we can define this model by the following convergence κ:

$$\kappa = \frac{b_{\rm SIE}(q)}{2}\left[\frac{1}{\sqrt{s^2(q)+\tilde{x}^2+\tilde{y}^2/q^2}} - \frac{1}{\sqrt{a^2(q)+\tilde{x}^2+\tilde{y}^2/q^2}}\right], \tag{B.44}$$

where $a(1) = r_{\rm trun}$ and $a(q)/a(1) = f(q)$. This is simply the difference of two convergences of isothermal ellipsoids; thus, the lensing properties of this model can be computed easily using those of isothermal ellipsoid shown in §B.5.

B.7 Power-Law Profile (`pow`/`powpot`)

The power-law model, $\rho(r) \propto r^{-\gamma}$ $(1 < \gamma < 3)$, is useful, say in studying the effect of the radial density slope on lensing. I adopt the Einstein radius of the lens (for a fiducial source redshift $z_{s,\rm fid}$) for the characteristic scale, $r_0 = r_{\rm Ein}$. The dimensionless lensing properties then become (see §A.1)

$$\phi_{\rm dl}(u) = \frac{u^{3-\gamma}}{3-\gamma}, \tag{B.45}$$

$$\alpha_{\rm dl}(u) = u^{2-\gamma}, \tag{B.46}$$

$$\kappa_{\rm dl}(u) = \frac{3-\gamma}{2}u^{1-\gamma}, \tag{B.47}$$

$$\kappa'_{\rm dl}(u) = \frac{(3-\gamma)(1-\gamma)}{2}u^{-\gamma}. \tag{B.48}$$

Note that $b_{\rm norm} = 1$. While in version 1 the ellipticity is introduced using methods described in §A.2 or §A.3, in version 2 the analytic calculation presented in Tessore & Metcalf (2015) is implemented and is used by default. This enables much faster calculations in the power-law elliptical density model.

B.8 Sérsic Profile (`sers`/`serspot`)

The Sérsic profile (Sérsic 1968) is widely used to describe light profiles of galaxies. The profile is defined in projection:

$$\Sigma(r) = \frac{M_{\rm tot}}{\pi r_s^2 \Gamma(2n+1)} \exp\left[-\left(\frac{r}{r_s}\right)^{1/n}\right], \tag{B.49}$$

where $M_{\rm tot}$ is the total mass, r_s is the scale radius, and $\Gamma(z)$ is the Gamma function defined by

$$\Gamma(z) = \int_0^\infty t^{z-1}e^{-t}dt. \tag{B.50}$$

The Sérsic index n controls the shape of the profile; the exponential disk is expressed by $n = 1$, whereas the de Vaucouleurs profile corresponds to $n = 4$. In the code, the half-light (half-mass, actually) radius r_e instead of r_s is used as a parameter. They are related with each other by

$$r_s = r_e b_n^{-n}, \tag{B.51}$$

where b_n is computed from

$$P(2n,\, b_n) = \frac{1}{2}, \tag{B.52}$$

with $P(z,\, x)$ being the regularized Gamma function:

$$P(z,\, x) = \frac{\gamma(z,\, x)}{\Gamma(z)} = \frac{1}{\Gamma(z)} \int_0^x t^{z-1} e^{-t} dt. \tag{B.53}$$

In practice I adopt fitting formulae of b_n. For $n > 0.36$ (Ciotti & Bertin 1999)

$$b_n = 2n - \frac{1}{3} + \frac{4}{405n} + \frac{46}{25515n^2} + \frac{131}{1148175n^3} - \frac{2194697}{30690717750n^4}, \tag{B.54}$$

and for $0.06 < n < 0.36$ (MacArthur et al. 2003)

$$b_n = 0.01945 - 0.8902n + 10.95n^2 - 19.67n^3 + 13.43n^4. \tag{B.55}$$

Note that the range of the Sérsic index is restricted to $0.06 < n < 20$ in the code.

To compute dimensionless lensing properties, I adopt

$$b_{\rm norm} = \frac{M_{\rm tot}}{\pi r_s^2 \Gamma(2n+1) \Sigma_{\rm crit}}, \tag{B.56}$$

$$r_0 = r_s. \tag{B.57}$$

Then I obtain (Cardone 2004)

$$\phi_{\rm dl}(u) = \frac{1}{2} u^2 {}_2F_2\left(2n,\, 2n;\, 2n+1,\, 2n+1;\, -u^{1/n}\right), \tag{B.58}$$

$$\alpha_{\rm dl}(u) = \Gamma(2n+1) u^{-1} P(2n,\, u^{1/n}), \tag{B.59}$$

$$\kappa_{\rm dl}(u) = \exp(-u^{1/n}), \tag{B.60}$$

$$\kappa'_{\rm dl}(u) = -\frac{1}{n} u^{1/n-1} \exp(-u^{1/n}). \tag{B.61}$$

The ellipticity is introduced using methods described in §A.2 or §A.3.

B.9 Hernquist Profile (`hern/ahern/hernpot`)

The Hernquist profile (Hernquist 1990) was proposed to model the light profile of an elliptical galaxy. The three-dimensional radial profile is

$$\rho(r) = \frac{M_{\rm tot}}{2\pi (r/r_b)(1 + r/r_b)^3}. \tag{B.62}$$

Note that the scale radius r_b is related to the effective (half-mass) radius R_e of a project surface mass density by

$$r_b = 0.551 R_e. \tag{B.63}$$

A natural choice of the dimensionless normalization and characteristic scale should be (see §A.1)

$$b_{\rm norm} = \frac{M_{\rm tot}}{2\pi r_b^2 \Sigma_{\rm crit}}, \tag{B.64}$$

$$r_0 = r_b. \tag{B.65}$$

In this case, the dimensionless lensing properties are (Keeton 2001b)

$$\phi_{\rm dl}(u) = 2F(u) + \ln\frac{x^2}{4}, \tag{B.66}$$

$$\alpha_{\rm dl}(u) = \frac{2u}{u^2-1}\left[1 - F(u)\right], \tag{B.67}$$

$$\kappa_{\rm dl}(u) = \frac{1}{(u^2-1)^2}\left[-3 + (2+u^2)F(u)\right], \tag{B.68}$$

$$\kappa'_{\rm dl}(u) = \frac{1}{u(u^2-1)^3}\left[2 + 13u^2 - 3u^2(u^2+4)F(u)\right], \tag{B.69}$$

where $F(u)$ is defined by

$$F(u) = \begin{cases} \dfrac{1}{\sqrt{1-u^2}} \operatorname{arctanh}\sqrt{1-u^2} & (u<1), \\ \dfrac{1}{\sqrt{u^2-1}} \arctan\sqrt{u^2-1} & (u>1). \end{cases} \tag{B.70}$$

The ellipticity is introduced using methods described in §A.2 or §A.3. `ahern` uses the fast approximated calculation of `hern` proposed by Oguri (2021).

B.10 NFW Profile (`nfw/anfw/nfwpot`)

The NFW density profile (Navarro et al. 1997) describes the structure of dark matter halos seen in N-body simulations. Te three-dimensional spherical density profile is defined by

$$\rho(r) = \frac{\rho_s}{(r/r_s)(1+r/r_s)^2}, \tag{B.71}$$

where r_s is the scale radius and ρ_s is the characteristic density. Instead of the scale radius, the concentration parameter c is used in most cases to describe the profile. It is defined by the ratio of the virial radius $r_{\rm vir}$ to the the scale radius:

$$c = \frac{r_{\rm vir}}{r_s}. \tag{B.72}$$

The scale radius and the characteristics density can be related to the virial mass M and the concentration parameter c using the following relation.

$$M = \frac{4\pi}{3} r_{\rm vir}^3 \Delta(z) \bar{\rho}(z) = \int_0^{r_{\rm vir}} \rho(r) 4\pi r^2 dr, \tag{B.73}$$

where $\Delta(z)$ is the nonlinear overdensity (see Appendix C). This leads to

$$r_s = \frac{r_{\rm vir}}{c} = \left[\frac{3M}{4\pi\Delta(z)\bar{\rho}(z)}\right]^{1/3} \frac{1}{c}, \tag{B.74}$$

$$\rho_s = \frac{\Delta(z)\bar{\rho}(z)c^3}{3m_{\rm nfw}(c)}, \tag{B.75}$$

$$m_{\rm nfw}(c) = \int_0^c \frac{r}{(1+r)^2} dr = \ln(1+c) - \frac{c}{1+c}. \tag{B.76}$$

Here $\bar{\rho}(z)$ denotes the mean matter density of the universe at redshift z.

Now let's consider the lensing properties. I choose the dimensionless normalization and characteristic scale as (see §A.1)

$$b_{\rm norm} = \frac{4\rho_s r_s}{\Sigma_{\rm crit}}, \tag{B.77}$$

$$r_0 = r_s. \tag{B.78}$$

Then lensing properties can be described as (Bartelmann 1996)

$$\phi_{\rm dl}(u) = \frac{1}{2}\left[\left(u^2-1\right)F^2(u) + \ln^2\left(\frac{u}{2}\right)\right], \tag{B.79}$$

$$\alpha_{\rm dl}(u) = \frac{1}{u}\left[F(u) + \ln\frac{u}{2}\right], \tag{B.80}$$

$$\kappa_{\rm dl}(u) = \frac{1}{2(u^2-1)}\left[1 - F(u)\right], \tag{B.81}$$

$$\kappa'_{\rm dl}(u) = \frac{1}{2u(u^2-1)^2}\left[-1 - 2u^2 + 3u^2 F(u)\right], \tag{B.82}$$

where $F(u)$ was defined in equation (B.70). The ellipticity is introduced using methods described in §A.2 or §A.3. `anfw` uses the fast approximated calculation of `nfw` proposed by Oguri (2021).

B.11 Generalized NFW Profile (`gnfw/gnfwpot`)

The controversy over the inner slope of the NFW profile motivated the following generalization of the radial profile (e.g., Jing & Suto 2000)

$$\rho(r) = \frac{\rho_s}{(r/r_s)^\alpha (1 + r/r_s)^{3-\alpha}}. \tag{B.83}$$

In this model the inner slope is parametrized by α ($0 < \alpha < 2$); the original NFW profile corresponds to $\alpha = 1$. Instead of the standard concentration parameter for the NFW profile, we adopt the following "modified" concentration parameter as a model parameter:

$$c_{-2} = \frac{r_{\rm vir}}{r_{-2}} = \frac{r_{\rm vir}}{(2-\alpha)r_s} = \frac{c}{2-\alpha}, \tag{B.84}$$

where r_{-2} indicates the radius where the radial slope becomes $d\ln\rho/d\ln r = -2$. The characteristic density is

$$\rho_s = \frac{\Delta(z)\bar{\rho}(z)c^3}{3m_{\rm gnfw}(c)}, \tag{B.85}$$

$$m_{\rm gnfw}(c) = \int_0^c \frac{r^{2-\alpha}}{(1+r)^{3-\alpha}} dr. \tag{B.86}$$

Using the same $b_{\rm norm}$ and r_0 as those for the NFW profile, the lensing properties can be described as

$$\phi_{\rm dl}(u) = \int_0^u \alpha_{\rm dl}(u')du', \tag{B.87}$$

$$\alpha_{\rm dl}(u) = \frac{1}{u}m_{\rm gnfw}(u) + u^{2-\alpha}\int_0^1 (z+u)^{\gamma-3}\frac{1-\sqrt{1-z^2}}{z}dz, \tag{B.88}$$

$$\kappa_{\rm dl}(u) = \frac{1}{2}\int_0^\infty \frac{1}{\left(\sqrt{u^2+z^2}\right)^\alpha\left(1+\sqrt{u^2+z^2}\right)^{3-\alpha}}dz, \tag{B.89}$$

$$\kappa'_{\rm dl}(u) = -\frac{u}{2}\int_0^\infty \frac{\alpha+3\sqrt{u^2+z^2}}{\left(\sqrt{u^2+z^2}\right)^{2+\alpha}\left(1+\sqrt{u^2+z^2}\right)^{4-\alpha}}dz. \tag{B.90}$$

The ellipticity is introduced using methods described in §A.2 or §A.3. Since no analytic expression is available even for the circular symmetric case, the elliptical density of the generalized NFW model requires multiple numerical integrals, making the computation quite slow. Thus I prepare `gnfw_usetab` flag – if this is on, the code first make tables of dimensionless deflection angles and convergences as a function of u and α, and compute lensing properties by interpolating the values in the tables.

B.12 Truncated NFW Profile (`tnfw/tnfwpot`)

A possible concern of the NFW profile (eq. [B.71]) is its outer radial slope of -3 which suggests that the total mass of the profile diverges. In practice one might expect real dark halos to be truncated due to tidal effects. Bearing this in mind, Baltz et al. (2009) proposed the truncated NFW profile of the following form:

$$\rho(r) = \frac{\rho_s}{(r/r_s)(1+r/r_s)^2}\left[\frac{1}{1+(r/r_t)^2}\right]^2, \tag{B.91}$$

where r_t denotes the tidal radius. The model should also be useful to model subhalos whose density profiles are well approximated by the NFW profile with the tidal truncation.

Instead of r_t, the code adopts the following dimensionless tidal radius in units of the virial radius as a model parameter:

$$t = \frac{r_t}{r_{\rm vir}}, \tag{B.92}$$

here we compute $r_{\rm vir}$ without the truncation, i.e., in the $r_t \to \infty$ limit. The use of untruncated $r_{\rm vir}$ (or M) is convenient because it assumes $\rho_{\rm tnfw}(r) \approx \rho_{\rm nfw}(r)$ at $r \ll r_t$, including overall normalizations.

An advantage of this truncated NFW profile is that lensing properties can be computed analytically. First we introduce a auxiliary parameter τ defined by

$$\tau = \frac{r_t}{r_s} = ct. \tag{B.93}$$

Adopting the same $b_{\rm norm}$ and r_0 as those used for the NFW profile (eqs. [B.77] and [B.78]), the lensing properties can be written as (Baltz et al. 2009)

$$\begin{aligned}
\phi_{\rm dl}(u) &= \frac{1}{2(\tau^2+1)^3}\Bigg\{\tau^3\pi\left[(3\tau^2-1)\ln(\tau+\sqrt{\tau^2+u^2}) - 4\tau\sqrt{\tau^2+u^2}\right] \\
&\quad +(3\tau^4-6\tau^2-1)\tau\sqrt{\tau^2+u^2}\,L(u) + \tau^4(\tau^2-3)\,L^2(u) \\
&\quad +8\tau^4(u^2-1)F(u) + \tau^4(\tau^2-3)(u^2-1)F^2(u) \\
&\quad +\tau^2\left[2\tau^2(\tau^2-3)\ln\tau - 3\tau^4 - 2\tau^2 + 1\right]\ln u \\
&\quad +\tau^2\Big[\tau^2(4\tau\pi + (\tau^2-3)\ln^2 2 + 8\ln 2) \\
&\quad -\ln(2\tau)(1+6\tau^2-3\tau^4+\tau^2(\tau^2-3)\ln(2\tau)+\tau\pi(3\tau^2-1))\Big]\Bigg\},
\end{aligned} \tag{B.94}$$

$$\begin{aligned}
\alpha_{\rm dl}(u) &= \frac{\tau^4}{2(\tau^2+1)^3 u}\Bigg\{2[\tau^2+1+4(u^2-1)]F(u) + \frac{1}{\tau}\left[\pi(3\tau^2-1) + 2\tau(\tau^2-3)\ln\tau\right] \\
&\quad +\frac{1}{\tau^3\sqrt{\tau^2+u^2}}\Big(-\tau^3\pi[4(\tau^2+u^2)-\tau^2-1] + \\
&\quad +[\tau^2(1-\tau^4)+(\tau^2+u^2)(3\tau^4-6\tau^2-1)]L(u)\Big)\Bigg\},
\end{aligned} \tag{B.95}$$

$$\begin{aligned}
\kappa_{\rm dl}(u) &= \frac{\tau^4}{4(\tau^2+1)^3}\Bigg\{\frac{2(\tau^2+1)}{u^2-1}\left[1-F(u)\right] + 8F(u) + \frac{\tau^4-1}{\tau^2(\tau^2+u^2)} - \frac{\pi[4(\tau^2+u^2)+\tau^2+1]}{(\tau^2+u^2)^{3/2}} \\
&\quad +\frac{\tau^2(\tau^4-1)+(\tau^2+u^2)(3\tau^4-6\tau^2-1)}{\tau^3(\tau^2+u^2)^{3/2}}\,L(u)\Bigg\},
\end{aligned} \tag{B.96}$$

$$\begin{aligned}\kappa'_{\rm dl}(u) &= \frac{\tau^4}{4(\tau^2+1)^3}\Bigg\{\frac{2(\tau^2+1)}{u(u^2-1)^2}\left[-1-2u^2+3u^2F(u)\right] \\ &+\frac{8}{u(u^2-1)}\left[1-u^2F(u)\right]-\frac{2u(\tau^4-1)}{\tau^2(\tau^2+u^2)^2}+\frac{\pi u[4(\tau^2+u^2)+3(\tau^2+1)]}{(\tau^2+u^2)^{5/2}} \\ &+\frac{\tau^2(\tau^4-1)+(\tau^2+u^2)(3\tau^4-6\tau^2-1)}{u\tau^2(\tau^2+u^2)^2} \\ &-\frac{3\tau^2(\tau^4-1)+(\tau^2+u^2)(3\tau^4-6\tau^2-1)}{\tau^3(\tau^2+u^2)^{5/2}}uL(u)\Bigg\}. \end{aligned} \tag{B.97}$$

Here $F(u)$ is from equation (B.70) and $L(u)$ is defined by

$$L(u)=\ln\left(\frac{\sqrt{\tau^2+u^2}-\tau}{u}\right). \tag{B.98}$$

The ellipticity is introduced using methods described in §A.2 or §A.3.

B.13 Einasto Profile (`ein/einpot`)

Recent very high-resolution N-body simulations suggest that the radial density profile of dark matter halos may be fitted better by the Einasto profile (e.g., Merritt et al. 2006; Gao et al. 2008). I consider the following form:

$$\rho(r)=\rho_s\exp\left[-\frac{2}{\alpha_{\rm E}}\left(\frac{r}{r_s}\right)^{\alpha_{\rm E}}\right]. \tag{B.99}$$

With this particular form, the density profile has the logarithmic slope of -2 at $r=r_s$. In N-body simulations a typical value of $\alpha_{\rm E}$ for dark matter halos is ≈ 0.2. In a manner analogous to an NFW profile, I define the concentration parameter as

$$c_{\rm E}=\frac{r_{\rm vir}}{r_s}. \tag{B.100}$$

Then ρ_s can be described as

$$\rho_s=\frac{\Delta(z)\bar{\rho}(z)c_{\rm E}^3}{3m_{\rm Ein}(c_{\rm E})}, \tag{B.101}$$

$$m_{\rm Ein}(x)=\frac{1}{\alpha_{\rm E}}\left(\frac{\alpha_{\rm E}}{2}\right)^{3/\alpha_{\rm E}}\gamma\left(\frac{3}{\alpha_{\rm E}},\frac{2}{\alpha_{\rm E}}x^{\alpha_{\rm E}}\right). \tag{B.102}$$

If I adopt the following normalization and characteristic scale

$$b_{\rm norm}=\frac{4\rho_s r_s}{\Sigma_{\rm crit}}, \tag{B.103}$$

$$r_0=r_s, \tag{B.104}$$

the lensing properties are

$$\phi_{\rm dl}(u)=\int_0^u\alpha_{\rm dl}(u')du', \tag{B.105}$$

$$\alpha_{\rm dl}(u) = \frac{1}{u}\int_0^\infty \frac{m_{\rm Ein}(u\sqrt{1+z^2}) + u\, m_{\rm Ein}(u\sqrt{1+z^2}/z)}{(1+z)^{3/2}} dz, \tag{B.106}$$

$$\kappa_{\rm dl}(u) = \frac{1}{2}\int_0^\infty \exp\left[-\frac{2}{\alpha_{\rm E}}\left(u^2+z^2\right)^{\alpha_{\rm E}/2}\right] dz, \tag{B.107}$$

$$\kappa'_{\rm dl}(u) = -\frac{u}{4}\int_0^\infty \left(u^2+z^2\right)^{\alpha_{\rm E}/2-1} \exp\left[-\frac{2}{\alpha_{\rm E}}\left(u^2+z^2\right)^{\alpha_{\rm E}/2}\right] dz. \tag{B.108}$$

See Keeton (2001b) for the derivation of equation (B.106). The ellipticity is introduced using methods described in §A.2 or §A.3.

B.14 Multiple galaxies (`gals`)

This lens model is mainly for modeling member galaxies in clusters. This model makes use of a list of galaxies with the positions, relative luminosities, ellipticities and position angles, and assumes that the lens parameters scales with the luminosity. Individual galaxies are modeled by the pseudo-Jaffe ellipsoid (see §B.6). The velocity dispersion and truncation radius for each galaxy is computed by

$$\frac{\sigma}{\sigma_*} = \left(\frac{L}{L_*}\right)^{1/4}, \tag{B.109}$$

$$\frac{r_{\rm trun}}{r_{\rm trun,\,*}} = \left(\frac{L}{L_*}\right)^{\eta}. \tag{B.110}$$

Note tat the mass-to-light ratio becomes constant if the parameter $\eta = 1/2$.

Appendix C

Halo Overdensity

The nonlinear overdensity $\Delta_{\rm vir}(z)$ is required to compute a "virial mass" for some profiles such as NFW. For a given cosmological model, it is usually calculated by using the so-called spherical collapse model. For $\Omega_M + \Omega_{\rm DE} = 1$ and $0 < \Omega_M < 1$, I adopt the following fitting formula (Nakamura & Suto 1997)

$$\Delta_{\rm vir}(z) \approx 18\pi^2 \left[1 + 0.40929 w(z)^{0.90524}\right], \tag{C.1}$$

where $w(z) \equiv 1/\Omega_M(z) - 1$. For the universe with $0 < \Omega_M < 1$ and $\Omega_{\rm DE} = 0$, I use

$$\Delta_{\rm vir}(z) = 4\pi^2 \frac{\left[\cosh \eta(z) - 1\right]^3}{\left[\sinh \eta(z) - \eta(z)\right]^2}, \tag{C.2}$$

where $\eta(z) \equiv \cosh^{-1}\left[2/\Omega_M(z) - 1\right]$. For the other cases I assume $\Delta_{\rm vir}(z) = 18\pi^2$. Therefore, for some non-standard cosmological models (e.g., models with $w_0 \neq -1$) the nonlinear overdensity differs from what the spherical collapse model predicts. One way to get around this problem is to set `flag_hodensity`=1 or =2 and use a fixed value of your choice for the nonlinear overdensity.

Appendix D

Catalog of Extended Source Models

For all models, the ellipticity is introduced by replacing r to v defined in equation (A.18).

D.1 Gaussian profile (`gauss`)

The surface brightness profile $\Sigma(r)$ of the Gaussian profile is simply given by

$$\Sigma(r) = \Sigma_0 \exp\left(-\frac{r^2}{2\sigma^2}\right). \tag{D.1}$$

The total flux is

$$f_{\rm tot} = 2\pi\sigma^2\Sigma_0. \tag{D.2}$$

The FWHM of the profile is

$$\mathrm{FWHM} = \sqrt{8\ln 2}\sigma \approx 2.3548\sigma. \tag{D.3}$$

D.2 Sérsic profile (`sersic`)

The Sérsic profile is one of the most popular profiles to study the galaxy morphology. It is defined as

$$\Sigma(r) = \Sigma_0 \exp\left[-b_n\left(\frac{r}{r_e}\right)^{1/n}\right]. \tag{D.4}$$

The radius r_e is the effective radius. The total flux is

$$f_{\rm tot} = \pi r_e^2 \Sigma_0 b_n^{-2n}\Gamma(2n+1). \tag{D.5}$$

For b_n, I adopt the approximations given by equations (B.54) and (B.55).

D.3 Tophat profile (`tophat`)

The tophat profile is in fact unrealistic for describing astronomical objects, but should useful for some theoretical investigations. It is given by

$$\Sigma(r) = \begin{cases} \Sigma_0 & (r < r_0), \\ 0 & (r > r_0). \end{cases} \tag{D.6}$$

The total flux is computed as

$$f_{\rm tot} = \pi r_0^2 \Sigma_0. \tag{D.7}$$

D.4 Moffat profile (`moffat`)

The Moffat profile (Moffat 1969) is sometimes used to model the PSF. It is given by

$$\Sigma(r) = \Sigma_0 \left[1 + \left(\frac{r}{\alpha}\right)^2\right]^{-\beta}. \tag{D.8}$$

The total flux is computed as

$$f_{\rm tot} = \pi \alpha^2 \Sigma_0 / (\beta - 1). \tag{D.9}$$

The FWHM of the profile is

$$\mathrm{FWHM} = 2\alpha\sqrt{2^{1/\beta} - 1}. \tag{D.10}$$

D.5 Jaffe profile (`jaffe`)

This is the same model as the one used for lens mass model, and is given by

$$\Sigma(r) = \frac{\Sigma_0}{1/r_{\rm core} - 1/r_{\rm trun}} \left[\frac{1}{\sqrt{r_{\rm core}^2 + r^2}} - \frac{1}{\sqrt{r_{\rm trun}^2 + r^2}}\right]. \tag{D.11}$$

The total flux is computed as

$$f_{\rm tot} = 2\pi r_{\rm core} r_{\rm trun} \Sigma_0. \tag{D.12}$$

D.6 Multiple sources (`srcs`)

One can define a set of many Sérsic sources by this. The flux, location, and shape of each source galaxy are defined in the input file. $f_{\rm norm}$ controls overall normalization for all source galaxies, i.e., the peak surface brightness for each source is replaced with $f_{\rm norm} \times \Sigma_0$.

References

Baltz, E. A., Marshall, P., & Oguri, M. 2009, JCAP, 1, 15

Bartelmann, M. 1996, A&A, 313, 697

Bernstein, G., & Fischer, P. 1999, AJ, 118, 14

Cardone, V. F. 2004, A&A, 415, 839

Chae, K.-H. 2003, MNRAS, 346, 746

Ciotti, L., & Bertin, G. 1999, A&A, 352, 447

Gao, L., Navarro, J. F., Cole, S., Frenk, C. S., White, S. D. M., Springel, V., Jenkins, A., & Neto, A. F. 2008, MNRAS, 387, 536

Hernquist, L. 1990, ApJ, 356, 359

Jaffe, W. 1983, MNRAS, 202, 995

Jing, Y. P., & Suto, Y. 2000, ApJ, 529, L69

Kassiola, A., & Kovner, I. 1993, ApJ, 417, 450

Keeton, C. R. 2001a, preprint (astro-ph/0102340)

Keeton, C. R. 2001b, preprint (astro-ph/0102341)

Kochanek, C. S. 1991, ApJ, 373, 354

Kormann, R., Schneider, P., & Bartelmann, M. 1994, A&A, 284, 285

MacArthur, L. A., Courteau, S., & Holtzman, J. A. 2003, ApJ, 582, 689

Merritt, D., Graham, A. W., Moore, B., Diemand, J., & Terzić, B. 2006, AJ, 132, 2685

Moffat, A. F. J. 1969, A&A, 3, 455

Nakamura, T. T., & Suto, Y. 1997, Prog. Theor. Phys., 97, 49

Navarro, J. F., Frenk, C. S., & White, S. D. M. 1997, ApJ, 490, 493

Oguri, M. 2010, PASJ, 62, 1017

Oguri, M. 2021, arXiv:2106.11464

Oguri, M., et al. 2008, AJ, 135, 512

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. 1992, *Numerical recipes in C.*, Cambridge: University Press

Schramm, T. 1990, A&A, 231, 19

Sérsic, J. L. 1968, Cordoba, Argentina: Observatorio Astronomico, 1968

Tessore, N., & Metcalf, R. B. 2015, A&A, 580, A79