Inverse Ray Shooting Tutorial (II)

Jorge Jiménez Vicente
Dpto. Física Teórica y del Cosmos
Universidad de Granada
Spain
Session II

• Playing around with lenses and sources
  – Two point lens
  – Chang Refsdal Lens
  – SIS (+ Shear)
  – NonSIS (+ Shear)
  – SIE (+ Shear)

• Critical Curves and Caustics

• Magnification maps
Today’s goal #1
Today’s goal #2
Today’s goal #3
Today’s goal #4
Play around with lenses/sources

• For the first part of the session we will play around with different combinations of lenses/sources.
• Pay attention to number of images, location, magnification, ...
Lenses

• Let’s try:
  – Binary point source lens
    \[ \alpha x = m_1(x-x_1)/d_1^2 + m_2(x-x_2)/d_2^2 \]
  – Point lens + shear (+ kappa)
    \[ \alpha x = (\kappa + \Upsilon)x + m_1(x-x_1)/d^2 \]
  – SIS (+ shear)
    \[ \alpha x = k(x-x_1)/d(\Upsilon x) \]
  – NonSIS (+ shear)
    Substitute \( d \) by \( \sqrt{(x-x_l)^2+(y-y_l)^2+rc^2} \)
  – SIE(+shear)
    \[ c_1=1-e, c_2=1+e \]
    \[ d=\sqrt{c_1(x-x_l)^2+c_2(y-y_l)^2} \]
    \[ \alpha x = k(x-x_1)/d \]
Sources

• We will try:
  – 2D Circular Gaussian
  – Face on disk galaxy \(\leftarrow\) From fits file \(\leftarrow\) pyfits
  – Edge on disk galaxy
  – Field of galaxies
  – Whatever takes you fancy
Example: Binary

Schneider & Weiss (1986)
Tests

• Try to produce:
  – 1 image
  – 2 images
  – 3 images
  – 4 images
  – 5 images
  – Many (micro-)images
  – Arcs
  – Reproduce your favorite lens system..
  – ....
Critical curves and caustics

- A critical curve is the set of points at the image plane for which \( \det(A) = 0 \).
- Caustic curve is the set of points at the source plane with infinite magnification. The source locations whose images are the critical curves.
- We calculate \( A \) from derivatives of the deflection angle and then its determinant.
- Therefore we will:
  - Calculate \( A \) from derivatives of the deflection angle.
  - Calculate \( \det(A) \)
  - Locate the places with \( \det(A) = 0 \) \( \Rightarrow \) Critical curves (**auxfun.levloc**)
  - We trace back those rays to the source plane \( \Rightarrow \) Caustics
- It's a bit tricky because of topological properties around the critical curve/caustic
- We may use **lens.py** for the lenses from now onwards.
Magnification maps I

• To calculate magnification maps we will use the fact that:
  \( \mu = \frac{d\Omega_i}{d\Omega_s} = \frac{dS_i}{dS_s} = \frac{N_{\text{hits}}}{N_{\text{rays}}} \)
• To calculate this we will:
  – Divide the image plane into cells from which we will throw raypix rays per unlensed pixel.
  – Throw the rays backwards from the image/lens plane towards the source plane by deflecting them according to the lens equation.

  We can stop here for a while and have a look at the source plane

  – Collect hits at every pixel of the source plane
  – Compare (divide) to how many rays would have hit in the absence of lensing.
Magnification Maps II

- Shooting rays one at a time needs a nested loop → Python becomes slow.
- Shoot rays one row at a time to speed up calculations.
- Throwing the whole array at once is in principle possible, but will make it very memory demanding with high risk of crash.
- What happens if the throwing region is too small?
- Try magnification maps for a few lens configurations:
  - Point mass
  - Binary
  - N point lenses
  - Chang-Refsdal
  - (Non)SIS (+ shear)
  - ...

11/11/2012
IAC-WS-2012 Nov 2012