Monthly Notices
of the
ROYAL ASTRONOMICAL SOCIETY

MNRAS **445,** 1942–1953 (2014) doi:10.1093/mnras/stu1859

GLAMER – I. A code for gravitational lensing simulations with adaptive mesh refinement

R. Benton Metcalf[1]★ and Margarita Petkova[1,2,3]

[1]*Dipartimento di Fisica e Astronomia, Università di Bologna, viale Berti Pichat 6/2, I-40127 Bologna, Italy*
[2]*Department of Physics, Ludwig-Maximilians-Universität, Scheinerstr. 1, D-81679 München, Germany*
[3]*Excellence Cluster Universe, Boltzmannstr. 2, D-85748 Garching, Germany*

Accepted 2014 September 7. Received 2014 August 16; in original form 2013 December 3

ABSTRACT

A computer code is described for the simulation of gravitational lensing data. The code incorporates adaptive mesh refinement in choosing which rays to shoot based on the requirements of the source size, location and surface brightness distribution or to find critical curves/caustics. A variety of source surface brightness models are implemented to represent galaxies and quasar emission regions. The lensing mass can be represented by point masses (stars), smoothed simulation particles, analytic halo models, pixelized mass maps or any combination of these. The deflection and beam distortions (convergence and shear) are calculated by modified tree algorithm when haloes, point masses or particles are used and by fast Fourier transform when mass maps are used. The combination of these methods allow for a very large dynamical range to be represented in a single simulation. Individual images of galaxies can be represented in a simulation that covers many square degrees. For an individual strongly lensed quasar, source sizes from the size of the quasar's host galaxy ($\sim$100 kpc) down to microlensing scales ($\sim 10^{-4}$ pc) can be probed in a self-consistent simulation. Descriptions of various tests of the code's accuracy are given.

Key words: gravitational lensing: strong – gravitational lensing: weak – gravitational lensing: micro – methods: numerical – cosmology: miscellaneous.

1 INTRODUCTION

Gravitational lensing is becoming a more and more powerful tool for investigating cosmology and astrophysics. It has been used to study such disparate subjects as dark energy, the cold dark matter model for structure formation, the formation and structure of galaxies, the structure of active galactic nuclei and the distribution of stars and planets in and around our own galaxy. In most cases, detailed comparisons between theoretical models and lensing data require computer simulations.

Here, we seek to develop a computer code, called GLAMER, flexible enough that it can be used to simulate many different types of lensing data include weak lensing on large scales from cosmological simulations, weak and strong lensing by galaxy clusters, galaxy–galaxy lensing, galaxy–quasar lensing, lensing by substructures in the lenses or somewhere along the line of sight and microlensing by stars in a lens galaxy. To achieve this, the code needs to be able to represent the sources and mass distribution within the lens in a very flexible and configurable way.

One of our goals is to study microlensing in the same simulation as strong lensing on galaxy scales. A galaxy acting as a strong lens for a Quasi-Stellar Object (QSO) might have a scale of $\sim$100 kpc, while the microlensing of the quasar by stars might take place on a scale of $\sim$0.01 kpc or smaller. From its host galaxy down to its X-ray emitting accretion disc, the QSO has emission regions on all scales that can be used to simultaneously constrain the lens and source model. A related goal is to simulate the images of galaxies in the same simulation as weak lensing across many square degrees. This would enable one to use different redshifts for each source and to represent the effects of lensing on all relevant scales simultaneously. Both these goals require a very large dynamical range in resolution for the ray-shooting.

Another problem we wish to address with GLAMER is that of the inner regions of strong lenses typically being dominated by baryons – stars and gas – which are not represented in the highest resolution *N*-body simulations. It is also true that lensing quantities such as the magnification are very sensitive to the numerical noise that comes from the discreteness of the simulation particles. Recently, Angulo et al. (2014) have made some progress on this latter problem. Our approach is to allow for a combination of smoothed particles and analytic profiles to exist within the simulation.

We also wanted to make the code flexible enough to do many different lensing applications in a relatively user friendly way. To this end our code allows for a variety of source types – circular,

★ E-mail: robertbenton.metcalf@unibo.it

analytic galaxy models, pixelized images, multiple sources, QSO emission regions, etc. – which can be easily added to. It also allows for a variety of lenses – analytic haloes, analytic galaxies, *N*-body particles, stars, smooth shear fields – which can also be easily added to. The code also has the ability to do multiplane lensing although this capability will be discussed in a companion paper (Petkova, Metcalf & Giocoli 2014). Here, we will concentrate on what has been implemented to allow this flexibility and dynamic range.

There is an extensive literature on simulating gravitational lensing (for a very sparse sample; Wambsganss, Cen & Ostriker 1998; Fluke, Webster & Mortlock 1999; Jain, Seljak & White 2000; Hamana & Mellier 2001; Vale & White 2003; Amara et al. 2006; Mediavilla et al. 2006; Aubert, Amara & Metcalf 2007; Hilbert et al. 2007, 2009; Pace et al. 2007; Meneghetti et al. 2008; Sato et al. 2009; Poindexter & Kochanek 2010; Vegetti et al. 2010; Takahashi et al. 2011; Killedar et al. 2012; Angulo et al. 2014). GLAMER makes significant advances beyond previous codes in its dynamical range, adaptability to a wide variety of applications and ease of use. We will refer to some previous work as it becomes relevant to the discussion.

In Section 2, the basic problem that the code is required to solve is described. The calculation of the deflection angle is discussed in Section 3. The methods used for adaptive mesh refinement (AMR), image finding and caustic finding are discussed in Section 4. In Section 5, various tests of the code are presented and the paper is concluded in Section 6.

2 LENSING SIMULATIONS

The job of a ray-tracing code can be broken up into two broad tasks. One is to calculate the deflection angle which will be discussed in Section 3 and the other is to find and map the images which will be discussed in Section 4. At the heart of these tasks is the lensing equation which relates a point on the source plane, $\boldsymbol{y}$, to its image point, $\boldsymbol{x}$, on the image plane:

$$\boldsymbol{y} = \boldsymbol{x} - \boldsymbol{\alpha}(\boldsymbol{x}). \tag{1}$$

In this paper, we will restrict ourselves to a single lens plane. The more complicated case of multiple lens planes will be discussed in a companion paper (Petkova et al. 2014). For a single lens plane, the coordinates can be related to points on the sky by

$$\boldsymbol{x} = \mathcal{D}_\mathrm{l}\boldsymbol{\theta},\ \boldsymbol{y} = \mathcal{D}_\mathrm{l}\boldsymbol{\beta}, \tag{2}$$

where $\mathcal{D}_\mathrm{l}$ is a reference distance which will usually be taken to be the angular size distance to the lens in the case of a single lens plane. The deflection angle, $\boldsymbol{\alpha}(\boldsymbol{x})$, (which has units of length in this form) can be related to the true deflection angle in the path of a light-ray, $\tilde{\boldsymbol{\alpha}}(\boldsymbol{x})$, if the lens consists of a single plane by

$$\boldsymbol{\alpha}(\boldsymbol{x}) = \frac{\mathcal{D}_\mathrm{l}\mathcal{D}_\mathrm{ls}}{\mathcal{D}_\mathrm{s}}\tilde{\boldsymbol{\alpha}}(\boldsymbol{x}) \tag{3}$$

where $\mathcal{D}_\mathrm{s}$ is the angular size distance to the source and $\mathcal{D}_\mathrm{ls}$ is the angular size distance between the lens and the source. In the case of multiple lens planes, $\boldsymbol{\alpha}(\boldsymbol{x})$ can be interpreted as simply the displacement of the light-ray when it reaches the source plane projected on to a plane at $\mathcal{D}_\mathrm{l}$. Another very useful quantity is the critical surface density defined as

$$\Sigma_\mathrm{crit} \equiv \frac{c^2}{4\pi G}\frac{\mathcal{D}_\mathrm{s}}{\mathcal{D}_\mathrm{l}\mathcal{D}_\mathrm{ls}} \tag{4}$$

where G is Newton's constant and c is the speed of light.

3 CALCULATING THE DEFLECTION

GLAMER has several options for calculating the deflection angle, $\boldsymbol{\alpha}(\boldsymbol{x})$, designed for different applications. The lens can be represented by an analytic model in which case the surface density, deflection and shear at each point $\boldsymbol{x}$ can be calculated analytically. Analytic models are discussed at length elsewhere (see for example Schneider, Ehlers & Falco 1992) and the implementation in GLAMER code is not greatly different so they will not be discussed further here. A more difficult case arises when the lens consists of many components. These might be particles from a *N*-body/SPH simulation, individual stars in the lensing galaxy, or many clumps and subclumps of matter represented by analytic profiles. In these cases, it might not be optimal or practical to simply sum up the contributions from each component because of their large number, perhaps billions. In the case of *N*-body/SPH simulations, there is the additional complication of smoothing the mass distribution.

3.1 Smoothing

A method for smoothing the density distribution is required when dealing with *N*-body/SPH particles from a simulation. GLAMER uses an SPH-like approach where the smoothing scale for a particle, r_sm, is the radius around the particle which contains the nearest N_sph particles. This gives an adaptive smoothing which is smaller where the particles are denser.

The question arises as to whether to smooth according to the number of particle neighbours in three dimensions or in projection, two dimensions. The former is clearly truer to the intent of the simulation, but the latter is faster. Li et al. (2006) looked at several different smoothing methods in lensing simulations and found that smoothing in three dimensions is better than in two. Our experiments confirm this conclusion. Two-dimensional smoothing tends to undersmooth the low-density particles in the foreground and background of the lens. These particles often constitute a substantial fraction if not the majority of the total integrated density along the line of sight. Note that in the 3D case, there is not a unique smoothing scale for a point on the image plane which introduces an additional complication that will be discussed in Section 3.2. The 3D smoothing can be done once for a simulation since it is valid for all projections.

Calculating the number of particle neighbours is facilitated by sorting the particles into a tree structure, which is also used in the deflection calculation that is described in the next section. This procedure is parallelizable in a straightforward way and need only be done once per simulation so it is not a significant bottleneck.

The mass of each particle is treated as if it were distributed as a 2D Gaussian

$$\Sigma(\boldsymbol{x}) = \frac{m}{2\pi r_\mathrm{sm}^2}\frac{\exp\left[-\frac{|\boldsymbol{x}|^2}{2r_\mathrm{sm}^2}\right]}{\left[1-\mathrm{e}^{-q^2/2}\right]} \times \begin{cases} 1, & |\boldsymbol{x}| < q\, r_\mathrm{sm} \\ 0, & |\boldsymbol{x}| > q\, r_\mathrm{sm} \end{cases}, \tag{5}$$

where m is the particle mass and q is a cutoff so that it has compact support. Other profiles are possible, for example the traditional smoothed particle hydrodynamics (SPH) polynomial kernel, but (5) is what is usually used in GLAMER.

In some cases, the simulation has particles of different masses. These may represent different components – stars, gas, dark matter – or the structure under consideration might have been re-simulated with higher resolution so that there are small mass particles near the centre and very massive particles that represent the larger-scale structure surrounding the lens. In this case, each species of particle is considered separately, the smoothing scale depends only on the

density of particles of that species. If this is not done the density distribution can become unrealistically grainy; very massive particles on the outskirts of a high-resolution region can be given very small smoothing lengths because of their many less massive neighbour particles.

3.2 Calculating the lensing quantities

The lensing equation (1) relates points on the lens plane, $\boldsymbol{x}$, to points on the source plane, $\boldsymbol{y}$. The deflection angle $\boldsymbol{\alpha}(\boldsymbol{x})$ and the derivatives of the lens equation need to be calculated. By tradition the derivatives are grouped into the convergence, $\kappa(\boldsymbol{x})$, two components of shear, $\boldsymbol{\gamma}(\boldsymbol{x})$ and time delay, $\delta t(\boldsymbol{x})$, which are defined as

$$\boldsymbol{\alpha}(\boldsymbol{x}) = \nabla\psi(\boldsymbol{x}) \tag{6}$$

$$\kappa(\boldsymbol{x}) = \frac{1}{2}\mathrm{tr}\left[\frac{\partial\boldsymbol{\alpha}}{\partial\boldsymbol{x}}\right] = \frac{\Sigma(\boldsymbol{x})}{\Sigma_{\rm crit}} \tag{7}$$

$$\gamma_1(\boldsymbol{x}) = \frac{1}{2}\left[\frac{\partial\alpha_1}{\partial x_1} - \frac{\partial\alpha_2}{\partial x_2}\right] \tag{8}$$

$$\gamma_2(\boldsymbol{x}) = \frac{\partial\alpha_2}{\partial x_1} = \frac{\partial\alpha_1}{\partial x_2} \tag{9}$$

$$\delta t(\boldsymbol{x}) = \frac{1}{2}|\boldsymbol{\alpha}(\boldsymbol{x})|^2 - \psi(\boldsymbol{x}), \tag{10}$$

where the second equality in (7) and (10) are valid only for a single, thin lens plane. The potential can be calculated from the Poisson equation

$$\nabla^2\psi(\boldsymbol{x}) = 2\,\kappa(\boldsymbol{x}). \tag{11}$$

The magnification of a point is $\mu(\boldsymbol{x}) = \left[(1-\kappa)^2 - |\boldsymbol{\gamma}|^2\right]^{-1}$. Quantities (6) through (10) are the lensing quantities that we wish to calculate. However, when calculating the images of finite sized sources, it is often the case that only the deflection angle (6) is needed. For this reason, the deflection solver has an option that turns off the calculation of these quantities to save time.

The majority of ray-shooting simulations in the past have used a fast Fourier transform (FFT) based deflection solver. In this scheme, the surface density is interpolated on to a regular, 2D grid (for example Amara et al. 2006; Meneghetti et al. 2008; Hilbert et al. 2009). The lensing quantities can then be found by FFT through their connection to the lensing potential, (6) through (11). This method has the advantage of being fast – scaling as $N\log N$ and not N^2 as in the case of a direct summation. The FFT method, however, does suffer from several drawbacks. The first is that the resolution of the lensing calculation is limited to the grid size on which the FFT is performed. This introduces an additional scale into the problem. When working with simulation particles, the conservative thing to do is to make the grid scale much smaller than the smoothing scale everywhere on the plane. This can be very inefficient in regions where the smoothing scale is large. It is possible to use a multigrid approach Hilbert et al. (2007), where a smaller region around the area of interest is gridded at higher resolution. This adds complexity especially when combined with some adaptive scheme as will be described in Section 4. Also, the FFT method often requires padding the field with zeros to neutralize periodic boundary conditions. This further expands the grid. The time bottleneck with the FFT is usually interpolating the mass to the grid. This makes it laborious when many projections of the density or many changes in the mass distribution are needed when lens fitting for example. Since the entire highresolution grid needs to be stored, the FFT approach can require a great deal of memory especially when many lens planes are needed (see Paper II). Finally, point masses representing stars cannot be handled with a pure FFT approach.

GLAMER has the ability to read in mass maps and use them as lens planes. In this case, it uses the FFT approach to find the lensing quantities. In all other cases, for the reasons given above and others, GLAMER adopts a tree deflection solver very similar to tree force solvers commonly used for N-body simulations (Barnes & Hut 1989). Some modifications have been made to the algorithm to improve its performance for lensing applications as will be discussed below. This method starts with an expression for the lensing quantities for a single particle or halo. It can be applied to both simulation particles and clumps of matter with analytic profiles which might represent galaxies, dark matter haloes, subhaloes, stars or other components of a lens model. Here, we will concentrate on simulation particles and make note of the places where other types of 'particles' differ.

In the code, there are expressions for all of the lensing quantities for each type of 'particle' (or halo). For example, the deflection angle for the smoothed particle (5) is

$$\boldsymbol{\alpha}(\boldsymbol{x}) = \frac{m}{\pi\,\Sigma_{\rm crit}}\frac{\boldsymbol{x}}{|\boldsymbol{x}|^2} \tag{12}$$

$$\times\begin{cases}[1-\mathrm{e}^{-\frac{|\boldsymbol{x}|^2}{2r_{\rm sm}}}]/[1-\mathrm{e}^{-q^2/2}]\,, & |\boldsymbol{x}| < q\,r_{\rm sm}\\ 1\qquad\qquad\qquad\quad\,, & |\boldsymbol{x}| > q\,r_{\rm sm}\end{cases} \tag{13}$$

(see Aubert et al. 2007). Outside of the outer radius of any spherically symmetric mass profile all the lensing quantities are identical to those for a point-mass. In some cases, the analytic expressions contain special functions that are time consuming to evaluate [Navarro, Frenk & White (NFW) haloes for example]. In these cases, a look-up table is created and the values are interpolated from the tables when they are needed.

All the lensing quantities obey the superposition principle so the contribution from each particle or halo can be added together to form the total. The particles are sorted into a quad-tree structure. This is a tree data structure where each of the branches contains four equal area sub-branches or children. We have experimented with binary trees and equal particle number divisions of the boxes, but found this equal area quad-tree method to be superior. A branch is not subdivided and becomes a *leaf* when it contains less than a certain number of particles, typically five.

As the tree is constructed, the monopole and quadrupole moments of the mass in each box are calculated. These moments are calculated with only the projected, 2D, distances of the particles. Also while the tree is constructed, the quantity

$$r_{\rm cut} \equiv \frac{r_{\rm cm}}{\theta_{\rm force}} \tag{14}$$

is calculated for each box where $r_{\rm cm}$ is the largest distance from the centre of mass to an edge of the box. The quantity $\theta_{\rm force}$ is an adjustable parameter that needs to be set after numerical convergence tests (see Section 5). At the same time as the moments are calculated, a list is made of particles within that branch that have sizes larger than half the box size and have not been included in any parent branch's list. The purpose of this list will be explained later.

To calculate the deflection at a point, or ray, $\boldsymbol{x}$ the code 'walks the tree'. As it reaches each box it evaluates the *multipole acceptability criterion* (MAC) which in this case is

$$|\boldsymbol{x} - \boldsymbol{x}_{\rm cm}| > r_{\rm cut} \text{in 2D}, \quad (15)$$

where $\boldsymbol{x}_{\rm cm}$ is the box's centre of mass. If this expression is true the multipole moments of this box are used to calculate the lensing quantities. If the expression is false, the code descends into the box's children and repeats the process. When the code reaches a leaf – a box with no children – the particles inside this box are added by direct summation. In this way boxes that fill a small angle as 'seen' from the point $\boldsymbol{x}$ are taken as a single unit because of (14).

An additional problem arises in the lensing application of this algorithm that is not present for an *N*-body or SPH simulation. This we will call the *big particle problem*. In the lensing case, the 2D position of a particle might not be related to its size. This might arise because small haloes might be within larger haloes or, in the case of smoothed *N*-body particles, the size of the particles depends of the 3D position so background and foreground particles in low-density regions will be much larger than the particles near the centres of the haloes/galaxies. An additional term can be added to the MAC, (equation 14), containing the size of the biggest particle within the branch. In this way direct summation would be triggered for any particle that the ray intersects. This solution is highly inefficient in some circumstances. The algorithm will always descend to the leafs containing the particles the ray intersects with and in the process will over resolve neighbouring branches that may contain many particles that do not intersect the ray. Our solution is to keep a list of particles in each branch that have sizes close to the size of the branch. This allows for a fast method of identifying which particles intersect the ray so that they can be added individually and their contribution removed from the standard tree calculation. This amounts to essentially pre-sorting the particles by 2D position and size so that they can be searched according to a combination of these. We have experimented with other methods to solve the big particles problem, but have found this to be the most efficient. The accuracy of this algorithm is tested by comparison to direct summation.

When the line of sight through a simulation is changed the particle positions are updated in place and a new tree structure is constructed with the z-axis corresponding to the line of sight. Having boxes aligned with the line of sight rather than at an arbitrary angle significantly reduces the time spent calculating the lensing quantities which more than makes up for the initial cost of building the tree.

There is a choice of whether to use a 3D or 2D tree in the calculation of the lensing quantities. With our solution to the big particle problem and our decision to use the planer lens approximation the 2D tree is more efficient in memory and speed although a 3D tree option is still implemented.

3.3 Parallelization

This tree deflection solving algorithm is easily parallelized by distributing the points on the image plane that need to be calculated amongst processors. This becomes more efficient the bigger the chunks of grid points that are done in each call to the deflection solver, see Section 4. Currently GLAMER uses POSIX threads to distribute the points into separate threads to be executed by different cores on a shared memory machine.

3.4 Interpolation

It often occurs that a grid cell needs to be refined based on a criterion related to the source's surface brightness (see Section 4), but the grid cell is small enough compared to a structure in the lens that the lensing can be considered a linear distortion within the cell. Shooting rays for the newly added grid points with the full deflection solver is inefficient. GLAMER uses a scheme that automatically detects if the magnification matrix is uniform over the cell and uses a linear expansion of the deflection angle based on the surrounding grid points to find the source positions of the new subcells. This method requires no a priori assumption about the scale of structures in the lens relative to the source. The process is complicated by the fact that the grid is not uniform, but the machinery developed for finding and mapping images and discussed in Section 4 make the basic required straightforward.

As will be discussed in Section 4, cells are defined by adding rays around an already shot ray. When the magnification matrix is uniform across the cell the source position of any point within the cell can be found from the position of the centre of the cell $\boldsymbol{x}_{\rm centre}$ and its source position $\boldsymbol{y}_{\rm centre}$ which was already calculated in an earlier stage of refinement:

$$\boldsymbol{y} = \boldsymbol{y}_{\rm centre} + \mathbf{A}_{\rm centre}(\boldsymbol{x} - \boldsymbol{x}_{\rm centre}), \quad (16)$$

where $\mathbf{A}_{\rm centre}$ is the magnification matrix (A1) which can be found from the lensing quantities (7) through (9).

For each cell that is to be subdivided the code finds all of its neighbour cells. The κ and $\boldsymbol{\gamma}$ at these points are compared to the centre of the cell and if they are found to be within a small tolerance (typically $\pm 10^{-4}$) the approximation (16) is used.

It is sometimes advantageous for precision and speed to not calculate κ and $\boldsymbol{\gamma}$. Calculating these quantities for a collection of asymmetric masses can incur a significant time penalty. Also to save memory, the κ and $\boldsymbol{\gamma}$ values at each point are stored to floating point precision, but it is often necessary to calculate the deflection to higher precision. In these cases, the magnification matrix can be found by using the deflection angles of the neighbouring points only. Appendix A shows how this is done. Sets of two neighbours are used to find $\mathbf{A}$ and if they all agree to the tolerance level, the average $\mathbf{A}$ is used to calculated the deflection in (16).

3.5 Implanting stars in a smooth background

GLAMER has the ability to do lensing by a uniform distribution of stars or by stars contained within a much larger lens galaxy. The deflection caused by stars can be calculated in the same way as for any other 'particle', but in this case the particle size is zero. Stars can cause *microlensing* where what would have been a single image is broken up into many micro-images which are generally not individually resolved because of their small separation ($\sim$ micro-arcseconds). The distinguishable separate patches of images are referred to as macro-images.

It is not possible, or necessary, to represent all of the stars in a lens galaxy ($\sim 10^{11}$) as particles. Only stars relatively close to the images are needed. The stars cannot simply be added to the smooth component of the lens because the smooth mass distribution is meant to represent the stars, gas and the dark matter so the mass in stars must be subtracted from the smooth model near the images. The code must also take care not to overpopulate regions from separate macro-images that might overlap.

When implanting the stars, the code subtracts a uniform density disc equal to the mass in stars surrounding each macro-image. The

positions of the macro-images are found by either taking observed ones or finding them initially without stars. The stars are then given random positions within each disc region. In this way, structure in the lensing on all scales from the size of the whole dark matter halo down to tens of au can be represented.

The deflection from each star is

$$\boldsymbol{\alpha}_*(\boldsymbol{r}) = \frac{m}{\pi \Sigma_{\rm crit}} \frac{\hat{r}}{r} = \frac{r_{\rm E}^2}{r}\hat{r}. \quad (17)$$

where this defines the Einstein radius $r_{\rm E}$. The same tree algorithm with a 2D tree is used in this case to calculate the total deflection. Each of the stars may have equal masses or different masses randomly drawn from a mass function.

Schneider et al. (1992) estimate that the number of stars that need to be represented in a simulation is $\sim 100\langle\mu\rangle\kappa_*^2$ where $\langle\mu\rangle$ is the magnification if there were no stars and κ_* is the surface density of stars in units of the critical density. This criterion is always amply satisfied. Typically 10 000 stars per macro-image are used. Good convergence is found with this large number of stars.

3.6 Mass sheets

As mentioned earlier, it is possible to read a map of the surface mass density into GLAMER and use it as a lens. In this case, the lensing quantities are calculated by FFT and interpolated from the grid. This is particularly useful when dealing with *N*-body/SPH simulations of entire light cones on cosmological scales where the number of particles can be very large. The particle distribution can be smoothed and projected on to planes which results in a very large savings in memory usage.

3.7 Multiple deflection planes

GLAMER does have the ability to calculate the deflection from multiple planes as is necessary for ray-tracing through a large-scale cosmological *N*-body simulations. This ability is discussed in detail in a companion paper (Petkova et al. 2014).

4 FINDING AND MAPPING THE IMAGES

Generally a source's position and shape is chosen and the task is to find its images. The number, size and shape of images are not known a priori. Some images can be highly elongated which presents a problem when doing calculations on a rectangular grid. Some images can be much smaller than others. For finding the images of point sources the problem can be reduced to minimizing $|\boldsymbol{y}(\boldsymbol{x}) - \boldsymbol{y}_{\rm source}|$ with respect to $\boldsymbol{x}$ where $\boldsymbol{y}_{\rm source}$ is the source position. As in many minimization problems, difficulties arise from not knowing the number of minima (images) and in separating close images. For finite size sources the most widely used method is to simply fill the image plane with a uniform grid, calculate the source position of each point and find which points are within the source. This is highly inefficient if ray-shooting is expensive since the grid spacing needs to be fine enough that many points will land inside the smallest image that one needs to resolve. For small sources with widely separated images the problem is especially acute. It is also important for cosmological simulations where the images of high-redshift galaxies are typically shifted by arcminutes while their images can be smaller than an arcsecond.

We adopt an AMR scheme. The points on the source plane are linked to their image points and vice versa. The source and image points are sorted into separate kd-trees so that they can be quickly searched for nearest neighbours and neighbour points within a fixed distance. These kd-trees are 'live' in the sense that points can be continuously added to them as the grid is refined.

We will first describe how a circular, uniform surface brightness source is found and refined, and then discuss sources with varying surface brightness. Because lensing conserves surface brightness, in the case of uniform surface brightness the magnification is found by finding the area of the image and dividing by area of the source. A point on the image plane whose corresponding point on the source plane is within the source is a point in the image.

The first step is to find the images. On a coarse grid one would not expect to find any points within any image that is much smaller than the grid resolution. To avoid this problem, we employ a telescoping source strategy. First, an initial course grid is put down with grid spacing $\Delta_{\rm init}$. The source size is first set to $\Delta y_{\rm source} = \Delta_{\rm init}/\sqrt{\mu_{\rm min}}$, where $\mu_{\rm min}$ is the absolute value of the magnification of the smallest image that one is required to find. If the source is not circular, $\Delta y_{\rm source}$ represents the largest linear dimension of the source. All the points within source are found. If any of these points have a grid size larger than $\Delta y_{\rm source}\sqrt{\mu_{\rm min}}/3$ it is refined. A grid cell is always refined by dividing it into nine subcells and thus reducing the grid spacing by 1/3. After the refinements the source size is reduced by 1/3 and the process is repeated. This is done until the desired source size is reached at which point a more complex refinement scheme is adopted. Sometimes the full range of telescoping is not necessary when the source has not moved, but has only changed in size. It is possible for this algorithm to miss small, separated images while refining other images to a much smaller scale than the missed image. The extent to which this happens is investigated in Section 5.3.

When the target source size is reached in the telescoping stage the code separates individual images by a neighbours-of-neighbours algorithm on the image points. Neighbours are cells that border one another. Note that this means that images of different parity that touch each other, at a critical curve, will be classified as one image. The points on the edges or borders of each image are also found. The inner border includes points inside the image and the outer border is the points outside the image, but bordering on it. The code goes through repeated grid refinements until a termination criterion is met. In each refinement, only the grid points within the image or its outer border with the largest grid size are refined. First, a uniform grid refinement – all cells within the image and outer border – is done which ensures that each detected image has at least 50 points in it.

There are several choices for the termination criterion depending on the application. Common to all of them is that the outer border points must be at least as well refined as the inner border points for each image. This allows the refined grid to expand out along narrow images whose boundaries were not well characterized in the initial refinement steps. If all that is required is an image with a certain resolution the grid can be refined until all the points within the image have that resolution. If we are interested in the magnification ratios of images it is desirable that the area of each image be calculated with equal fractional accuracy. In this case, the grid is refined until the largest cell in the image border is a small fraction of the total area of that image, typically $<5.0 \times 10^{-4}$. This results in the grid refining to smaller scales around smaller images. Some images might be so small that they are unobservable as in the central image of a singular mass distribution. For this reason, an additional minimum grid size requirement is imposed so that the code does not refine indefinitely. In other cases, such as microlensing, the images are observationally indistinguishable and only the total area of all images is important.

In this case, the same fractional criterion is imposed applied to the images in total or the macro-image.

Each refinement of the grid requires a batch of evaluations of the lensing quantities. These are calculated in parallel as mentioned in Section 3. This makes the timing scales almost linearly with the number of cores until the neighbours-of-neighbours and image border search become the bottleneck. For a simple analytic lens model, the calls to the deflection solver can be very fast in which case the grid refinement is the bottleneck. When there are many lens planes with many haloes on each many cores ($\gtrsim$20) can be used before this bottleneck is reached.

A way to reduce the grid refinement bottleneck is to break the image plane up into regions, do the image finding and grid refinement in them separately. We are so far able to do this for widely spaced patches of sky whose projections on to the source plane do not overlap. This mode of parallelization will be further developed in the future.

4.1 Continuous surface brightness sources

Realistic sources will not be circularly symmetric or constant surface brightness. For a varying surface brightness the refinement strategy must be modified. The same telescoping and initial refinement is done with a circular source that is known to circumscribe the regions of the source that has significant flux. The edge refinements are not done. Instead, a refinement scheme based on the surface brightness of the source is employed. We have tested several schemes for this and found two to be useful. The first is to simply require that the surface brightness in every cell not differ from the surface brightness in all its neighbours by more than a fixed threshold. This makes smooth images but can sometimes over refine peaked surface brightness profiles. Another strategy is to estimate the Gaussian curvature of the surface brightness at each cell by comparing it to its neighbours. From the curvature an estimate of the error in the flux in that cell is made. This error is then compared to the total flux of the image or all the images and refinement is stopped when this goes below a tolerance (typically 10^{-4}). This second method does a good jobs of measuring the total flux in an image, but occasionally when making images of realistic galaxies the outskirts of the galaxy, which contain a small fraction of the flux, can be less smooth than is desired. We retain the two methods as options and continue to seek a more perfect solution.

4.2 Mapping critical curves and caustics

It is often desired to calculate the caustic curves and their images, the critical curves. This is done by a modification of the process used to finding images. The regions of negative magnification are treated as images. Since points on either side of a critical curve have magnifications of different sign the borders of all the negative magnification regions are the critical curves. Instead of refining the whole image region as for the images only the borders are refined. This is done until the required resolution is reached. The corresponding caustic curve is automatically found in the process since the caustic is its image. The points in the curves are separated into separate curves by a neighbours-of-neighbour algorithm and then each curve is ordered to make a continuous curve. The critical curves are always closed curves or lead to the border of the region being simulated.

4.3 Further comparison between the tree and FFT algorithms

A final note on the relative scaling of FFT method versus the tree-code-AMR method employed within GLAMER, as noted before the bottleneck for an FFT code is generally putting the mass on to a uniform grid which scales as $N_{\rm grid}N_{\rm smooth}$, where $N_{\rm grid}$ is the number of grid points and $N_{\rm smooth}$ is a representative number of grid points within a smoothing length of a grid point, in projection. Within strong lenses, if accuracy at high magnifications is required, a large amount of smoothing is often necessary and $N_{\rm grid}$ could be 10 000 or larger. Thus this part of the code scale less well then sorting the particles into a tree($N_{\rm part}\log N_{\rm part}$). Calculating the force in the tree code scales like $\sim N^*_{\rm grid}\log N_{\rm part}$, where $N^*_{\rm grid}$ is the number of grid points in the AMR grid which can be orders of magnitude smaller than the one for an FFT grid to reach the same resolution. It is true that one can interpolate to smaller scales from the FFT uniform grid, but it is required that the initial grid spacing be smaller than the smallest smoothing length (zero in the case of stars).

Once the deflection angles have been found in the FFT approach the image finding is fairly trivial if it has been done to high enough resolution – simply find which source points lie within the source. In this way the hard work can be done up front and the source can be changed at will later. The difficulties come in when many lenses configurations and/or projections are required, or extreme dynamical range or very small particles are required.

5 TESTS

In this section, we briefly describe some of the tests we have performed on the code.

5.1 The tree deflection solver

To test the tree deflections solver, we have done a series of test some of which will be presented here. First, we test the convergence of the deflection solver's output with decreasing $\theta_{\rm force}$ (defined in Section 3.2). With $\theta_{\rm force}=0$ the deflection solver adds the particles' contributions to the lensing quantities one by one. As $\theta_{\rm force}$ increases the code uses the mass moments of larger and larger tree branches to calculate these quantities and gets faster.

We start with random set of point masses in a circular region and calculate the deflection and shear at the centre with increasing $\theta_{\rm force}$. The background convergence, κ, is identically zero in this case. Figs 1 and 2 show how these quantities converge to the $\theta_{\rm force}=0$ values as $\theta_{\rm force}$ decreases. The masses are given unit mass, the radius of the circle is set to unity so that the effective optical depths (the density of stars in units of their Einstien radius, $\Sigma_{\rm star}\pi r_{\rm E}^2/m_{\rm star}$) are 10^4 and 10^3 which are very high compared to those usually encountered in observations, but are extreme case that challenge the deflection solver.

We repeat the convergence test for our halo models. Here we will show the NFW halo. Again a unit circle is populated with masses. The sizes of the haloes are taken to be uniformly distributed up to 0.3. The masses are taken from a distribution $\frac{{\rm d}n}{{\rm d}m}\propto m^{-2}$ with a maximum mass of unity and a minimum of 10^{-3}. They are all given a concentration of 3. The algorithm should identify all haloes that intersect the ray and calculate their $\boldsymbol{\alpha}$, κ and $\boldsymbol{\gamma}$ by direct summation. Since only haloes that intersect the ray contribute to κ there should be no error for any choice of $\theta_{\rm force}$ compared to the $\theta_{\rm force}=0$. We verify that this is true. The errors in $\boldsymbol{\alpha}$ and $\boldsymbol{\gamma}$ are shown in Fig. 3

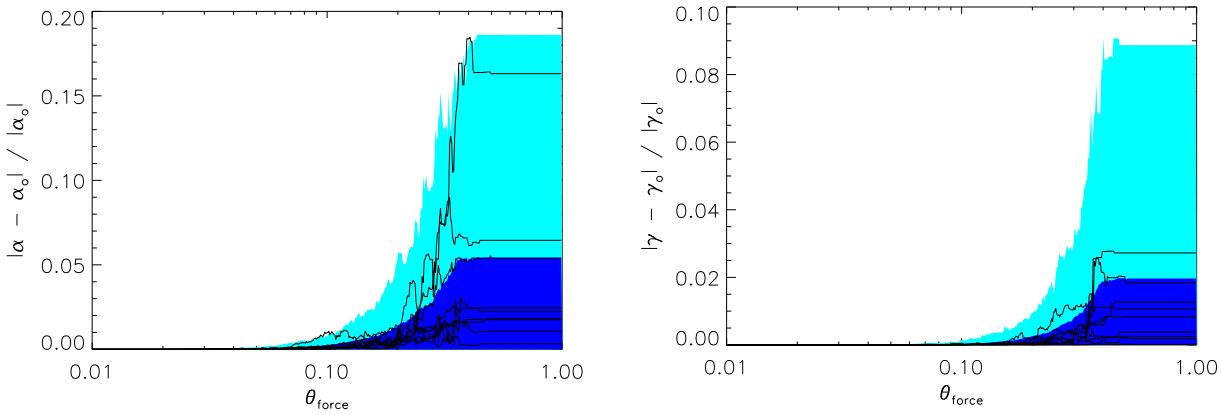

Figure 1. Convergence test for tree deflection solver in the case of point masses. Within a circular region 10 000 point masses are placed randomly and a ray is shot through the centre of the circle. The parameter $\theta_{\rm force}$ (equation 14) is increased from 0 to 1. This is repeated 1000 times with different positions for the masses. On the left is the deflection angle $\boldsymbol{\alpha}$ relative to the deflection angle for $\theta_{\rm force} = 0$, direct summation case, the fractional error caused by the tree calculation. On the right is the same for the shear, $\boldsymbol{\gamma}$. The blue regions are where 90 per cent of the cases fall and the cyan regions are where 99 per cent of the cases fall. The curves are the first 10 realizations of the masses.

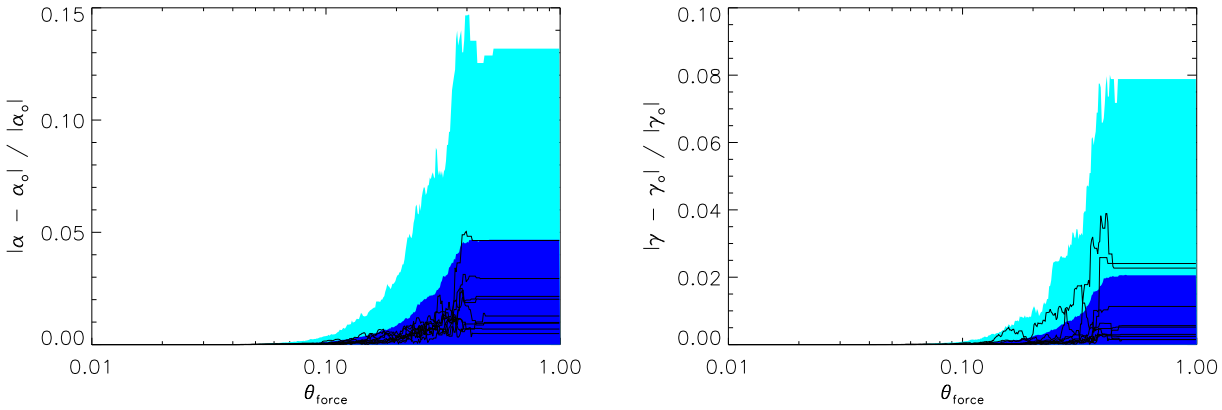

Figure 2. Convergence test for tree deflection solver in the case of point masses. There are 1000 point masses per realization. There curves and shaded regions are as in Fig. 1.

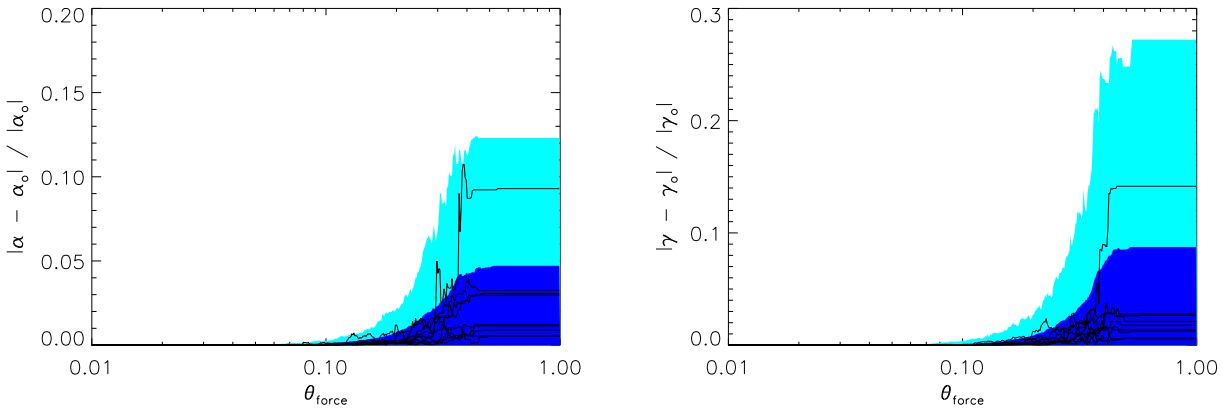

Figure 3. Convergence test for tree deflection solver in the case of halo masses. The halo properties are distributed as described in the text. There are 10 000 haloes per realization. There curves and shaded regions are as in Fig. 1.

for 10 000 haloes within the circle. We also repeated the exercise with 100 haloes per realization with smaller sizes (uniform up to 0.2) shown in Fig. 4.

The convergence of $\boldsymbol{\gamma}$ with $\theta_{\rm force}$ is generally better than for $\boldsymbol{\alpha}$ because these quantities fall off faster with distance between the ray and the centre of mass. For this reason, it is the accuracy of $\boldsymbol{\alpha}$ that determines how large a $\theta_{\rm force}$ is acceptable. We conclude from these test and others that a value of $\theta_{\rm force} = 0.1$ is sufficiently small for most applications.

5.2 Singular isothermal sphere (SIS) tests

Lensing by an SIS is particularly easy to work with analytically. Appendix B summarizes some analytic properties of an SIS lens

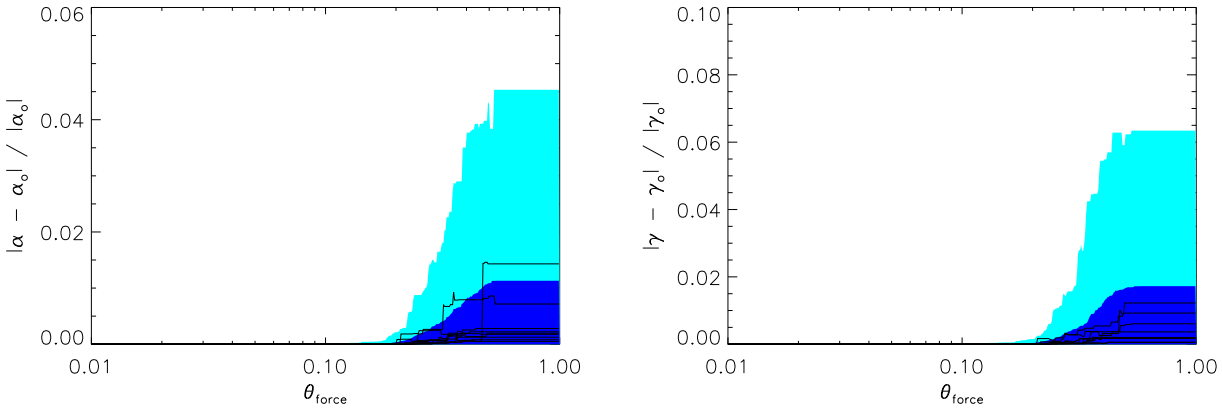

Figure 4. Convergence test for tree deflection solver in the case of halo masses. The halo properties are distributed as described in the text. There are 100 haloes per realization. There curves and shaded regions are as in Fig. 1.

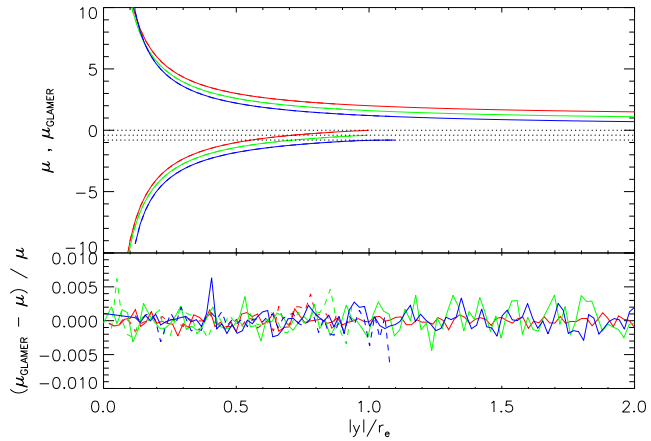

Figure 5. A comparison of the analytically calculated image magnifications for an SIS with those calculated with GLAMER. Above are the magnifications for three different source sizes – 0.1 (blue), 0.01 (green) and 0.0001 (red) Einstein radii. The results for each source size are offset vertically so that they can be differentiated. The dotted line marks zero magnification in each case. The negative magnification curves are for the negative parity image that is closest to the centre of the lens. This image does not appear when no part of the source is within one Einstein radius, r_E, of the centre of the lens. Both the analytic result (solid curves) and the GLAMER result (dashed curves) are shown in the top panel, but they are hard to differentiate. The fractional difference between them is shown on the bottom with the same colour scheme.

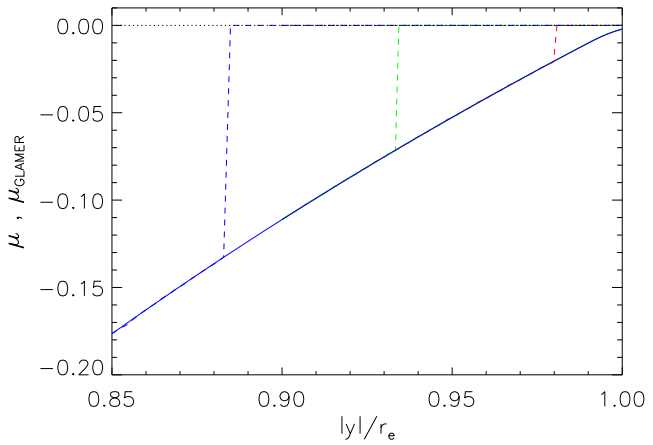

Figure 6. The magnification of the negative parity image in an SIS model according to analytic calculation (solid curves) and the GLAMER code (dashed curves) as a function of source position. The source radius is $0.01r_E$. The GLAMER code loses the image and its magnification goes to zero before the correct solution does. The values of μ_{min} from left to right are 0.16 (blue), 0.09 (green) and 0.01 (red). A μ_{min} of 0.09 should catch all images with a magnification above 0.07 for example.

and how to calculate the magnification of a finite source with simple numerical integrations. These solutions can be compared to the results of GLAMER to test its grid refinement.

We set up an SIS lens with GLAMER for a lens at $z = 0.34$, a source at $z = 3.62$ and $\sigma = 300\,\mathrm{km\,s^{-1}}$ although these parameters should not have any effect on the results since all quantities are a function of the y/r_E where r_E is the Einstein radius (equation B2). With an initial 64-by-64 grid over an area of 20 r_E squared GLAMER finds the images and their magnifications. The centre of the source, y_c is moved from zero to $2r_E$ for three source sizes. The grid is erased and recreated for each source position. Otherwise hysteresis in the grid refinement can help in finding small images. The grid convergence requirement is set for each image individually.

The resulting magnifications are compared with the analytic results in Fig. 5. The errors are generally below 0.5 per cent and never greater than 0.7 per cent. The errors show no significant dependence on source size, image magnification or image shape. The smallest source tested here is 10^5 times smaller than the initial grid.

When the source passes outside of the circle $|y| < r_E$, the magnification of the negative image goes to zero. If the image is too small relative to the source size GLAMER will miss the image because of the finite resolution of the initial grid, as explained in Section 4. Fig. 6 shows the region of the magnification curves where the negative image is disappearing for a source size of $0.01r_E$. Decreasing the value of the parameter μ_{min} makes smaller images detectable at the expense of shooting and sorting more rays. In most realistic cases, there is at least one image with a magnification greater than or equal to one[1] so μ_{min} can also be seen as controlling the smallest

[1] One notable exception is when simulating a single macro-image of a larger strong lens separately as might be done for investigating microlensing. In this case, the total magnification could be significantly less than one and special care needs to be taken. In cosmological simulations it is also possible to get total magnifications below one for lines of sight that pass through under dense regions. In realistic cases, these are only weakly demagnified and the same argument holds that the total magnification (all images) of a source is $\gtrsim$1.

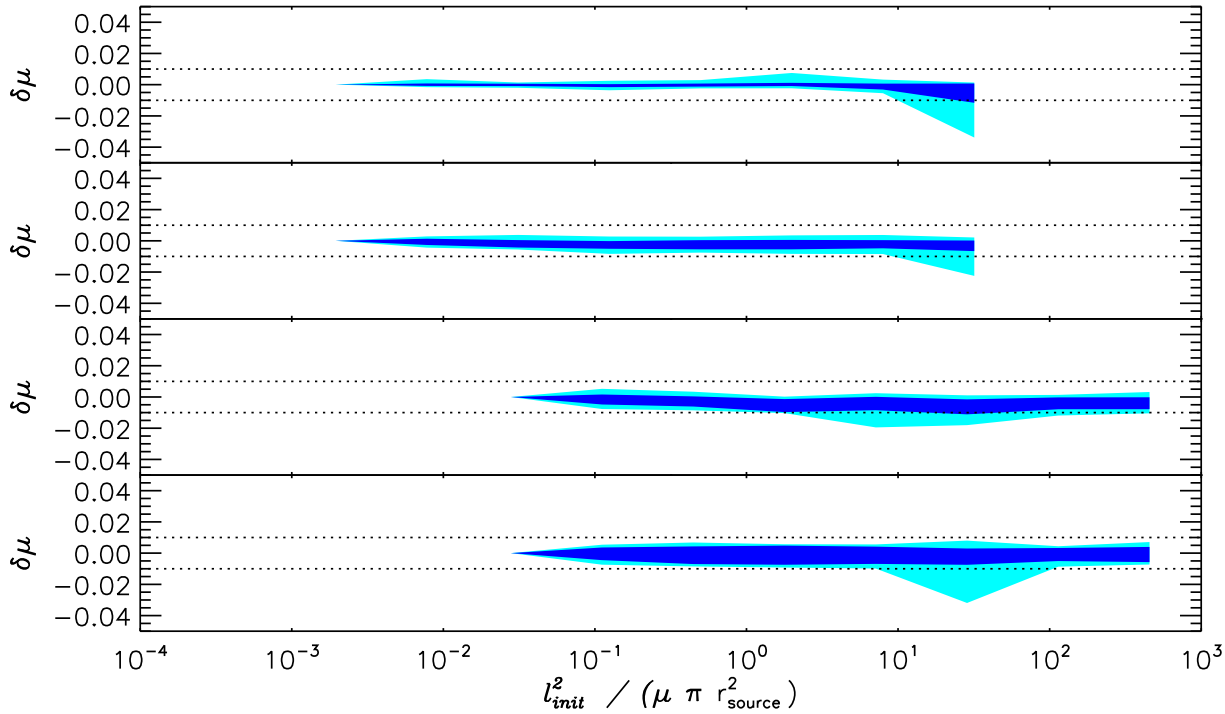

Figure 7. A test for how well the adaptive mesh method tracks images. A uniform distribution of stars with a background shear. The initial grid resolution is denoted $l_{\rm init}$ and $\bar{\mu}$ is what the magnification would be if the mass in the stars where spread uniformly. The error in the magnification caused by the adaptive mesh calculation is defined as $\delta\mu = \{\mu(l_{\rm init}) - \mu(l_{\rm init} = l_{\rm min})\}/\bar{\mu}$ where $l_{\rm min}$ is the smallest initial grid size used in the particular run. The top two panels are for $\bar{\kappa} = 0.45$ and $\bar{\boldsymbol{\gamma}} = \{0, 0\}$ and – from top to bottom – source sizes $r_{\rm source} = 0.001$ and 0.01 pc. The Einstein radius of the stars in these cases is 0.175 pc. The bottom two panels are for $\bar{\kappa} = 0.54$ and $\bar{\boldsymbol{\gamma}} = \{0, 0.4\}$ and with the same source sizes. The dark blue region contains 90 per cent of the cases at each initial grid size and the light blue regions shows the envelope of all the realizations. There are typically hundreds of micro-images in these simulations. For each panel there are 100 realizations of the stellar distribution except in the second from top where there are 120.

magnification ratio that is expected to be found. The $\mu_{\rm min}$ is adjusted for the particular application, but has a default value of 0.09.

5.3 Image finding and grid refinement

Quasar microlensing is the most challenging case for any ray-shooting code. In this case, a high-redshift quasars that is being strongly lensed by an intervening galaxy is also being affected by individual stars in that lens. The dynamic range is very large (tens of kpc down to $\sim 10^{-5}$ pc) and many (hundreds) of irregular shaped micro-images can be present. Because the deflection caused by a (Newtonian) point-mass diverges, each star is capable of producing an image no matter how far away it is from the direct line of sight to the source. But images around stars that are very far from the line of sight will be very small. It is not practical, or desirable, to find and map every image because some will be insignificant to the total magnification of the macro-image. The challenge here lies in finding the micro-images that contribute while shooting as few rays as possible. It has already been demonstrated that the code can calculate the magnification of an individual image accurately once it is found.

To test this aspect of the code, we compare the magnifications calculated with AMR to those calculated with a high-resolution initial grid that is guaranteed to find images smaller than a certain size even before any refinement has been done. We use what we call a uniform lens for this which initially has a uniform convergence and shear, $\bar{\kappa}$ and $\bar{\boldsymbol{\gamma}}$. Stars are then implanted randomly in a circular region and the corresponding mass is subtracted from the initial mass sheet. For these trials we use 10 000 stars. Each trial starts with a very coarse initial grid of 16 × 16 rays and increases the initial grid resolution until a point where small images would be identifiable on the initial grid without any refinement. The initial grid resolution is denoted $l_{\rm init}$. Getting a consistent result independent of the initial grid resolution indicates that the adaptive refinement is doing its job well and gives an indication of the error in the magnification. This is repeated 100 times with different star positions, but the same $\bar{\kappa}$ and $\bar{\boldsymbol{\gamma}}$. Then it was repeated for different $\bar{\kappa}$, $\bar{\boldsymbol{\gamma}}$ and source sizes. Fig. 7 shows some of these tests.

The code is usually able to measure the total magnification of the macro-image to better than 1 per cent. There are some outliers where the error is as big as 4 per cent, but less than 1 per cent of the time. This seems sufficiently accurate considering that typical observational uncertainties are ∼10percent. More accuracy is attainable by decreasing $\mu_{\rm min}$ at the expense of increasing the computational time.

Fig. 8, shows a portion of the macro-image produced in a simulation along with the grid pattern that the code chose to find its area. It can be seen how the code locates the images and then tries to refine around the edges of the images in the case of a uniform surface brightness circular source. This is accomplished even in the case of many very irregularly shaped images.

In Fig. 9 the code's ability to cover a very large range of angular scales is demonstrated. Here, the image magnifications are plotted for a source as a function of its radius from super-galactic scales down to 10^{-4} pc scales. A galaxy sized SIE (single isothermal ellipsoid) lens ($\sigma = 300$ km s^{-1}) is included and 10 000 stars are implanted in regions surrounding each macro-image. The locations

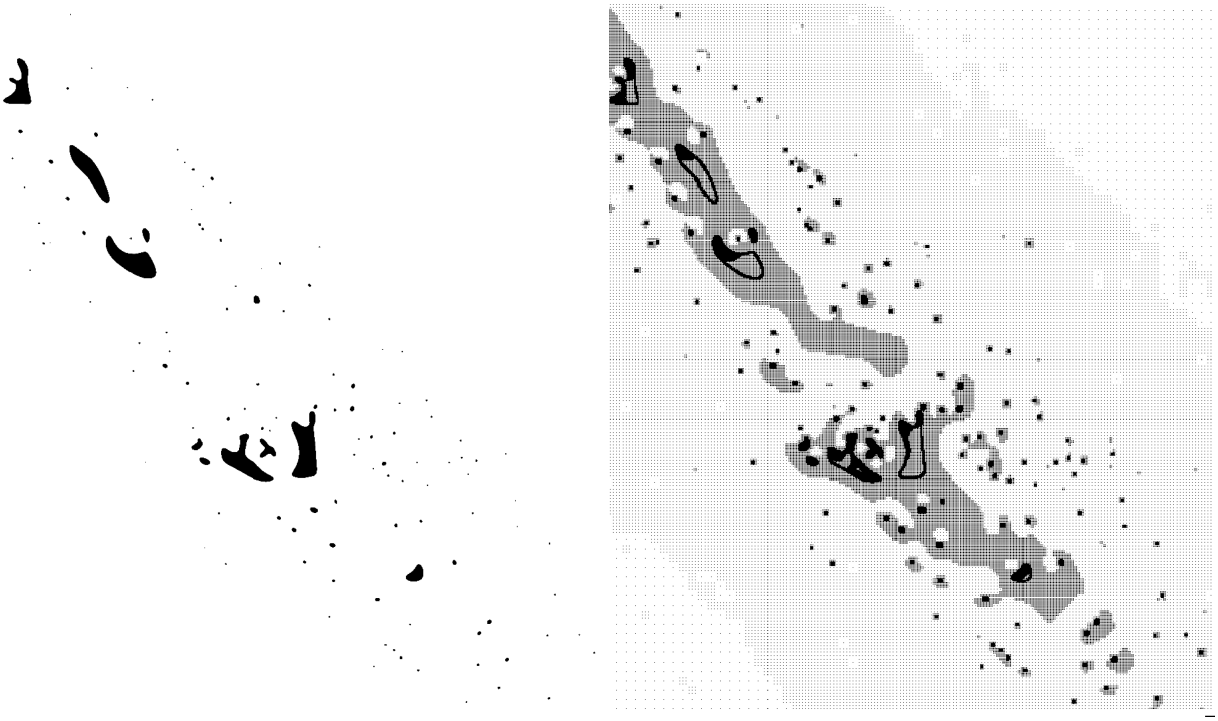

Figure 8. On the left is a detail of the images produced for a circular source lensed by stars with an average convergence in stars of $\bar{\kappa} = 0.54$ and a background shear of $\bar{\gamma} = \{0, 0.4\}$. On the right is the grid or ray pattern created during the refinement process to these images. There is a dot at each ray, although the pixelization of the image might make this hard to see. The grid was initialized in a field two orders of magnitude larger than this one with 16×16 pixels. The images extend well beyond the pictured region. It can be seen that for some images just the edges are refined to the highest visible resolution.

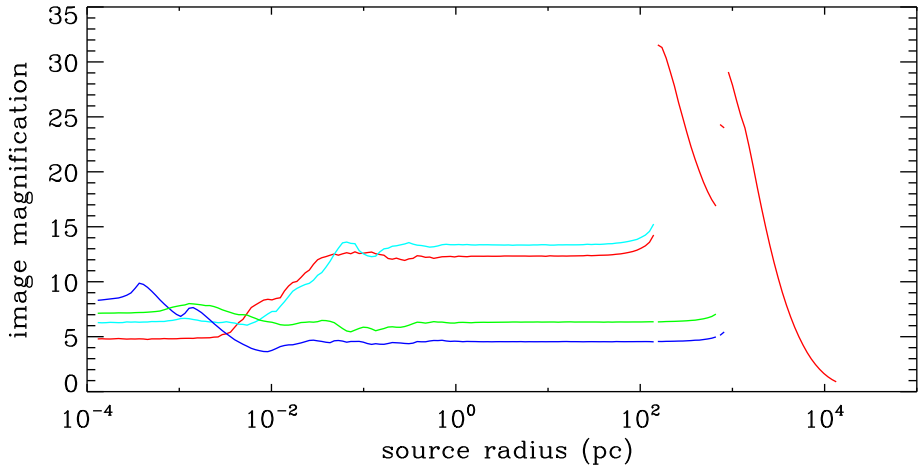

Figure 9. The magnification as a function of source size for a source at z=3.62 and an SIE lens with stars in it at z=0.3. The SIE has a velocity dispersion of $\sigma = 300$ km s^{-1}. At the positions of the images all the mass is in stars. For the largest source sizes the image is larger than the lens and has a magnification of about one. As the source gets smaller it forms an Einstein ring with high magnification. The ring splits into two (for a very short range), then three and then four as the source gets smaller. When the source gets below 1 pc in radius the four macro-images start to break up into micro-images and the magnifications start to fluctuate because of microlensing.

for implanting the stars are found by initially finding the image positions with the smooth host model without stars. All the mass is in stars at the image positions. Distinct ranges in source size can be seen in the plot where the host lens is important and where stars are important to the magnifications. The magnification curves have been calculated both from an adapted grid initialized with a size of 5 arcsec and with four separate grids initialized with a size of ∼10 pc on the lens plane centred on each of the macro-images. The same calculation has been performed with different values for $\mu_{\rm min}$ and numbers of points in the initial grid to confirm consistency of the magnification curves.

Countless other checks have been made during the development of the code. For example, all the analytic halo profiles have been checked and the interpolation has been compared to directly ray-shooting the points and found to agree to a part in 10^4 for the deflection. We have also constructed approximations to analytic lens profiles out of particles and tested that their lensing properties agree with analytic results.

6 DISCUSSION

We have succeeded in developing an adaptive lensing code that is flexible enough with its representation of sources and lenses that it can be used for many different applications. This code is capable of simulating everything from QSO microlensing to weak shear on multidegree scales while reconstructing individual images. The lensing mass can be represented by point masses (stars), smoothed simulation particles, analytic halo models, pixelized mass maps or any combination of these and easily switch between them. Likewise, the source can be represented by a uniform surface brightness circle, an analytic surface brightness model or a pixelized image.

We continue to develop GLAMER to make it a more useful tool. In a companion paper (Petkova et al. 2014), we describe how the code has been extended to allow for lensing through three-dimensional mass distributions so that whole light cones can be simulated. We continue to add to the number of source models available. More sophisticated methods of representing galaxies and QSO emission regions are being implemented. More sophisticated halo models are being implemented with a variety of profiles and asymmetries. Finally, the machinery used to do simulations is being adapted to do lens fitting for weak and strong lensing.

ACKNOWLEDGEMENTS

We would like to thank our colleagues in Bologna – F. Bellagamba, C. Giocoli, D. Leier and N. Tessore – for helpful contributions and for their further development of the code that will appear in future publications. This research is part of the project GLENCO, funded under the Seventh Framework Programme, Ideas, Grant Agreement no. 259349.

REFERENCES

Amara A., Metcalf R. B., Cox T. J., Ostriker J. P., 2006, MNRAS, 367, 1367
Angulo R. E., Chen R., Hilbert S., Abel T., 2014, MNRAS, 444, 2925
Aubert D., Amara A., Metcalf R. B., 2007, MNRAS, 376, 113
Barnes J. E., Hut P., 1989, ApJS, 70, 389
Fluke C. J., Webster R. L., Mortlock D. J., 1999, MNRAS, 306, 567
Hamana T., Mellier Y., 2001, MNRAS, 327, 169
Hilbert S., White S. D. M., Hartlap J., Schneider P., 2007, MNRAS, 382, 121
Hilbert S., Hartlap J., White S. D. M., Schneider P., 2009, A&A, 499, 31
Jain B., Seljak U., White S., 2000, ApJ, 530, 547
Killedar M., Lasky P. D., Lewis G. F., Fluke C. J., 2012, MNRAS, 420, 155
Li G.-L., Mao S., Jing Y. P., Kang X., Bartelmann M., 2006, ApJ, 652, 43
Mediavilla E., Muñoz J. A., Lopez P., Mediavilla T., Abajas C., Gonzalez-Morcillo C., Gil-Merino R., 2006, ApJ, 653, 942
Meneghetti M. et al., 2008, A&A, 482, 403
Pace F., Maturi M., Meneghetti M., Bartelmann M., Moscardini L., Dolag K., 2007, A&A, 471, 731
Petkova M., Metcalf R., Giocoli C., 2014, MNRAS, 445, 1954
Poindexter S., Kochanek C. S., 2010, ApJ, 712, 658
Sato M., Hamana T., Takahashi R., Takada M., Yoshida N., Matsubara T., Sugiyama N., 2009, ApJ, 701, 945
Schneider P., Ehlers J., Falco E. E., 1992, Gravitational Lenses. Springer-Verlag, Berlin
Takahashi R., Oguri M., Sato M., Hamana T., 2011, ApJ, 742, 15
Vale C., White M., 2003, ApJ, 592, 699
Vegetti S., Koopmans L. V. E., Bolton A., Treu T., Gavazzi R., 2010, MNRAS, 1510
Wambsganss J., Cen R., Ostriker J. P., 1998, ApJ, 494, 29

APPENDIX A: MAGNIFICATION MATRIX FROM DEFLECTIONS

In this appendix, it is shown how the magnification matrix can be estimated from three points on the image plane and their corresponding points on the source plane. This is useful for avoiding the direct calculation of the lensing quantities with the deflection solver to high precision.

The magnification matrix is the Jacobian matrix of the map between the source and the image planes,

$$\mathbf{A} = \left[\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}}\right] = \begin{pmatrix} 1-\kappa-\gamma_1 & -\gamma_2 \\ -\gamma_2 & 1-\kappa+\gamma_1 \end{pmatrix} \tag{A1}$$

$$= \begin{pmatrix} a_{11} & a_{21} \\ a_{12} & a_{22} \end{pmatrix}, \tag{A2}$$

where κ, γ_1 and γ_2 are the lensing quantities referred to in equations (7) through (9). For a single plane lens $a_{12} = a_{21}$ as implied in (A1), but in general this may not be the case because of numerical noise or multiplane lensing (see Paper II) and we will not impose this restriction here.

If we have three points on the image plane ($\boldsymbol{x}^{(0)}, \boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$) and the corresponding source points ($\boldsymbol{y}^{(0)}$, $\boldsymbol{y}^{(1)}$ and $\boldsymbol{y}^{(2)}$), we can define the vectors $\Delta\boldsymbol{x}^{(1,2)} = \boldsymbol{x}^{(1,2)} - \boldsymbol{x}^{(0)}$ and $\Delta\boldsymbol{y}^{(1,2)} = \boldsymbol{y}^{(1,2)} - \boldsymbol{y}^{(0)}$. To linear order in an expansion of the lensing deflection field $\Delta\boldsymbol{y}^{(1,2)} = \mathbf{A}\Delta\boldsymbol{x}^{(1,2)}$. These equations can be rewritten in matrix form as

$$\begin{pmatrix} \Delta y_1^{(1)} \\ \Delta y_2^{(1)} \\ \Delta y_1^{(2)} \\ \Delta y_2^{(2)} \end{pmatrix} = \begin{pmatrix} \Delta x_1^{(1)} & \Delta x_2^{(1)} & 0 & 0 \\ 0 & 0 & \Delta x_1^{(1)} & \Delta x_2^{(1)} \\ \Delta x_1^{(2)} & \Delta x_2^{(2)} & 0 & 0 \\ 0 & 0 & \Delta x_1^{(2)} & \Delta x_2^{(2)} \end{pmatrix} \begin{pmatrix} a_{11} \\ a_{21} \\ a_{12} \\ a_{22} \end{pmatrix}. \tag{A3}$$

As long as the three image points are not colinear the matrix can be inverted to solve for the elements of the magnification matrix. The result is

$$a_{11} = \left(x_2^{(2)} y_1^{(1)} - x_2^{(1)} y_1^{(2)}\right)/\det \tag{A4}$$

$$a_{22} = \left(x_1^{(1)} y_2^{(2)} - x_1^{(2)} y_2^{(1)}\right)/\det \tag{A5}$$

$$a_{12} = \left(x_2^{(2)} y_2^{(1)} - x_2^{(1)} y_2^{(2)}\right)/\det \tag{A6}$$

$$a_{12} = \left(x_1^{(1)} y_1^{(2)} - x_1^{(2)} y_1^{(1)}\right)/\det \tag{A7}$$

with

$$\det = x_1^{(1)} x_2^{(2)} - x_2^{(1)} x_1^{(2)}. \tag{A8}$$

This is clearly very fast to implement in a computer.

A test for uniform magnification over a grid cell is performed by taking all pairs of neighbours to that cell that are not colinear with the centre of the cell and calculating $\mathbf{A}$ by this method and then comparing each element of $\mathbf{A}$ for each pair of points. If none of the elements differ by more than the tolerance level the magnification matrix is considered to be uniform over the cell.

APPENDIX B: SINGULAR ISOTHERMAL SPHERE

Here, we summarize some standard analytic results for a SIS lens for reference.

The lensing equation in this case is

$$\boldsymbol{y} = \boldsymbol{x} - r_{\rm E}\frac{\boldsymbol{x}}{|\boldsymbol{x}|}. \tag{B1}$$

where $r_{\rm E}$ is the Einstein radius

$$r_{\rm E} = 4\pi\left(\frac{\sigma}{c}\right)^2\frac{\mathcal{D}_{\rm sl}\mathcal{D}_{\rm l}}{\mathcal{D}_{\rm s}} \tag{B2}$$

and σ is the velocity dispersion. For a point source with $|\boldsymbol{y}| < r_{\rm E}$ there are two images. One has positive magnification and one has a negative magnification (negative parity). Their positions are

$$\boldsymbol{x}_{\pm} = \boldsymbol{y}\left(1 \pm \frac{r_{\rm E}}{|\boldsymbol{y}|}\right). \tag{B3}$$

Only the positive magnification image exists if $|\boldsymbol{y}| > r_{\rm E}$. The magnifications of the images are

$$\mu_{+}(\boldsymbol{y}) = 1 + \frac{r_{\rm E}}{|\boldsymbol{y}|}. \tag{B4}$$

$$\mu_{-}(\boldsymbol{y}) = \begin{cases} 1 - \frac{r_{\rm E}}{|\boldsymbol{y}|}, & |\boldsymbol{y}| < r_{\rm E} \\ 0, & |\boldsymbol{y}| > r_{\rm E} \end{cases}. \tag{B5}$$

The magnification of a finite size source with surface brightness $f(\boldsymbol{y})$ is

$$\mu_{\pm} = \frac{1}{F}\int {\rm d}^2 y \mu(\boldsymbol{y}) f(\boldsymbol{y}), \tag{B6}$$

where F is the integral of $f(\boldsymbol{y})$ over all $\boldsymbol{y}$. For a circular, constant surface brightness source this integral can be done numerically. When the source intersects with the centre of the lens, the negative and positive images will be joined creating an Einstein ring. In Section 5.2, the joined images are considered to be a single positive magnification image since this is how GLAMER classifies them.

This paper has been typeset from a TEX/LATEX file prepared by the author.