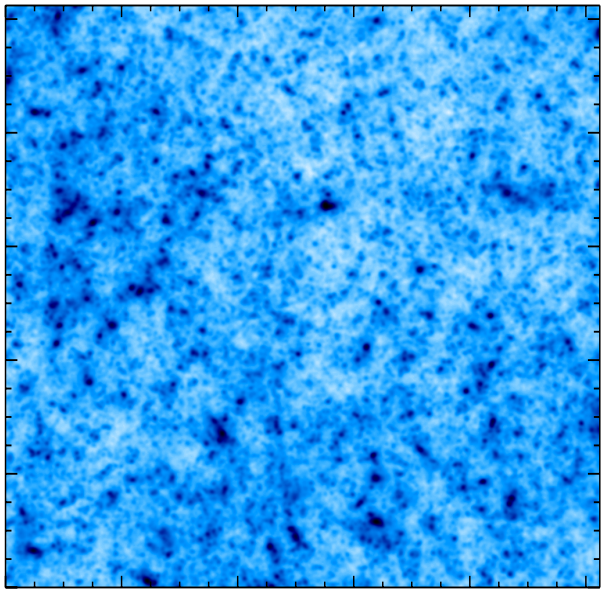
MRLENS : Multi-Resolution methods for gravitational LENSing

URL: http://www-dapnia.cea.fr/Phocea/Vie_des_labos/Ast/ast_visu.php?id_ast=878

S. Pires, J.L. Starck and A. Réfrégier

DAPNIA/SEDI-SAP, CEA/Saclay,
Orme des Merisiers
91191 Gif-sur-Yvette, France

Version 1.0



Contents

Acknowledgments

The authors of this software package would like to thank their colleagues who helped us in this project and specially would like to acknowledge : Joël Berge, Yassir Moudden, Richard Massey, Savita Mathur,...
And also we would like to thank Phil Marshall for his help relative to the use of the LensEnt2 package.

This package is a compilation of some algorithms and methods which were developed and/or used successfully in the applications reported in the 2 following publications:

- **Weak Lensing Mass Reconstruction using Wavelets**, J.-L. Starck, S. Pires and A. Réfrégier, *Astronomy and Astrophysics*, March 2005, available at: http://arxiv.org/abs/astro-ph/0503373

- **Sunyaev-Zeldovich cluster reconstruction in multiband bolometer camera surveys**, S. Pires, J.-B. Juin, D. Yvon, Y. Moudden, S. Anthoine and E. Pierpaoli, submitted to *Astronomy and Astrophysics*, August 2005, available at: http://arxiv.org/pdf/astro-ph/0508641

More than a software dedicated to a new reconstruction method, this package includes many other tools useful to process, analyze and visualize lensing data. In this first version, we only focus on Weak Gravitational Lensing; Strong Gravitational Lensing will be added in a forthcoming version.

These notes are the first version of the software package "Multi-Resolution methods for gravitational LENSing", for this reason some precisions may have been forgotten and it is surely still incomplete.

Chapter 1

Introduction

In the beginning of the twentieth century Albert Einstein derived his Theory of General Relativity. One of the consequences of this theory was that massive bodies could bend the path of light rays. One of the first confirmations of Einstein's new theory was the observation during the 1919 eclipse of the deflection of light from distant stars by the sun. The first confirmed gravitational lens has been discovered in (1979) by D. Walsh, R.F. Carswell & R.J. Weymann perchance while searching for counterparts of radio sources. In fact, they observed the first mutiple images, two images : 0957+561A and 0957+561B of the same object (a quasar) only separated by 6". See Fig. 1.1.

Figure 1.1: The first discovered lensing case, the double quasar 0957+561 (Walsh et al, 1979)

Since, a range of lensing phenomena have been discovered. These include multiply imaged quasars, radio rings, giant luminous arcs and arclets. These important observational advances drove theoretical efforts to exploit lensing as an astrophysical tool, establishing gravitational lensing as one of the most dynamic area of research in observational astronomy.

The gravitational deflection of light generated by mass concentrations along light paths produces magnification, multiplication, and distortion of images. It also delays photon propagation from one light of sight to another.These effect illustrated by Fig. 1.2 show-

ing the Abell 2218 cluster which is one of the strongest lens observed. Galaxy Cluster Abell 2218 is a massive cluster of galaxies some 2 billion light years away towards the constellation Draco. Fig. 1.3 shows another gravitational lens in the Abell 1689 cluster 2.2 billion light years distant toward the constellation Virgo. These gravitational arcs are actually the lensed and distorted images of galaxies that are around 10 times more distant than the cluster. By magnifying the images of distant background galaxies, gravitational lenses such as galaxy cluster Abell 2218 or Abell 1689 allow detailed views of very distant galaxies to be obtained.

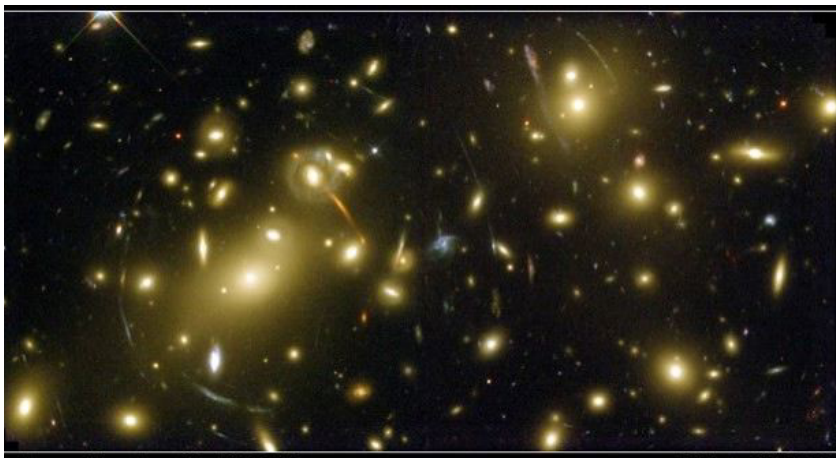


Figure 1.2: Strong Gravitational Lensing effect in the Abell 2218 cluster (W. Couch et al, 1975 - HST)

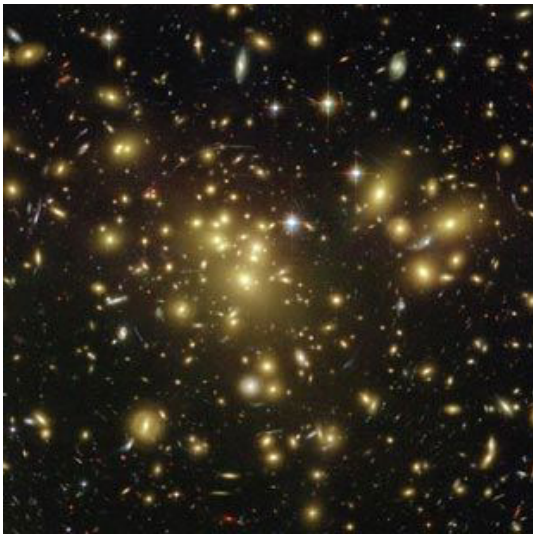


Figure 1.3: Strong Gravitational Lensing effect in the Abell 1689 cluster (N. Benitez et al, 2003 - HST)

The properties and the interpretation of this effect depend on the projected mass density integrated along the line of sight and on the cosmological angular distance between the observer, the lens and the source (see Fig. 1.4).

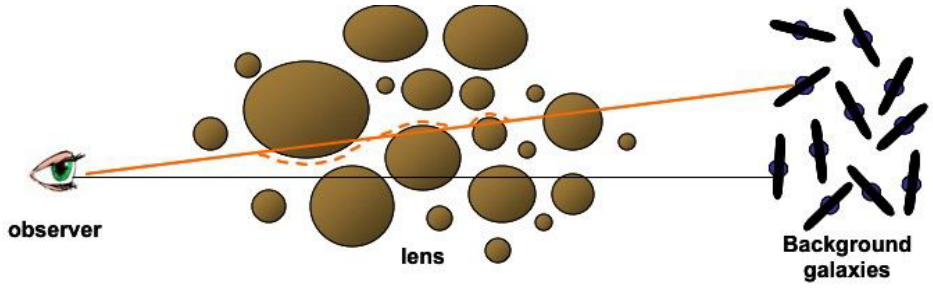


Figure 1.4: Gravitational Lensing effect

We can distinguish two regimes of Gravitational Lensing, in some cases (as seen previously) the bending of light is so extreme, that the light travels along two different paths to the observer, and multiple images of one single source appear on the sky. This effect is called strong lensing. In most cases, the lens is not strong enough to form multiple images or giant arcs. The background galaxies, however, are still distorted! They are stretched and magnified, but by small amounts. This is called "Weak Gravitational Lensing". Thanks to these effects, Gravitational lenses can provide crucial information on the geometry of the Universe, on its matter content and on the cosmological scenario of formation of its structures.

This first version of the package MRLENS (Multi-Resolution methods for gravitational LENSing) only focuses on Weak Gravitational Lensing. Nowadays Weak Lensing provides a unique method to directly map the distribution of dark matter in the universe. It relies on the measurement of the distortions that lensing induces in the images of background galaxies. For a review see (Refregier, 2003b; Mellier, 1999; Bartelmann and Schneider, 1999). Unlike other methods that probe the distribution of light, Weak Gravitational Lensing measures the mass and can thus be directly compared to reliable theoretical models of structure formation.

Ongoing efforts are made to improve the detection of cosmic shear on existing telescopes and future instruments are planned. Several methods are used to derive the lensing shear from the shapes of background galaxies. But the shear map obtained is always noisy, and when it is converted into a map of the projected mass κ, the result is dominated by the noise. To succeed in probing the Universe with Weak Lensing a sophisticated algorithm is required to filter the mass map κ. Some basic algorithms are commonly used like Gaussian filtering or Wiener filtering but the result is not optimal.

The MRLENS package offers a new algorithm for the reconstruction of weak lensing mass maps (Starck et al., 2005; Pires et al., 2005) . This new method uses the Multiscale Entropy concept (which is based on wavelets) and the False Discovery Rate (FDR) which allows us to derive robust detection levels in wavelet space. Many other tools useful to process lensing shears from real or simulated data are available in the MRLENS software.

In this user manual, we first introduce the weak gravitational lensing reconstruction problem in chapter 2. Chapter 3 describes the simulated data, we worked with. In Chapter 4, some earlier basic methods for the reconstruction of weak lensing mass maps are briefly described. And Chapter 5 is dedicated to the description of our new method. Some results are presented in Chapter 6. An accurate description of the IDL routines that makeup this package is given in Chapter 7.

Chapter 2

Weak Gravitational Lensing

2.1 Introduction

Weak lensing has now been convincingly detected (Bacon et al., 2000; Kaiser et al., 2000; Van Waerbeke et al., 2000; Wittman et al., 2000) and has proven to be a powerful tool in observational cosmology, because it provides a probe of the dark matter distribution. The measurement of the distortions that lensing induces in the images of background galaxies, enables a direct measurement of the large-scale structures in the universe.

Nowadays, many efforts are made to improve the detection of the weak lensing effect, because its effect is very small. Large surveys are planned and several telescopes have been built or dedicated to the activity (Subaru Telescope, CFHT and its Megacam Camera,...)

2.2 The gravitational lensing effect

We can try to find an equivalent in classical optics to Gravitational lensing. Move away immediately the idea to represent it with the simple convex lens because the more the ray paths are closed to the optical axis, the less they are deviated, as we can seen in Fig. 2.1. On the contrary, a Gravitational lens acts totally in an opposite way. Indeed, the more the light beam passes close to the deflecting mass, the stronger the deflection is.

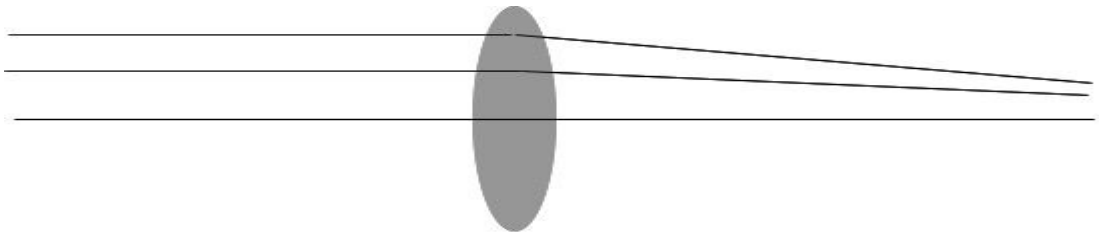


Figure 2.1: Light deflection by a thin convex lens

A good equivalent could be a foot of wine glass, see as sketched on Fig. 2.2.

We can try the experiment to observe a source (for example a dot in a paper) through an upside down wine glass, depending on the distance to the glass axis, we can see one or

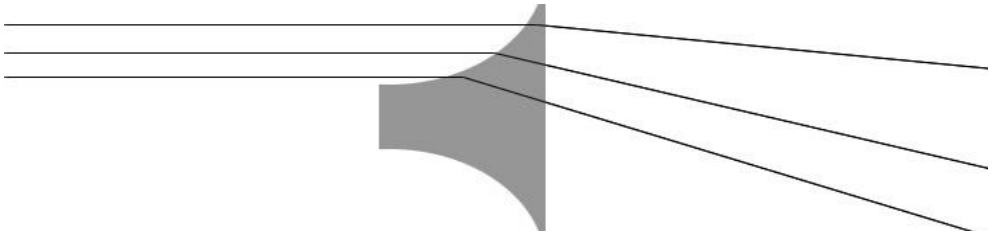


Figure 2.2: Light deflection by a foot of glass

two distorted images. The gravitational lensing effect is obviously more tortuous because the bending of light is linked to the matter distribution integrated along the line of sight.

2.3 The distortion measurement

The measurement of the weak lensing signal is difficult, but not impossible, as has been demonstrated in the last few years. The applications of weak lensing are numerous. The technique has been applied to clusters of galaxies and groups of galaxies. More recent studies concentrate on lensing by large scale structures. And then it has been used to constrain cosmological parameters, and to study the relation between galaxies and dark matter.

For these applications, we need to measure the distortions of galaxies accurately. we measure the stretching or the compression along x axis, termed γ_1 and in the same way, we measure the stretching or the compression along the 45° to x axis, called γ_2. See the Fig. 2.3. Thanks to γ_1 and γ_2, we can derive the shear maps.

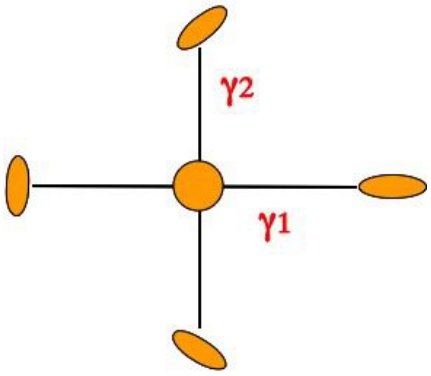


Figure 2.3: The distortion measurement

An essential aspect of weak gravitational lensing is that measurements of its effects are statistical. Indeed, galaxies have a shape of their own, and the change in the shape of an individual galaxy caused by weak lensing is too small to be useful. Generally, galaxies that are closen to each other on the sky experience similar deflections. We can therefore average over the shapes of many galaxies.

If all background galaxies would be intrinsically round, gravitational lensing would transform them into ellipses with the orientation of the local gravitational shear. In reality, galaxies can intrinsically have some shape and orientation. Usually the intrinsic shape is described by an ellipse. Since the orientation of the intrinsic ellipticities is random, each observed image ellipticity provides an unbiased but very noisy estimate of the lensing shear.

Then, several methods are used to derive the lensing shear from the shapes of background galaxies. One emergent method based upon the shapes, seems to be ideal to deal with this problem, it is the Shapelets formalism. This method is based on the decomposition of a galaxy background image into elementary shapes, that is to say in a dictionnary of shapes. For more details see (Refregier, 2003a; Refregier and Bacon, 2003).

2.4 PSF correction

The weak lensing effects are quite subtle, or weak, and many of the current challenges in the field are in the PSF (Point Spread Function) correction due to the camera, the response of the telescope and the atmosphere. In fact, the real image is convolved by a transfer function due on the one hand to the limited size of the telescope and on the other hand, to the atmospheric turbulence (the seeing). And we have to add the CCD camera noise (nearly Poisson noise) due to the finite number of photons per pixel.

Because of the imperfection of imaging properties in the telescope (tracking errors, wind shake, or other reasons) the PSF is not necessarily circularly symmetric. Even a circular image would then appear elongated if observed through an isotropic PSF. Then, an isotropic PSF mimics shear effect. Actually, the PSF has an 8-10% effect and the weak lensing effect in the observed ellipticities of galaxies, called "cosmic shear", is expected to be 1-3 % effect. Consequently if we want to measure the shear, we need to correct background images for the PSF. This remains a major problem in weak lensing (see (Berge, 2005) for more informations).

The MRGL software doesn't try to solve this problem. We deal either with the lensing shear corrected; a simple diagnostic test is available to look for a wide range of systematic errors (see subsection §2.7) or directly with the mass map assuming that there are no systematic errors.

2.5 Weak Lensing Shear

As we have seen previously, in weak lensing surveys, the shear $\gamma_i(\theta)$ with $i = 1, 2$ is derived from the shapes of galaxies at positions θ in the image. The shear field $\gamma_i(\theta)$ can be written in terms of the lensing potential $\psi(\theta)$ as (see eg. (Bartelmann and Schneider, 1999))

$$\begin{aligned} \gamma_1 &= \frac{1}{2}\left(\partial_1^2 - \partial_2^2\right)\psi \\ \gamma_2 &= \partial_1\partial_2\psi, \end{aligned} \tag{2.1}$$

where the partial derivatives ∂_i are with respect to θ_i. The convergence $\kappa(\theta)$ can also be expressed in terms of the lensing potential as

$$\kappa = \frac{1}{2}\left(\partial_1^2 + \partial_2^2\right)\psi \tag{2.2}$$

and is related to the surface density $\Sigma(\theta)$ projected along the line of sight by

$$\kappa(\theta) = \frac{\Sigma(\theta)}{\Sigma_{\rm crit}} \tag{2.3}$$

where the critical surface density is given by

$$\Sigma_{\rm crit} = \frac{c^2}{4\pi G}\frac{D_s}{D_l D_{ls}} \tag{2.4}$$

and G is Newton's constant, c is the speed of light and D_s, D_l and D_{ls} are the angular-diameter distances between the observer and the galaxies, the observer and the lens, and the lens and the galaxies, see Fig.1.4. In practice, the galaxies are not at a fixed redshift, and the expression for κ is an average over the redshift of the galaxies (see eg. (Bartelmann, 1995)). The lensing gravitational effect is said to be weak or strong if $\kappa \ll 1$ or $\kappa \gtrsim 1$, respectively.

The left panel of Fig. 2.4 shows a simulated convergence map derived from ray-tracing through N-body cosmological simulations performed by (Vale and White, 2003). The simulation contains 512^3 particles with a box size of $300h^{-1}$ Mpc. The resulting convergence map covers 2×2 degrees with 1024×1024 pixels and an assumed galaxy redshift of 1. The overdensities correspond to the haloes of groups and clusters of galaxies. The rms value of κ binned in 0.12 arcmin pixels is $\sigma_\kappa = 0.023$. The typical values of κ are thus of the order of a few percent, apart from the core of massive halos (see figure 2.4). The weak lensing condition therefore holds in most regions of the sky and will be assumed throughout this handbook.

2.6 The Mass inversion problem

The weak lensing mass inversion problem consists in reconstructing the projected (normalised) mass distribution $\kappa(\theta)$ from the measured shear field $\gamma_i(\theta)$ by inverting equations (2.1) and (2.2). For this purpose, we take the Fourier transform of these equations and obtain

$$\hat{\gamma}_i = \hat{P}_i\hat{\kappa}, \quad i = 1, 2 \tag{2.5}$$

where the hat symbol denotes Fourier transforms and we have defined $k^2 \equiv k_1^2 + k_2^2$ and

$$\begin{aligned} \hat{P}_1(\mathbf{k}) &= \frac{k_1^2 - k_2^2}{k^2} \\ \hat{P}_2(\mathbf{k}) &= \frac{2k_1k_2}{k^2}, \end{aligned} \tag{2.6}$$

with $\hat{P}_1(k_1, k_2) \equiv 0$ when $k_1^2 = k_2^2$, and $\hat{P}_2(k_1, k_2) \equiv 0$ when $k_1 = 0$ or $k_2 = 0$.

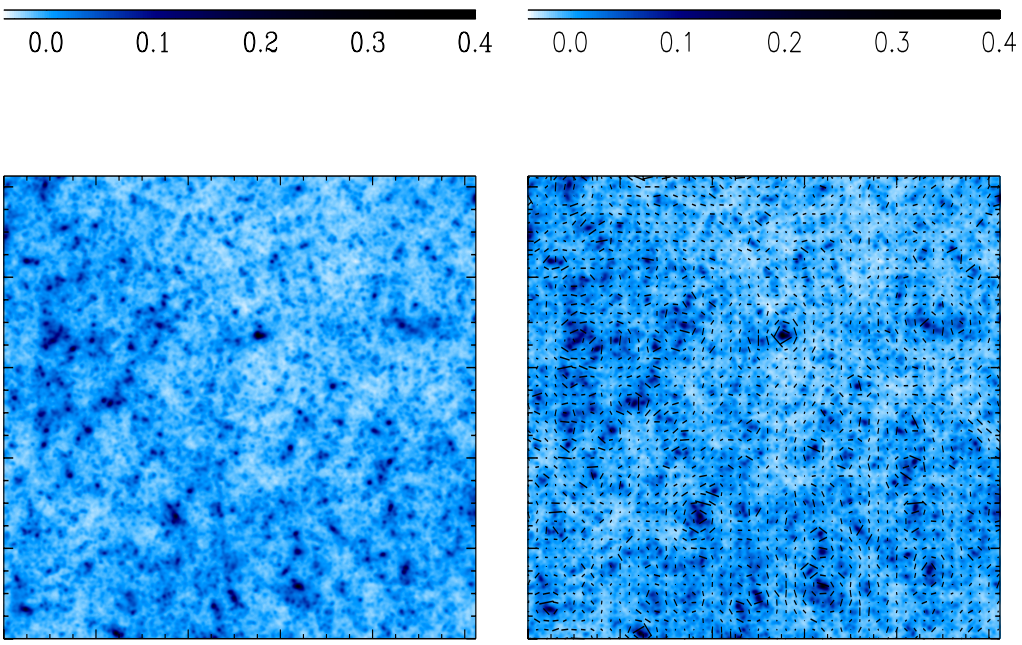


Figure 2.4: Left: simulated convergence map from (Vale and White, 2003) for a ΛCDM model. The region shown is 2×2 square degree. Right: Shear map superimposed on the convergence map. The size and direction of each line gives the amplitude and position angle of the shear at this location on the sky.

The shear map γ_i can be calculated from the convergence map κ using these expressions. The right panel of Fig.2.4, shows the shear field associated with the simulated convergence field. As is customary, the direction and size of the line segment represent the orientation and amplitude of the shear. The rms shear in the resulting map is $\sigma_\gamma = 0.0124$ ($\sigma_{\gamma_1} = 0.01658$ and $\sigma_{\gamma_2} = 0.01601$).

Note that to recover κ from γ_1 (resp. γ_2), there is a degeneracy when $k_1^2 = k_2^2$ (resp. when $k_1 = 0$ or $k_2 = 0$). To recover κ from both γ_1 and γ_2, there is a degeneracy only when $k_1 = k_2 = 0$. Therefore, the mean value of κ cannot be recovered from the shear maps. This is a special instance of the well known mass-sheet degeneracy in the weak lensing reconstruction if only shear information is available (see eg.(Bartelmann, 1995) for a discussion).

In practice, the observed shear γ_i is obtained by averaging over a finite number of galaxies and is therefore noisy. The relations between the observed data γ_{1b}, γ_{2b} binned in pixels of area A and the true mass map κ are given by:

$$\gamma_{ib} = P_i * \kappa + N_i \tag{2.7}$$

where N_1 and N_2 are noise contributions with zero mean and standard deviation $\sigma_n \simeq \sigma_\epsilon/\sqrt{N_g}$, where $N_g = n_g A$ is the average number of galaxies in a pixel and n_g is the average number of galaxies per arcmin^2. The rms shear dispersion per galaxy σ_ϵ arises both from measurement errors and the intrinsic shape dispersion of galaxies. In this analysis, we will assume $\sigma_\epsilon \simeq 0.3$ as is approximately found for ground-based and space-

based weak lensing surveys. From the central limit theorem, this means that for pixels with $A \gtrsim 1$ amin^2, the noise N_i is, in a good approximation, Gaussian in both cases and is uncorrelated.

2.7 The E and B inverse Filters

We can easily derive an estimation of the mass map by inverse filtering by noticing that

$$\hat{P_1}^2 + \hat{P_2}^2 = 1. \tag{2.8}$$

The least square estimator $\hat{\tilde{\kappa}}_b^{(E)}$ of the convergence $\hat{\kappa}$ in the Fourier domain is defined by :

$$\hat{\tilde{\kappa}}_b^{(E)} = \hat{P}_1\hat{\gamma}_{1b} + \hat{P}_2\hat{\gamma}_{2b} \tag{2.9}$$

The relation between this estimator and the true mass map is $\hat{\tilde{\kappa}}_b^{(E)} = \hat{\kappa} + \hat{N}$, where $\hat{N} = \hat{P}_1\hat{N}_1 + \hat{P}_2\hat{N}_2$.

Just as any vector field, the shear field $\gamma_i(\theta)$ can be decomposed into a gradient, or electric (E), component and a curl, or magnetic (B) component. Because the weak lensing arises from a scalar potential (the newtonian potential), it can be shown that weak lensing only produces E-modes. On the other hand, residual systematics may arise from imperfect correction of the intrumental PSF or telescope aberrations which generally generates both E and B modes. The presence of B-modes is thus used to test for the presence of uncorrected systematic effects in current weak lensing surveys.

The decomposition of the shear field into each of these components can be easily performed by noticing that a pure E-mode can be transformed into a pure B mode by a rotation of the shear by 45°: $\gamma_1 \to -\gamma_2$, $\gamma_2 \to \gamma_1$. As a result, we can form the following estimator for the B-mode "convergence" field

$$\hat{\tilde{\kappa}}_b^{(B)} = \hat{P}_2\hat{\gamma_{1b}} - \hat{P}_1 * \hat{\gamma_{2b}}, \tag{2.10}$$

and check that it is consistent with zero in the absence of systematics.

As follows from equation 2.8, the noise $N^{(E)}$ and $N^{(B)}$ in $\hat{\tilde{\kappa}}_b^{(E)}$ and $\hat{\tilde{\kappa}}_b^{(B)}$ is still Gaussian and uncorrelated.

The inverse filtering does not amplify the noise, but $\hat{\tilde{\kappa}}_b^{(E)}$ and $\hat{\tilde{\kappa}}_b^{(B)}$ may be dominated by the noise if $N^{(E)}$ and $N^{(B)}$ are large, which is the case in practice. Fig. 2.5 shows the reconstructed mass map using equation 2.9 when a realistic Gaussian noise has been added to the shear maps plotted on the right of Fig. 2.4 using vectors. As expected, it is dominated by noise.

To use weak lensing as a dark matter probe, we need to remove the noise in weak lensing maps. This point has motivated the use of different methods of filtering in the past which we describe in chapter 4 and this same desire has led to the development of a sophisticated method that we describe in chapter 5.

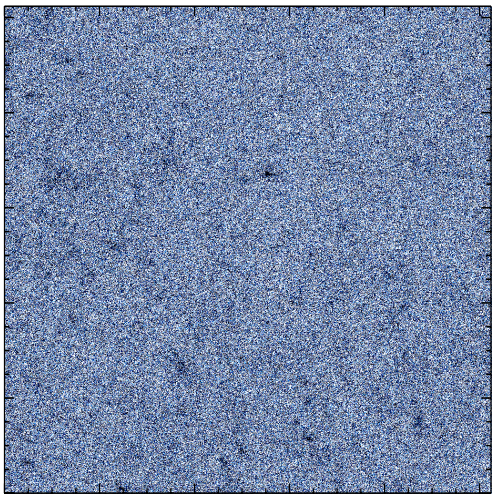


Figure 2.5: Noisy mass map $\kappa_b^{(E)}$ for the same simulation with $n_g = 100$ gal/arcmin^2, corresponding to space-based observations. Even in this case, the unfiltered mass map is dominated by noise.

Chapter 3

Data sets for Weak Gravitational Lensing

3.1 Simulated Data

3.1.1 Data Set characteristics

We have at our disposal some simulated convergence maps derived from ray-tracing through N-body cosmological simulations performed by (Vale and White, 2003). See also, the website where the data are available : http://mwhite.berkeley.edu/Lensing/ValeWhite/

The cosmological model is taken to be a concordance ΛCDM model with parameters :

Model	Box	OmM	OmL	h	s8
B	300	0.3	0.7	0.7	0.8

where the box side is in Mpc/h. The simulations have 512^3 particles. The particles have equal masses M $\sim$ 1.7e10 Msun/h.

The overdensities correspond to the halos of groups and clusters of galaxies. The typical values of κ are thus of the order of a few percent, apart from the core of massive halos.

3.1.2 Simulated Data description

The directory "Data" contains simulated weak lensing maps : the convergence map, the shear maps and the shear catalogs derived from simulations of the formation and evolution of large-scale structures. In all cases the fields are 2 x 2 degrees, downsampled to 1024^2 pixels and assume that the sources lie at exactly z=1.

Simulated Mass Map

The folder "Kappa_B" contains ten simulated lensing maps. The ten lensing maps are semi-independent using the multi-plane ray tracing algorithm (Vale and White, 2003). The convergence (mass) map is contained in kappa_B_?.fits for models B with ? a number in the range 0-9.

Simulated Shear Maps

Similarly, folder "Gamma_B" contains ten simulated shear maps. The two shear components are contained in gamma_B_?.fits.

Simulated Shear Catalogs

The folder "GCat" contains twenty shear structure catalogs corrupted by noise with the following fields :
- x, y = (1D IDL array) coordinates in pixel of each galaxy
- pixscale = (int) pixel size [in rad]
- weight = (1D IDL array) weight of each galaxy
- gamma1, gamma2 = (1D IDL array) shear γ_1 and γ_2 of each galaxy

The noisy shear catalogs are contained in gcat_?_b1.fits for simulated ground observations and gcat_?_b2.fits for simulated space observations. In weak lensing, the average number of galaxies per arcmin^2 n_g is equal typically to :

- $n_g = 20$ gal/arcmin^2 for ground-based surveys.
- $n_g = 100$ gal/arcmin^2 for space-based surveys.

Simulated Noisy Shear maps : From simulated noiseless shear maps to noisy shear maps

As we have seen in the previous chapter, the observed shear γ_i are obtained by averaging over a finite number of galaxies and consequently this measurement is corrupted by noise.

In order to simulate the shear maps γ_{1b} and γ_{2b} obtained in real observations , we have added a Gaussian noise to the simulated shear maps as followed :

$$\gamma_{1b} = \gamma_1 + N_1 \tag{3.1}$$
$$\gamma_{2b} = \gamma_2 + N_2 \tag{3.2}$$

where N_1 and N_2 are noise contributions with zero mean and standard deviation $\sigma_n \simeq \sigma_\epsilon/\sqrt{N_g}$, where $N_g = n_g A$ is the average number of galaxies in a pixel, σ_ϵ is due both to measurement errors and to the intrinsic shape of galaxies and n_g is the average number of galaxies per arcmin^2.

The folder "Gamma_Noise" contains the noisy shear maps obtained. Then, the two noisy shear components are contained in gamma_B_?_b1.fits for simulated ground observations and gamma_B_?_b2.fits for simulated space observations.

Simulated Noisy Mass Maps : From simulated noisy shear maps to the noisy mass map

Then, we use the following relation to derive the κ_b map from the γ_{1b}, γ_{2b} maps :

$$\hat{\kappa_b} = \hat{P_1}\hat{\gamma_{1b}} + \hat{P_2}\hat{\gamma_{2b}} \tag{3.3}$$

Similarly, the noisy convergence maps obtained are contained in kappa_B_?_b1.fits for simulated ground observations and kappa_B_?_b2.fits for simulated space observations in the Kappa_noise folder.

Simulated Shear Maps with an Electric and a Magnetic component

The folder "Gamma_Magnetic" contains an example of noiseless shear maps with both E and B modes.

The decomposition of the shear field into each of these components can be easily performed. Weak Lensing only produces E-modes but systematic effects can produce both E and B modes. Thus, the presence of B-modes is used to trace out the systematic errors.

3.2 Missing Data

3.2.1 Presentation of the problem

Sometimes during the observations, an incident can cause a loss of data in the image. For instance, this can be due to a defect of the CCD camera, generating a dark line or a dark row in the image, or to the presence of a very bright star in the field of vision which forces us to remove this part of the image. In order to model this problem, all the pixels in some arbitrary square shaped region are set to zero in the shear maps γ_1 and γ_2. By inverse filtering, we have derived the noisy mass map κ_b in which we can also visualize the lack of data (Fig. 3.1 - left panel).

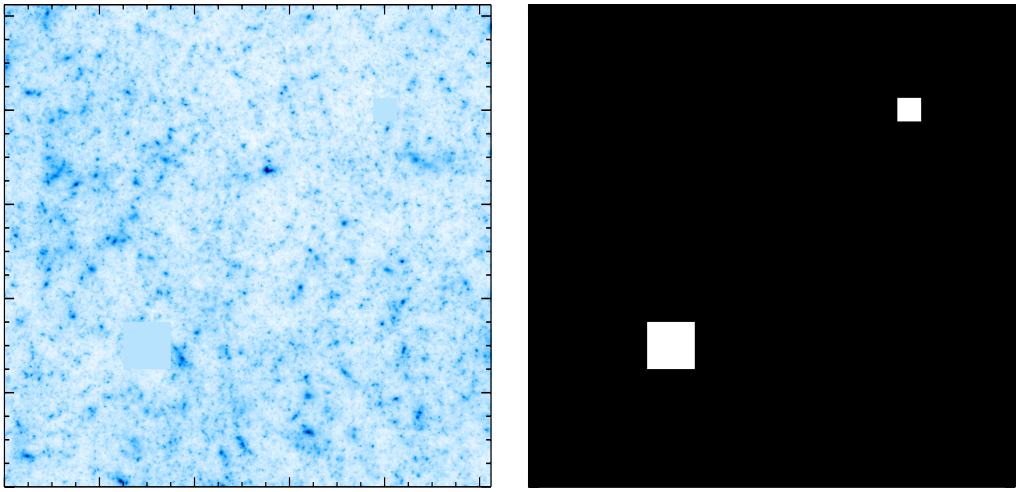


Figure 3.1: Mass map with missing data (left) and its mask (right).

3.2.2 How to overcome this problem ?

Filtering in Fourier space

In the presence of a hole, we need to be very careful because the missing data doesn't contain any information. For this reason, a data mask is required during a process like Wiener filtering to compute the Wiener weight function. To build a data mask, we need just to set to 1 all the pixels in the hole and to set to zero all the others pixels. Fig. 3.1 - right panel, shows the mask of the previous mass map with missing data 3.1 - left

panel. Fig. 3.2 shows an example of mask obtained for real data, the field is 0.5 x 0.5 square degrees. We can notice that some bright stars have been removed. And during the filtering, we just consider the pixels outside the mask.

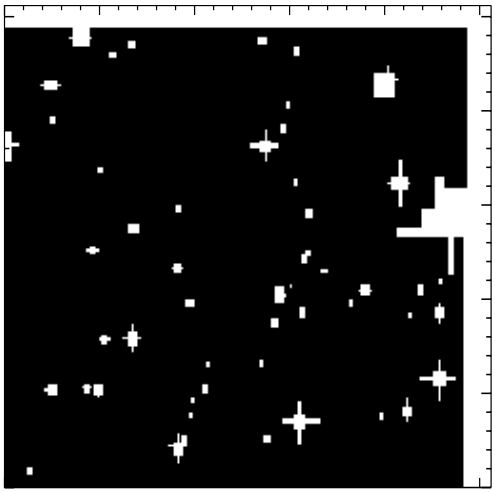


Figure 3.2: An example of real data mask, the field is 0.5 x 0.5 square degrees

Filtering in Wavelet space

In some cases of multi-resolution filtering, a multiscale data mask can be required to remove all the impact of the hole. Thus, we derive a multi-resolution mask from one scale to another, in dilating the data mask of the previous scale, see Fig.3.3 . All the pixels, in the mask are or can be distorted by the presence of the hole or by an edge effect.

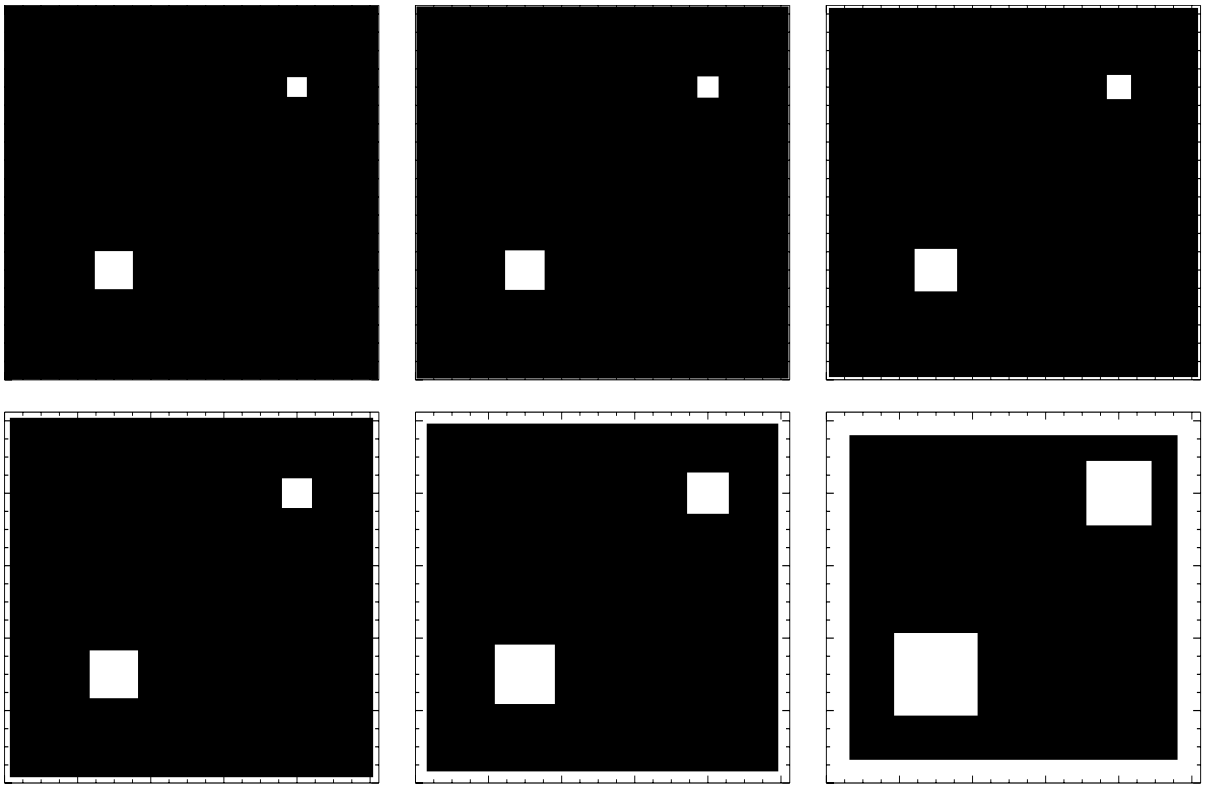


Figure 3.3: Multi-resolution mask of the previous mass map with some missing data

Chapter 4

Filtering Methods

4.1 Gaussian Filtering

The standard method (Kaiser and Squires, 1993) consists in convolving the noisy mass map κ_b with a Gaussian window G with standard deviation σ_G:

$$\kappa_G = G * \kappa_b \tag{4.1}$$

The quality of the resulting estimation depends strongly on the value of σ_G.

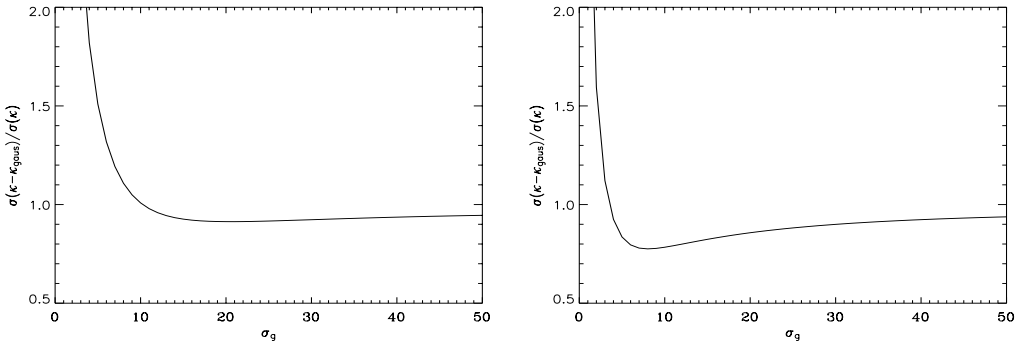


Figure 4.1: Reconstruction error as a function of the kernel size σ_G for the Gaussian smoothing method, with $n_g = 20$ gal/amin^2 (left) and $n_g = 100$ gal/amin^2 (right).

Fig. 4.1 shows the variation of the error between the original mass map κ shown in Fig. 2.4 and the filtered mass map κ_G. For this simulation, the optimal value of σ_G lies between 5 and 10 pixels (1 pixel = 0.12 arcmin) for space observations (i.e. $n_g = 100$ gal/amin^2) and lies between 20 and 25 pixels for ground observations (i.e. $n_g = 20$ gal/amin^2).

We have applied the Gaussian Filtering to the two noisy mass maps simulating the ground and space observations and the results are shown Fig. 4.2. According to the value of σ_G used the overdensities are more or less smoothed and we have a loss of resolution. Then, it is difficult to attribute any significance to these structures.

An alternative to Gaussian filtering is Wiener filtering.

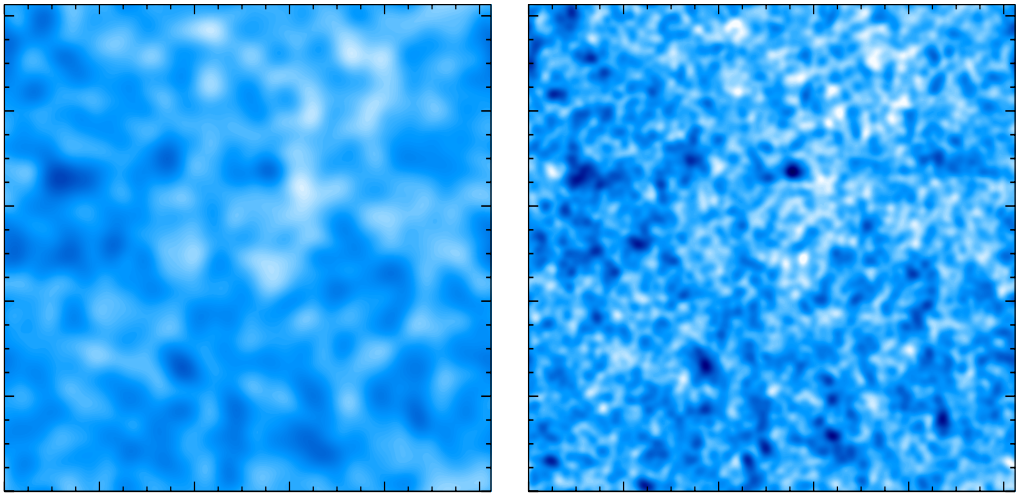


Figure 4.2: Gaussian filtering: left with $\sigma_G = 2.5$ amin for $n_g = 20$ gals amin^{-2} and right with $\sigma_G = 1$ amin for $n_g = 100$ gals amin^{-2}.

4.2 Wiener Filtering

Wiener filtering is a method that attempts to minimize the mean squared error between the original and the restored signal.

The method used here consists in calculating the variance of the signal and that of the noise on concentric rings increasing logarithmically in Fourier space and then in deriving a weight function and finally in convolving the observed map κ_{obs} with this weight function that is to say by assigning the following weight to each ring in Fourier Space:

$$\hat{w}(k) = \frac{< |\hat{\kappa}(u,v)|^2 >_k}{< |\hat{\kappa}(u,v)|^2 >_k + < |\hat{N}(u,v)|^2 >_k} \tag{4.2}$$

where k is the index of the rings, and $|\hat{\kappa}(u,v)|^2$ is a model of the map power spectrum and is in practice derived from the data. The weight function makes it possible to attenuate or to remove part of the frequencies if the signal-to-noise ratio is low. The filtering depends on the model of the noise. This Wiener filter is the optimal filter if both the signal and the noise are stationary and isotropic and well modeled as Gaussian Random Fields. As can be seen from Fig. 2.4 (left), this assumption is not valid for weak lensing mass maps which display non-Gaussian features such as galaxy clusters, groups and filaments. Nevertheless, Wiener filtering generally outperforms the simple Gaussian filtering.

For comparison, we have plotted (Fig. 4.3) the Wiener function and the Gaussian functions used in the previous Gaussian filtering for space observations, on the same graph.

The Gaussian filter nearest to the Wiener filter has a standard deviation lying between 5 and 10. This is in accordance with the previous observations.

Fig. 4.4, shows the results of the Wiener filtering in our data. The results seems

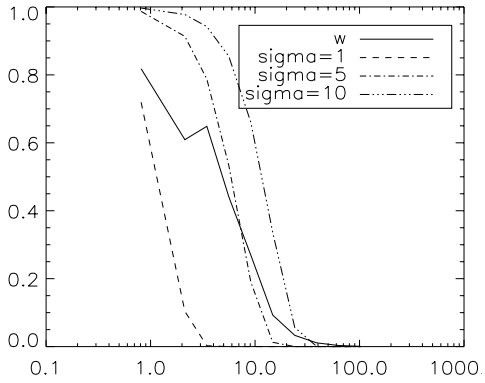


Figure 4.3: Comparison between the Gaussian filters and the Wiener filter ($n_g = 100$)

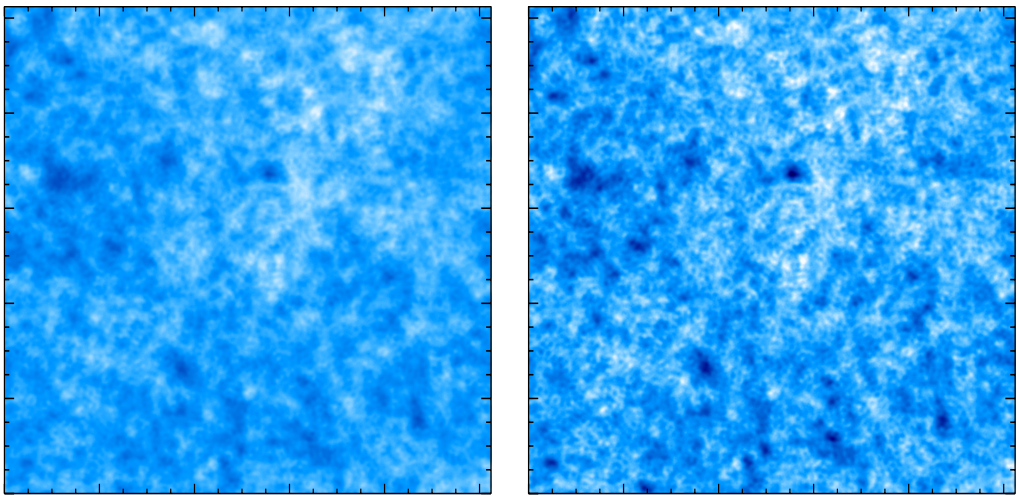


Figure 4.4: Wiener filtering: left for $n_g = 20$ gals amin^{-2} and right $n_g = 100$ gals amin^{-2}.

good but the texture reconstructed doesn't exist in the original image and among the reconstructed structures, some are false. Consequently, maybe we recover more clusters but with a bad level of confidence.

4.3 Maximun Entropy Method

The Maximum Entropy Method (MEM) is well-known and widely used in image analysis in astronomy (see (Bridle et al., 1998; Starck et al., 2001; Marshall et al., 2002; Starck and Murtagh, 2002) for a full description). It considers both the data and the solution as probability density functions and finds the solution using a Bayesian approach and adding a prior on the solution. Several definitions of entropy exist. The most common is the Gull and Skilling definition (1991):

$$H_g(\kappa) = \sum_x \sum_y \kappa(x,y) - m(x,y) - \kappa(x,y) \ln\left(\frac{\kappa(x,y)}{m(x,y)}\right) \tag{4.3}$$

where m is a model, chosen typically to be a sky background. H_g has a global maximum at $\kappa = m$. MEM does not allow negative values in the solution, which is a problem for data such weak shear data or the CMB data, where we measure fluctuations around zero. To overcome this problem, it has been proposed to replace H_g (Maisinger et al., 2004) by:

$$H_{+/-}(\kappa) = \sum_x \sum_y \psi(x,y) - 2m - \kappa(x,y) \ln \left(\frac{\psi(x,y) + \kappa(x,y)}{2m} \right) \tag{4.4}$$

where $\psi(x,y) = \sqrt{\kappa^2(x,y) + 4m^2}$. Here m does not play the same role. It is a constant fixed to the expected signal rms.

More generally MEM method present many drawbacks (Narayan and Nityananda, 1986; Starck et al., 2001).

We can find in the following website : "http://www.mrao.cam.ac.uk/projects/lensent/" a version of the MEM algorithm (LensEnt2 package).

Chapter 5

The Multiscale Entropy Filtering

In this chapter, we introduce a new reconstruction method (Starck et al., 2005) : an iterative filtering based on Bayesian methods. This filtering uses a Multiscale Entropy prior which is only defined for non-significant wavelet coefficients selected by the False Discovery Rate (FDR) Method (Benjamini and Hochberg, 1995).

We first describe the "*à* trous" wavelet transform which is well suited to astronomical data because of its isotropic scaling function. Then we introduce the multi-resolution entropy concept and we give our choice for the entropy function in this application. And finally we present the multiscale entropy filtering used by (Starck et al., 2005) with all the improvements of this new version.

5.1 The "*à* trous" Isotropic Wavelet Transform

In the early 1980s the wavelet transform was studied theoretically in geophysics and mathematics by Morlet, Grossman and Meyer. In the late 1980s, links with digital signal processing were pursued by Daubechies and Mallat, thereby putting wavelets firmly into the application domain.

The wavelet transform of a signal produces, at each scale j, a set of zero-mean coefficient values w_j. Using an algorithm such as the "*à* trous" method (Holschneider et al., 1989), each scale w_j has the same number of pixels as the signal and thus this wavelet transform is a redundant one. Furthermore, using a wavelet defined as the difference between the scaling functions of two successive scales $\left(\frac{1}{2}\psi\left(\frac{x}{2}\right) = \phi(x) - \phi\left(\frac{x}{2}\right)\right)$, the original signal κ, with a pixel at position (x, y), can be expressed as the sum of all the wavelet scales w_j and the smoothed array C_J

$$\kappa(x, y) = C_J(x, y) + \sum_{j=1}^{J} w_j(x, y)$$

Thus, the algorithm outputs $J + 1$ sub-band arrays of size $n \times n$. We will use an indexing convention such that $j = 1$ corresponds to the finest scale (high frequencies).

Hence, we have a multiscale pixel representation, i.e. each pixel of the input signal is associated to a set of pixels of the multiscale transform.

A summary of the "$\grave{a}$ trous" wavelet transform algorithm is as follows :

1. Initialize j to 0, starting with a signal $C_j(x, y)$ (where $C_0(x, y) = \kappa(x, y)$). Index (x, y) range over all pixels.

2. Carry out a discrete convolution of the data $C_j(x, y)$ using a filter h, yielding $C_{j+1}(x, y)$. The convolution is an interlaced one, where the filter's pixel values gap (growing with level, j) between them of 2^j pixels, giving rise to the name $\grave{a}$ trous ("with holes"). Mirroring is used at the data extremes

3. From this smoothing we obtain the discrete wavelet transform, $w_{j+1}(x, y) = C_j(x, y) - C_{j+1}(x, y)$.

4. If j is less than the number J of resolution levels wanted, then increment j and return to step 2.

The set $w = w_1, w_2, .., w_J, C_J$, where C_J is a last smooth array, represents the wavelet transform of the data. If the input data has N pixels, then its transform by the "$\grave{a}$ trous" Wavelet Transform has (J+1)*N pixels. The redundancy factor is J+1 whenever J scales are employed.

The discrete filter h is derived from the scaling function $\phi(x)$ which is a spline of degree 3, which leads to the filter $h = \left(\frac{1}{16}, \frac{1}{4}, \frac{3}{8}, \frac{1}{4}, \frac{1}{16}\right)$. A 2D implementation can be based on two 1D sets of (separable) convolutions.

The associated wavelet function is of mean zero, of compact support, with a central bump and two negative side-lobes. Of interest for us is that, like the scaling function, it is nearly isotropic.

Fig. 5.1 shows the $\grave{a}$ trous" transform of the mass map Fig. 2.4 (left). Five wavelet scales are shown and the final smoothed plane (lower right). The original image is given exactly by the sum of this six images.

5.2 Multiscale Entropy

1. Multiscale Entropy definition:

   The Multiscale Entropy method is based on the standard MEM prior derived from the wavelet decomposition of a signal. The idea is to consider the entropy of a signal as the sum of the information at each scale of its wavelet transform. And the information of a wavelet coefficient is related to the probability of its being due to noise.

   Denoting $H(\kappa)$ the information relative to the signal and $h(w_j(k, l))$ the information

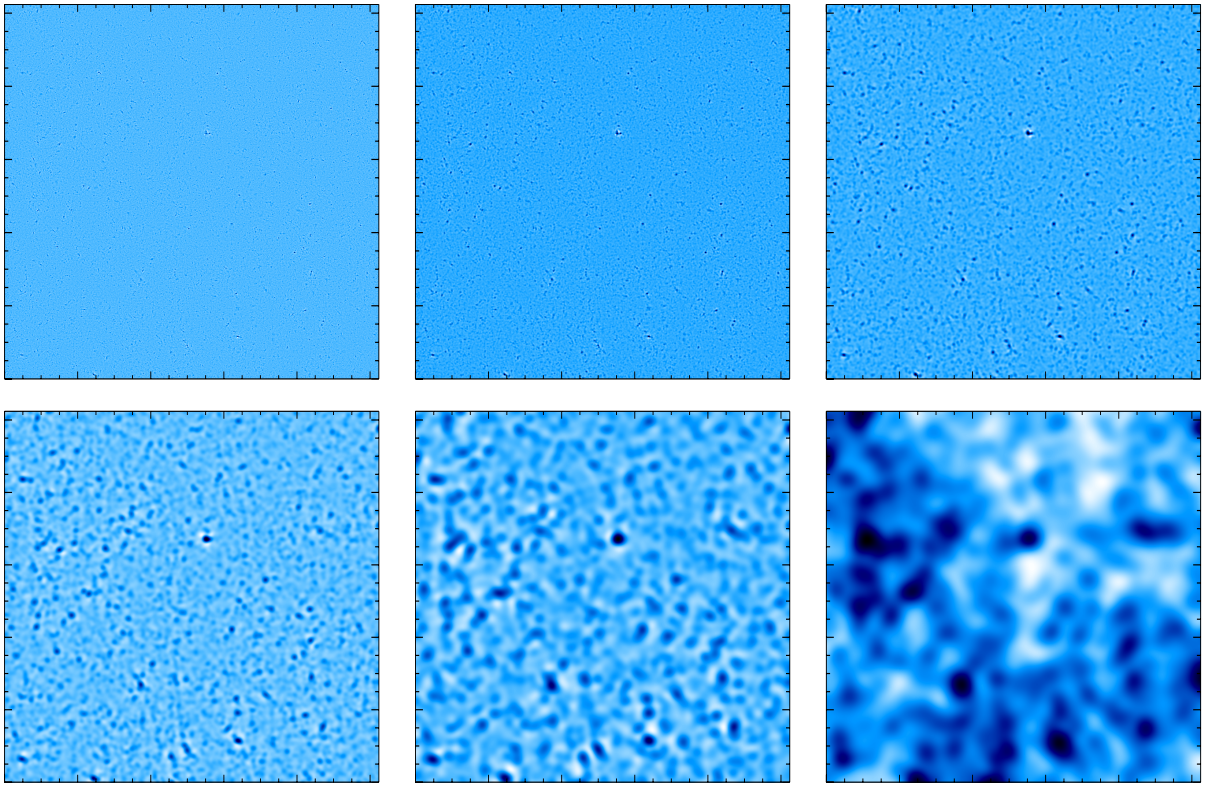


Figure 5.1: Wavelet transform of the previous mass map by the à trous" algorithm

relative to a single wavelet coefficient, the entropy is now defined as:

$$H(\kappa) = h(C_J(k,l)) + \sum_{j=1}^{l} \sum_{k,l=1}^{N_j} h(w_j(k,l)) \tag{5.1}$$

where l is the number of scales and N_j is the size of map in the band (scale) j.

2. Entropy definition:

   The function h in (5.1) assesses the amount of information carried by a specific wavelet coefficient. Several functions have been proposed for h. A discussion and comparison between different entropy definitions can be found in (Starck et al., 2005). We choose the NOISE-MSE entropy (Starck et al., 2001) for the Weak Lensing reconstruction problem in which the entropy is derived using a model of the noise contained in the data:

$$h(w_j(k,l)) = \int_0^{|w_j(k,l)|} P_n(\mid w_j(k,l) \mid -u)(\frac{\partial h(x)}{\partial x})_{x=u} du \tag{5.2}$$

   where $P_n(w_j(k,l))$ is the probability that the coefficient $w_j(k,l)$ can be due to the

noise: $P_n(w_j(k,l)) = \text{Prob}(W > \mid w_j(k,l) \mid)$. For Gaussian noise, we have:

$$\begin{aligned} P_n(w_j(k,l)) &= \frac{2}{\sqrt{2\pi}\sigma_j} \int_{|w_j(k,l)|}^{+\infty} \exp(-W^2/2\sigma_j^2) dW \\ &= \text{erfc}(\frac{\mid w_j(k,l) \mid}{\sqrt{2}\sigma_j}) \end{aligned} \tag{5.3}$$

and equation 5.2 becomes 5.4

$$h(w_j(k,l)) = \frac{1}{\sigma_j^2} \int_0^{|w_j(k,l)|} u \ \text{erfc}(\frac{\mid w_j(k,l) \mid -u}{\sqrt{2}\sigma_j}) du \tag{5.4}$$

The NOISE-MSE is very close to the l_1 norm (i.e. absolute value of the wavelet coefficient) when the coefficient value is large, which is known to produce good results for the analysis of piecewise smooth images (Donoho and Elad, 2003).

3. Signal and noise information:

   The mass map derived from shear maps is swamped by noise. The following algorithm assumes that the observed map can be decomposed as:

$$\kappa_{obs} = \kappa + N \tag{5.5}$$

   Then, we can decompose the information contained in our image in two components, the first one (H_s) corresponding to the non corrupted part, and the other one (H_n) describing a component which contains no information for us:

$$H(\kappa_{obs}(k,l)) = H_s(\kappa_{obs}(k,l)) + H_n(\kappa_{obs}(k,l)) \tag{5.6}$$

   For each wavelet coefficient $w_j(k,l)$, we have to estimate the fractions h_n and h_s of h:

$$H(\kappa_{obs}(k,l)) = \sum_{j=1}^{l} \sum_{k,l=1}^{N_j} h_s(w_j(k,l)) + \sum_{j=1}^{l} \sum_{k,l=1}^{N_j} h_n(w_j(k,l)) \tag{5.7}$$

$$\tag{5.8}$$

5.3 Multiscale Entropy Filtering

1. Filtering:
   The problem of filtering the observed map κ_{obs} can be expressed as follows. We look for a filtered map κ_f such that the difference between κ_f and κ_{obs} minimizes the information due to the signal (to recover all the signal) and such that κ_f minimizes the information due to the noise. These two requirements are somehow competing. A tradeoff is necessary, because, on one hand, we want to remove all the noise (heavy filtering) and on the other hand, we want to recover the signal with fidelity. In practice, we minimize for each wavelet coefficient $w_j(k,l)$:

$$l(\tilde{w}_j(k,l)) = h_s(w_j(k,l) - \tilde{w}_j(k,l)) + \beta . h_n(\tilde{w}_j(k,l)) \tag{5.9}$$

where $w_j(k,l)$ are the wavelet coefficients of the observed map κ_{obs}, $\tilde{w}_j(k,l)$ the wavelet coefficients of the filtered map κ_f and β is the so called regularization (trade-off) parameter.

2. Selecting significant Wavelet coefficients:
Whatever the filtering, the signal is always substantially modified. We want to fully reconstruct significant structures, without imposing strong regularization while eliminating efficiently the noise. The introduction of the multiresolution support (Murtagh et al., 1995), helps to do so. The idea is to apply the previous regularization (i.e. filtering) only on the non-significant (noisy) wavelet coefficients (Pantin and Starck, 1996). In this way, the choice of the regularization parameter is not really a critical point. The other components of the maps are left untouched. The new Multiscale Entropy becomes:

$$\tilde{h}(w_j(k,l)) = \bar{M}(j,k,l)h(w_j(k,l)) \tag{5.10}$$

where $\bar{M}(j,k,l) = 1 - M(j,k,l)$, and M is the "multiresolution support" defined as:

$$M(j,k,l) = \begin{cases} 1 & \text{if } w_j(k,l) \text{ is significant} \\ 0 & \text{if } w_j(k,l) \text{ is not significant} \end{cases} \tag{5.11}$$

M describes, in a Boolean way, whether the data contains information at a given scale j and at a given position (k,l). $w_j(k,l)$ is said to be significant if the probability that the wavelet coefficient is due to noise is small. In the case of Gaussian noise, a coefficient $w_j(k,l)$ is significant if $\mid w_j(k,l) \mid > k\sigma_j$, where σ_j is the noise standard deviation at scale j, and k is a constant. Without an objective method for selecting the threshold, it is adjusted arbitrarly, generally taken between 3 and 5 (Murtagh et al., 1995).

3. Selecting significant Wavelet coefficients using the FDR:
The False Discovery Rate (FDR) is a new statistical procedure due to (Benjamini and Hochberg, 1995) which offers an effective way to select an adaptative threshold to compute the multiresolution support. This technique has recently been described by (Miller et al., 2001; Hopkins et al., 2002; Starck et al., 2005; Pires et al., 2005) with several examples of astrophysical applications. The FDR procedure provides the means to adaptively control the fraction of false discoveries over total discoveries. The FDR is given by the ratio (5.12), that is, the proportion of declared active which are false positives:

$$\mathcal{FDR} = \frac{V_{ia}}{D_a} \tag{5.12}$$

where V_{ia} is the number of pixels truly inactive declared active, and D_a is the number of pixels declared active. The FDR formalism ensures that, *on average*, the False Discovery Rate is no larger than α which lies between 0 and 1. This procedure guarantees control over the FDR in the sense that:

$$\mathcal{E}(\mathcal{FDR}) \leq \frac{T_i}{V}.\alpha \leq \alpha \tag{5.13}$$

The unknown factor $\frac{T_i}{V}$ is the proportion of truly inactive pixels where T_i is the number of inactive pixels and V the total number of pixels.

The FDR procedure is as follows :
Let $P_1, ..., P_n$ denote the p_values from the N tests, listed from smallest to largest.
Let :

$$d = max\{k : P_k < \frac{k.\alpha}{c_N.N}\} \tag{5.14}$$

where $c_N = 1$, if p_values are statistically independants.
Now, declare actived all the pixels with p_values less than or equal to P_d.
Graphically, this procedure corresponds to plotting the P_k versus $\frac{k}{N}$, superposing the line through the origin of slope $\frac{\alpha}{c_N}$ (see Fig. 5.2), and finding the last point at which P_k falls below the line, termed P_d. From this p_value P_d, we can derive a threshold $\mathcal{T}$. All the pixels greater than $\mathcal{T}$ have a p_value less than P_d and are declared actives.

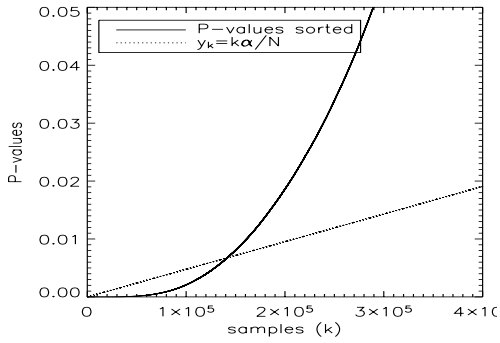


Figure 5.2: Finding a threshold graphically using the FDR procedure

A complete description of the FDR method can be found in (Miller et al., 2001). In (Hopkins et al., 2002; Starck et al., 2005; Pires et al., 2005), it has been shown that the FDR outperforms standard method for source detection.

In this application, we use the FDR method in a multiresolution framework (see (Starck et al., 2005)). We select a detection threshold $\mathcal{T}_j$ for each scale. We have chosen to take a different α value per scale using the following relation as in (Starck et al., 2005) : $\alpha_j = \alpha_0 * 2^j$ where α_0=0.0125 for $n_g = 100$ and α_0=0.017 for $n_g = 20$. A wavelet coefficient $w_j(k, l)$ is considered significant if its absolute value is larger than $\mathcal{T}_j$ as seen below.

4. Multiscale Entropy Filtering algorithm:

   Assuming Gaussian noise, the Multiscale Entropy restoration method reduces to finding the image κ_f that minimizes $J(\kappa_f)$, given the map κ_{obs} output of source separation with:

$$J(\kappa_f) = \frac{\| \kappa_{obs} - \kappa_f \|^2}{2\sigma_n^2} + \beta \sum_{j=1}^{J} \sum_{k,l} \tilde{h}_n((\mathcal{W}\kappa_f)_{j,k,l}) \tag{5.15}$$

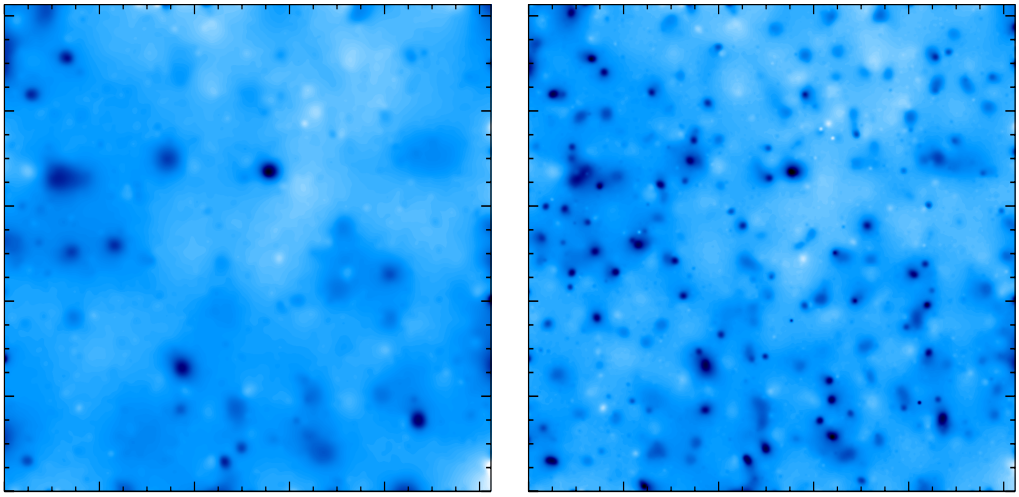


Figure 5.3: Multiscale Entropy filtering: left for $n_g = 20 gal/arcmin^2$ and right $n_g = 100 gal/arcmin^2$.

where σ_n is the noise standard deviation in κ_{obs}, J is the number of Wavelet scales, $\mathcal{W}$ is the Wavelet Transform operator and $\tilde{h}_n(w_{j,k,l})$ is the multiscale entropy only defined for non significant coefficient (outside de Support selected by the FDR thresholding). Full details of the minimization algorithm can be found in (Starck et al., 2001) .

The result of the filtering by the Multiscale Entropy can be improved by an iterative process. Its goal is to recover the information lost during the reconstruction by the inverse Wavelet Transform. In fact, the Wavelet Transform is not reversible anymore after thresholding.

This process consists in adding to the map κ_f^i, for each iteration i, a residual obtained by the inverse Wavelet Transform of the difference between the corrected wavelet coefficients and the wavelet coefficients calculated from the map reconstructed at the iteration i-1. The convergence is rather fast, approximately 5 iterations are enough.

$$\kappa_f{}^{iter} = \kappa_f{}^{iter-1} + \mathcal{W}^{-1}(\bar{M}(\tilde{w} - \mathcal{W}\kappa_f{}^{iter-1})) \tag{5.16}$$

where $\mathcal{W}$ and $\mathcal{W}^{-\infty}$ are the Wavelet Transform operator and its inverse operator; $\bar{M} = 1 - M$, and M is the "multiresolution support"; $\tilde{w}$ are the wavelet coefficients filtered by our multiscale entropy filtering and κ_f^{iter-1} is the mass map reconstructed at the iteration i.

Fig. 5.3 shows the result of the filtering by the Multiscale Entropy using the FDR to compute a different threshold per scale and using the iterative process (5 iterations). The FDR method, guarantees control over the false detection rate and the iterative process enables a better recovering of image levels.

Chapter 6

Results

6.1 Comparisons

As we have seen previously, we have used a simulated data set obtained using a standard Λ-CDM cosmological model. The κ mass map and the shear maps are shown on Fig. 2.4. The field size is 2×2 square degrees, sampled with $1024 * 1024$ pixels.

Noisy shear maps, corresponding to both spatial (i.e. $n_g = 100$ gals/amin^{-2}) and ground-based observations (i.e. $n_g = 20$ gals/amin^{-2}), have been created using equation 2.7. Then we have reconstructed the two noisy mass maps from equation 2.9 and applied the following filtering methods:

1. Gaussian filtering with a standard deviation equal to $\sigma_G = 1$ amin
2. Gaussian filtering with a standard deviation equal to $\sigma_G = 2.5$ amin
3. Wiener filtering.
4. Maximum Entropy Method (MEM) using the **LensEnt2** package. As this code has not been designed for manipulating large images, we had to restrict the restoration by this method to a field size of 0.5×0.5 square degree, sampled with 256*256 pixels.
5. Multiscale Entropy method.

The evaluation is done by i) visual inspection of the images, ii) computing the standard deviation of the difference between the original κ mass map and the reconstructed map (i.e. $E = \frac{STD(\kappa - \tilde{\kappa})}{STD(\kappa)}$), iii) computing the standard deviation of the difference between the original κ mass map and the reconstructed map scale by scale (i.e. $E_j = \frac{STD((\mathcal{W}\kappa)_j - (\mathcal{W}\tilde{\kappa})_j)}{STD((\mathcal{W}\kappa)_j)}$), iv) computing the log power spectrum of the error between the original κ mass map and the reconstructed map (i.e. $\mathrm{P} = log(PSD(\kappa - \tilde{\kappa}))$ where PSD is the Power Spectral Density.

6.1.1 Visual inspection of the images

Fig. 6.1 shows from top to bottom the reconstructed maps for the Gaussian, the Wiener and the Multiscale Entropy filtering. Fig. 6.1 left corresponds to ground-based observations (i.e. $n_g = 20$) and Fig. 6.1 right corresponds to spatial observations (i.e. $n_g = 100$).

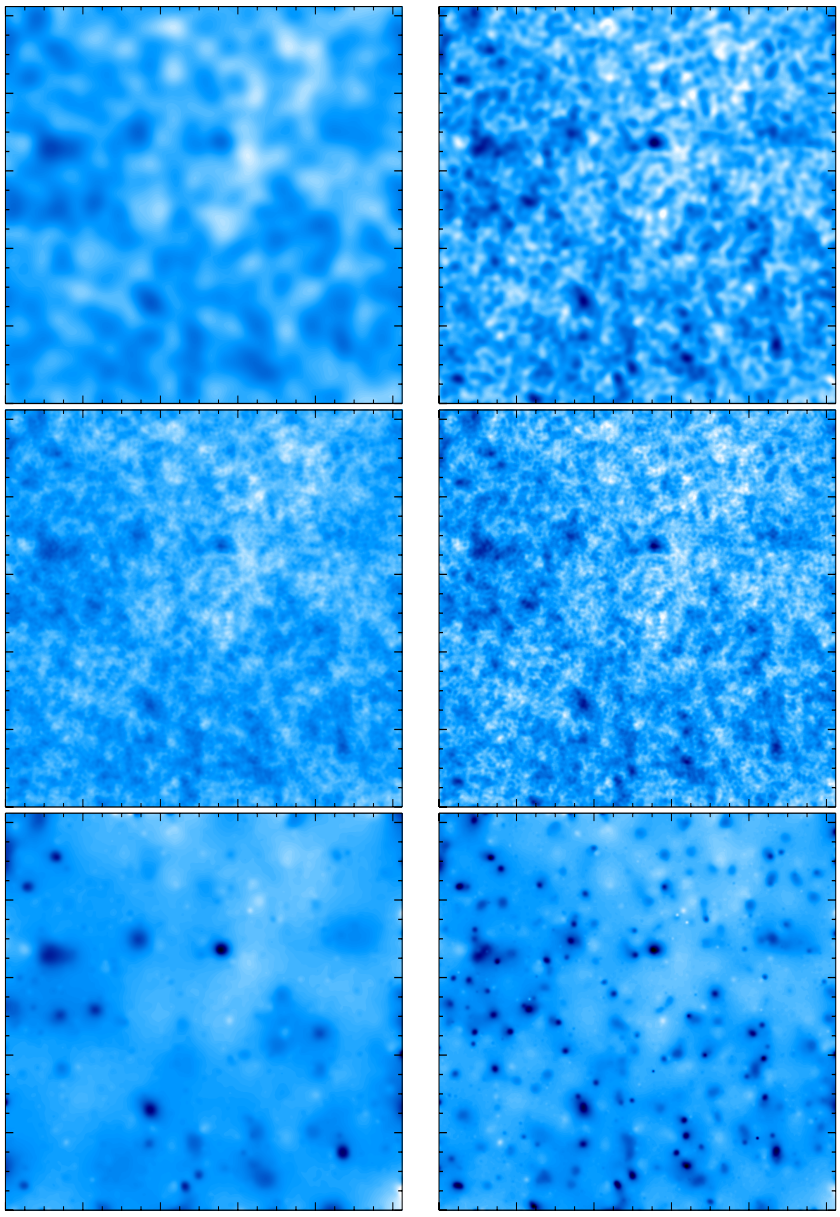


Figure 6.1: Restoration of the 2×2 square degrees ground-based observation (left) and spatial observation (right). From top to bottom, Gaussian filtering, Wiener filtering and Multiscale Entropy filtering.

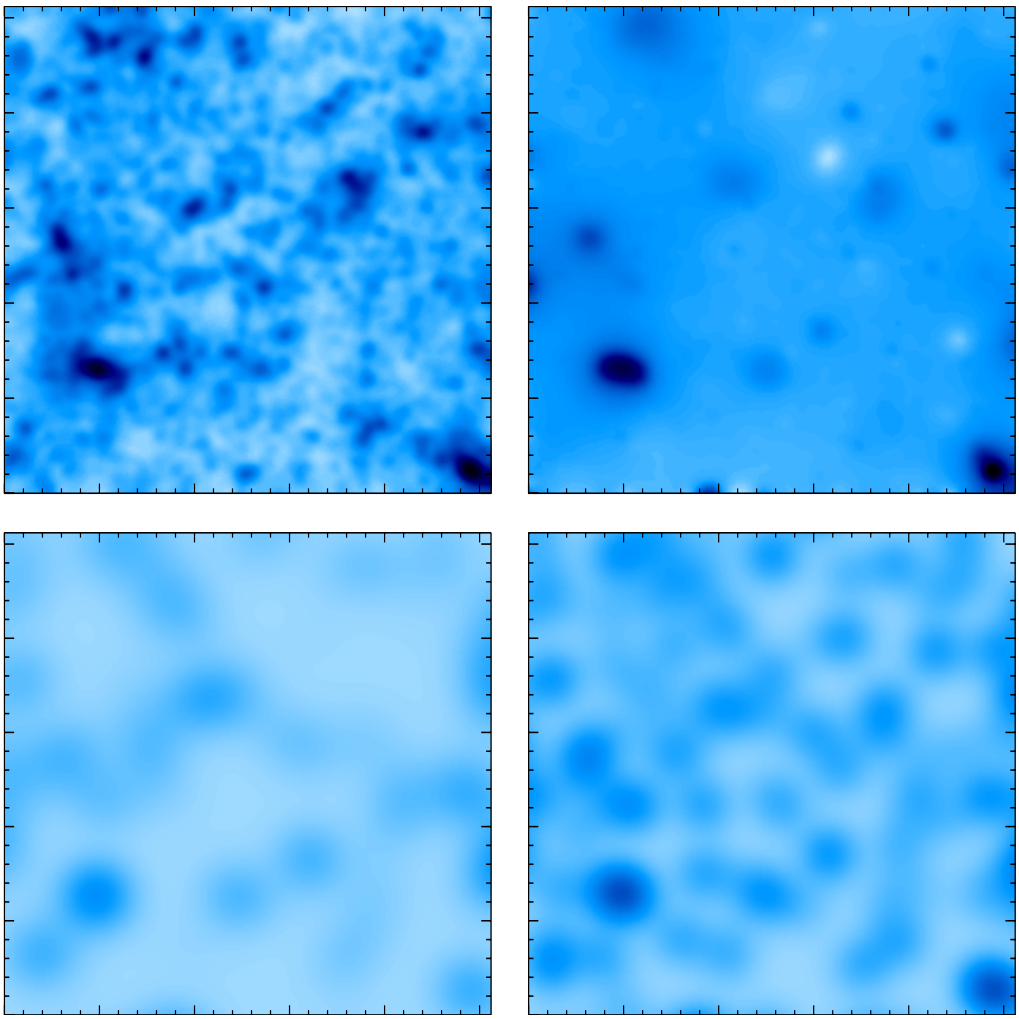


Figure 6.2: In a region of 0.5×0.5 square degrees, a sixteenth of the original field : Upper left, simulated mass map, upper right, Multiscale entropy filtering for $n_g = 100$ gal/arcmin^2. Bottom left, MEM filtering for $n_g = 20$ gals/amin^{-2} ($ICFwidth = 210$) and bottom right for $n_g = 100$ gals/amin^{-2} ($ICFwidth = 180$).

Fig. 6.2 shows the denoising results on a portion of the previous image. Fig. 6.2 shows the original noise free simulated image of the 0.5×0.5 square degrees field (upper left), the Multiscale Entropy Filtering for the spatial simulated observations ($n_G = 100$) (upper right), the MEM-LensEnt2 restoration with an ICF (Intrinsic Correlation function) equal to 210" for the ground based observations (bottom left) and the spatial observations with an ICF equal to 180" (bottom right).

The computation time for the 1024*1024 pixels map is 4 minutes for the Multiscale Entropy method, 26 seconds for the Wiener filtering and 4 seconds for the Gaussian smoothing. The computation time for the 256*256 pixels map is around 60 minutes (it depends on the convergence of the result) using the MEM-LensEnt2 package.

6.1.2 Quadratic error in direct space

Method	Error ($n_g = 20\ gal/amin^2$)	Error ($n_g = 100\ gal/amin^2$)
Gaussian Filtering ($\sigma_G = 1$ amin)	1.108	0.775
Gaussian Filtering ($\sigma_G = 2.5$ amin)	0.9138	0.868
Wiener Filtering	0.888	0.770
MEM-LensEnt2	1.091	0.821
Multiscale Entropy Filtering	0.888	0.746

Table 6.1: Standard deviation of the reconstruction error with five different methods.

Table 6.1 gives the standard deviation of the error for the four reconstructed mass maps. It shows that i) the Wiener filtering is better than the Gaussian filtering and the MEM-LensEnt2 method and ii) the Multiscale Entropy outperforms the three other methods.

6.1.3 Wavelet-based estimator

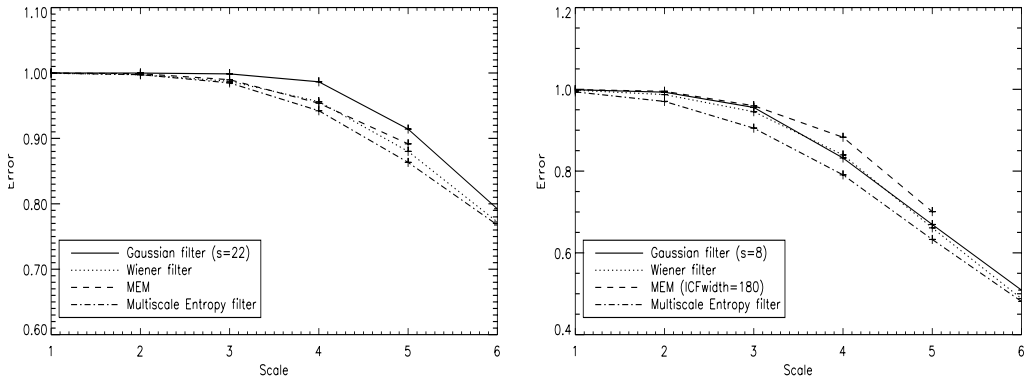


Figure 6.3: Standard deviation versus scale for the ground-based simulation (left) and the space-based simulation (right).

Fig. 6.3 shows the error versus the scale (each wavelet scale) for both simulations using the Gaussian filtering (continuous line), the Wiener filtering (dotted line), the MEM-LensEnt2 filtering (dashed line) and the Multiscale Entropy filtering (dotted-dashed line).

The wavelet scales 1 to 6 correspond to scales of $0.12, 0.23, 0.47, 0.94, 1.87, 3.75$ amin respectively. We can see that the Multiscale Entropy method produces better results for all scales.

6.1.4 Fourier-based estimator

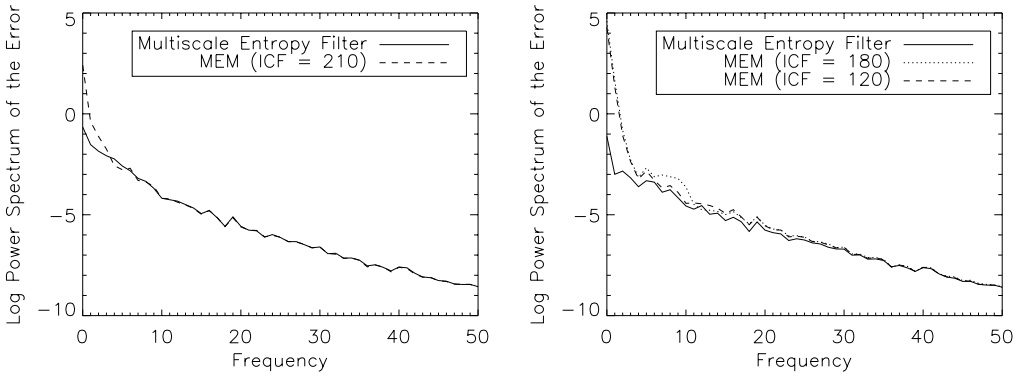


Figure 6.4: Log Power Spectrum of the Error by Multiscale Entropy Filter and MEM for the ground-based simulation (left) and the space-based simulation (right).

Fig. 6.4 shows the log power spectrum of the error. It is very consistent with the previous one. Indeed, the MEM error becomes very important toward the smallest frequencies (largest wavelet scales). The same experiment has been done with a smaller ICF (ICF = 120" for the spatial simulation), but the result is worse, which is surprising since this ICF value was chosen to get the best results by maximizing the evidence (See (Marshall et al., 2002))

6.2 Robustness to missing data

During the observations, various problem can cause a loss of data in the image. In order to study this problem, we mask two rectangular areas, setting all pixel values to 0, in the shear maps γ_1 and γ_2. By inverse filtering, we have derived the noisy mass map κ_b in which we can also visualize the lack of data (Fig. 6.5 upper left). Then we have applied the three methods, Gaussian filtering, Wiener filtering and Multiscale Entropy, to the noisy mass map and the results can be seen respectively in Fig. 6.5 upper right, Fig. 6.5 bottom left and bottom right. We can see that all three methods are robust to the missing data. Note however that, for the Wiener filtering, we have assumed perfect knowledge of the power spectrum of κ, while, in practice, its estimation is made more complicated by the complex field geometry.

Fig. 6.6 shows the error versus the scale for both simulations using the Gaussian filtering (continuous line), the Wiener filtering (dotted line) and Multiscale Entropy. We can see that the Multiscale Entropy still produces better results at all scales. Bayesian methods such as MEM could also be made to properly handle missing data, however this is not as straightforward as in multiscale methods.

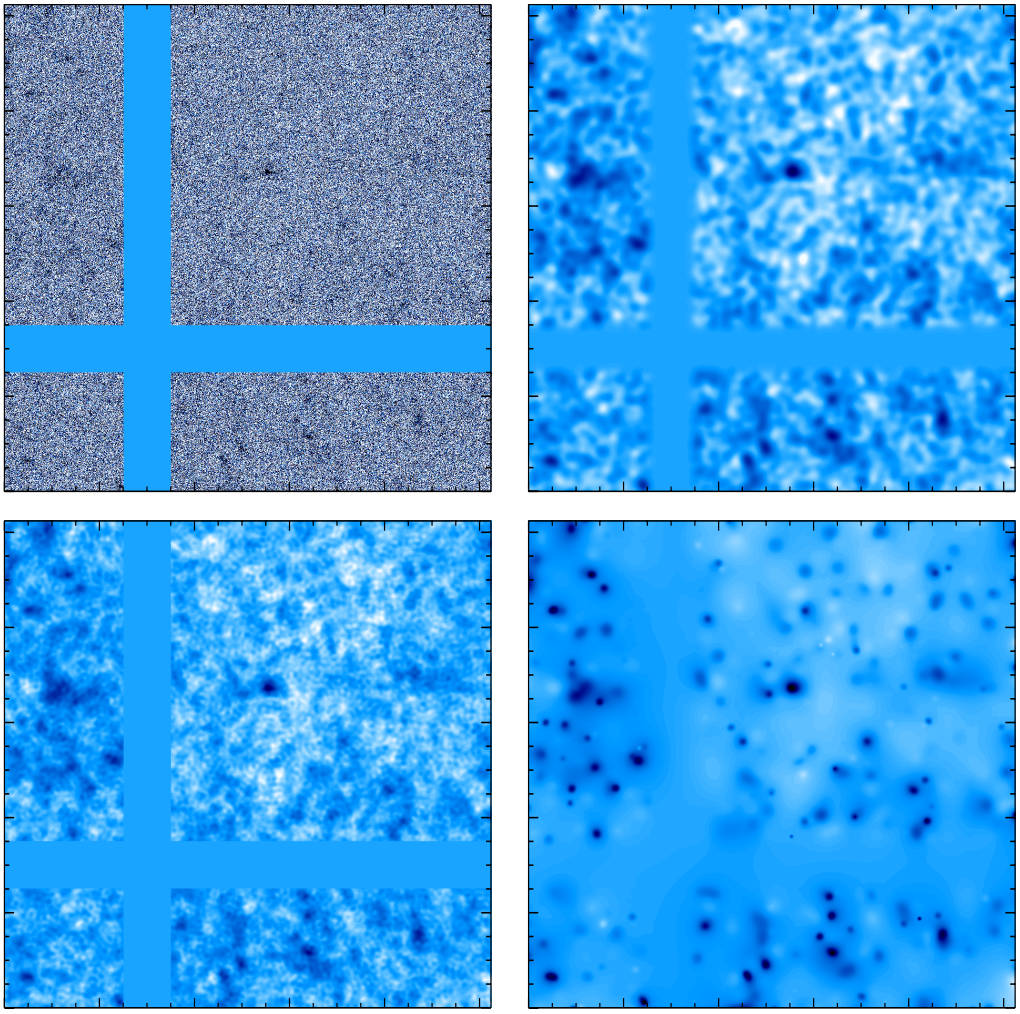


Figure 6.5: Upper left, noisy shear map ($n_g = 100\ gal/arcmin^2$). Upper right, Gaussian filtering. Bottom left, Wiener filtering, and bottom right, Multiscale Entropy filtering.

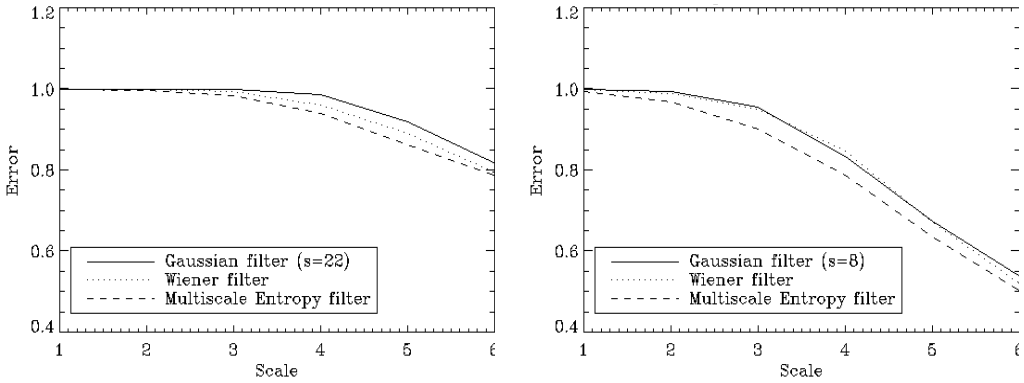


Figure 6.6: Standard deviation versus scale for the ground-based simulation(left) and the spatial simulation (right) with missing data

6.3 Cluster detection

Another important aspect of the weak shear mass reconstruction is the possibility to detect clusters and to build a catalog. Here, using the FDR in the wavelet space, we detect as significant a set of wavelet coefficients. We built an isophote map, where each isophote level corresponds to the detection level in a given scale. This isophote is overplotted on the true mass map, which allows us to visually check the false detections and the missed detections. A cluster surrounded by two isophotes means that it has been detected at two scales.

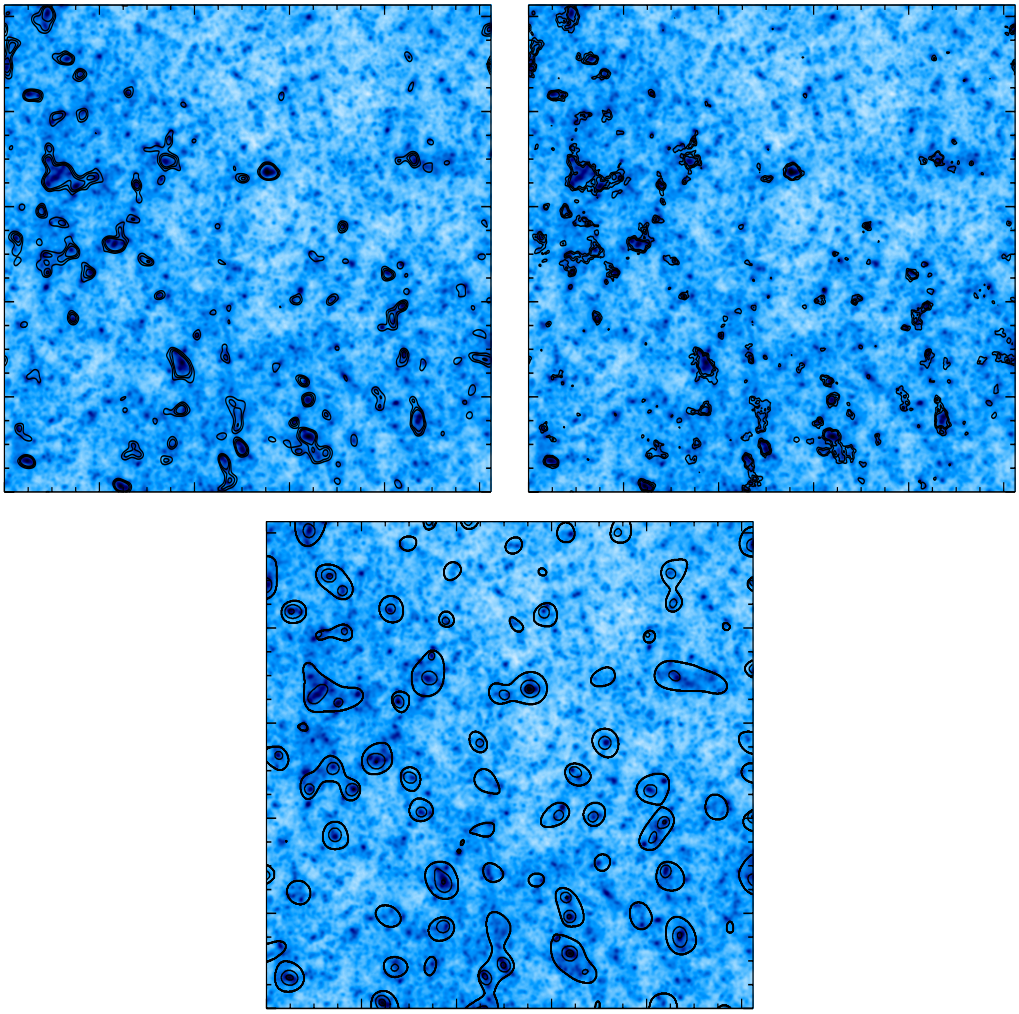


Figure 6.7: The isophotes represent the detected clusters using the Gaussian filtering (upper left), the Wiener filtering (upper right) and the wavelet-FDR method.

Figure 6.7 shows a comparison between the Gaussian filtering, the Wiener filtering and FDR-Wavelet method for the detection of clusters. In the Gaussian and Wiener maps, the isophotes corresponds to a $k\sigma$ detection level where $k = 3, 4, 5$. It shows clearly how the FDR-Wavelet method outperforms the other methods.

6.4 E/B Decomposition

The shear pattern induced by lensing has even parity. It is often referred to as a (positive) E-mode. In the absence of lens-lens coupling or higher order effects the shear pattern induced by lensing is pure E since lensing arises from a scalar gravitational potential. A 45° rotation of the shear to produce the B-mode, will null the lensing map. Thus as explained in section 2 §2.7, a simple diagnostic test for a wide range of systematic effects is to search for the presence of B-mode in the lensing maps. In order to test this, we have simulated mass maps with a B-mode.

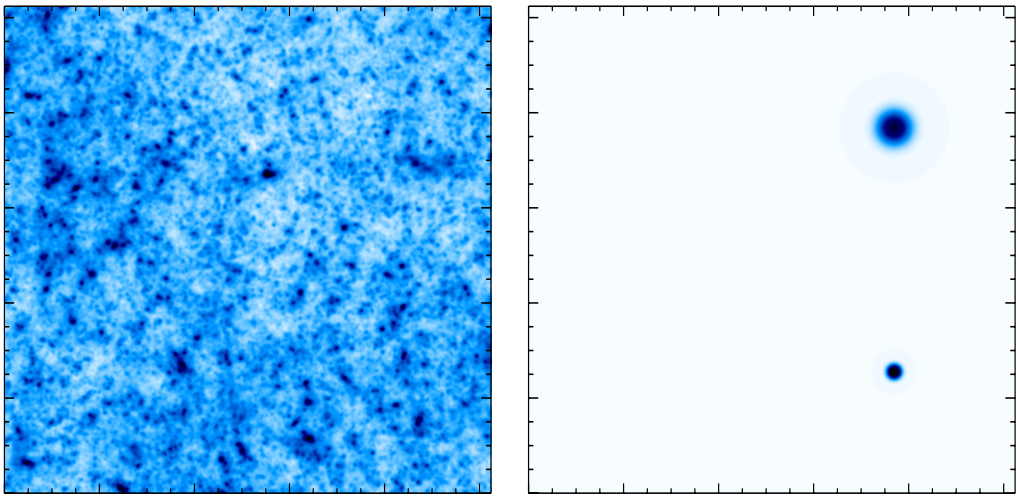


Figure 6.8: left mass map (E-mode), right mass map (B-mode)

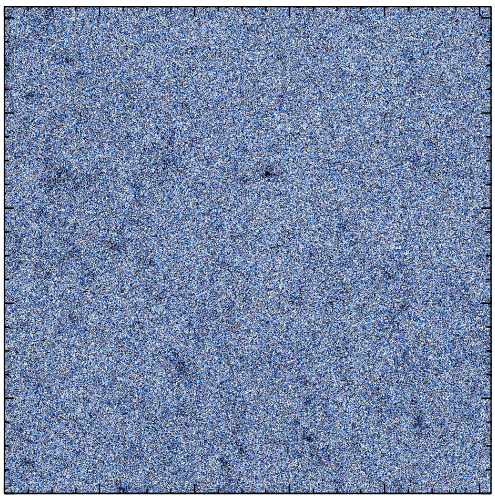


Figure 6.9: Noisy simulated mass map

Fig. 6.8 left shows a simulated mass map with a lensing E-mode signal (left) and an arbitrary B-mode signal (right). As usual, we have added a realistic space-based Gaussian noise to the shear of this simulation. Fig. 6.9 shows the resulting noisy mass map. Using the Multiscale Entropy filtering, we have then reconstructed the two components of the

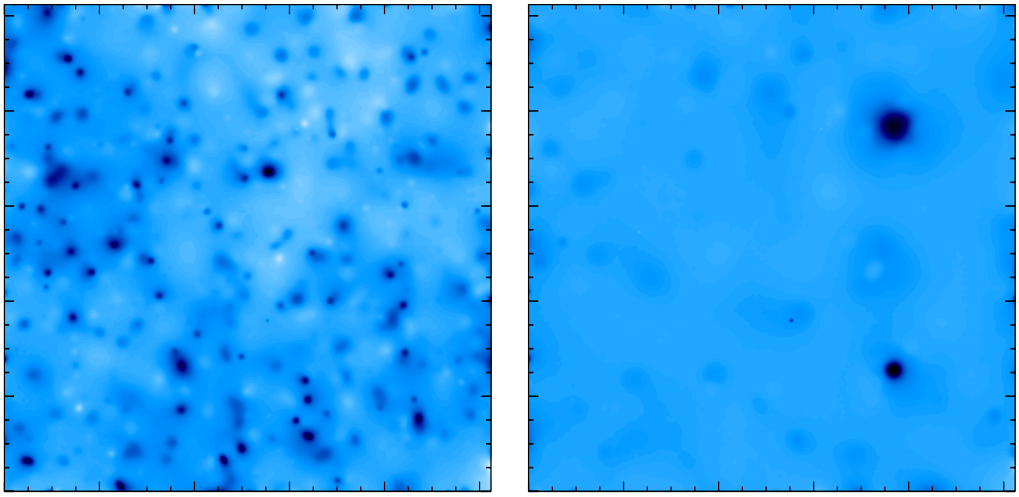


Figure 6.10: left filtered noisy mass map (E-mode), right filtered noisy mass map (B-mode)

mass map (see §2.7): E-mode in Fig. 6.10 left and B-mode in Fig. 6.10 right. We see clearly that the wavelet separation of the E and B modes is very good. Indeed, the two main features in the B-mode have been well recovered, without interfering with the reconstruction of the E-mode.

Chapter 7

IDL Routines

A set of routines has been developed in IDL (summarized in flowchart Fig. 7.1). Starting IDL using the script program *mrl.pro* allows the user to add the MRLENS software to the IDL environment. Thus, all routines described in the following can be called. An online help, facility is also available by calling the *mrh* IDL program.

7.1 Installation

7.1.1 System requirements

- Disk space : Make sure you have approximately 400 MB of disk space available. After installation MRLENS package occupies approximately 100 MB or 200MB (version with data) of disk space.

- Platform : The binaries C++ called by IDL routines are not available under all the systems therefore you cannot use the package on all platforms. The supported platforms are : Unix, Linux, Mac OS X. They will be soon available in Windows platform.

7.1.2 Download

Use the link to download the package MRL and copy the file in your home directory (/home/user/).
Then, uncompress the filename.tar.gz file by typing:
gunzip filename.tar.gz
tar -xvf filename.tar

7.1.3 Installation instructions

The MRLENS package requires that IDL (version 6.0 or later) to be installed. The alias **idl** should also be defined to launch the IDL environment. Then, installing the MRLENS package simply requires adding some lines in your shell environment profile depending on

your shell : (The command "echo $SHELL" will give your SHELL environment : bash, csh or tcsh)

- define the environment variable **MRL** :
  In csh or tcsh :
  setenv MRL /home/user/MRL
  In bash :
  MRL=/home/user/MRL;export MRL

- define the alias **mrl**
  In csh or tcsh :
  alias mrl 'idl $MRL/idl/mrl'
  In bash :
  alias mrl ='idl $MRL/idl/mrl'

7.1.4 Startup instructions

The command "mrl" will start the IDL session using the MRL environment. The programs can be found in $MRL/idl. These routines use data in directory $MRL/Data. The command "mrh" will open the online help.

Two scripts are included in the package giving examples of how to run some main routines :
- mk_test_1.pro
- mk_test_2.pro
These scripts use the data files provided with the package in $MRL/Data. The first use a simulated noiseless mass map and the second a simulated noisy catalogue.

7.2 Build an Electric noisy mass map

7.2.1 IDL routines for simulated data

▶ Starting from a mass map without noise

↬ **mk_kappa.pro** : Build a mass map structure from a mass map :
Build a mass map structure from the mass map and the field size in pixels and in degrees (or arcmin)

USAGE : mk_kappa, map, npix1, npix2, theta1, theta2, amin = amin, smap

INPUTS :

- map = IDL array of mass map
- npix1, npix2 = (int) map size in pixels
- theta1, theta2 = (int) map size in degrees (or arcmin if the keyword amin is set)

KEYWORD :

- amin = (string) if set, the map size theta1 and theta2 are assumed to be in arcmin

OUTPUTS :

- smap = mass map IDL structure with the following fields :
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = (IDL array) mass map

↬ **add_gnoise_kappa.pro** : Add a Gaussian noise to a mass map structure : Add a Gaussian noise to the mass map by adding a gaussian noise to the shear maps depending on the number of galaxies per pixel.

USAGE : add_gnoise_kappa, smap, ng, s, smapn

INPUTS :

- smap = IDL structure of mass map with the following fields :
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = (IDL array) mass map
- ng = (int) galaxies number per arcmin square

OUTPUTS :

- s = (int) the root mean square of the noise
- smapn = IDL structure of mass map embedded in a gaussian noise with the following fields :
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = (IDL array) noisy mass map

▶ Starting from a shear maps without noise

↬ **mk_gamma.pro** : Build a shear maps structure from shear maps : Build a shear maps structure from shear maps and the field size in pixels and in degrees (or arcmin)

USAGE : mk_gamma, gamma1, gamma2, npix1, npix2, theta1, theta2, amin = amin, sgamma

INPUTS :

- gamma1, gamma2 = IDL array of shear maps γ_1 and γ_2
- npix1, npix2 = (int) map size in pixels

• theta1, theta2 = (int) map size in degrees (or arcmin if the keyword amin is set)

KEYWORD :

• amin = (string) if set, the map size theta1 and theta2 are assumed to be in arcmin

OUTPUTS :

• sgamma = shear maps IDL structure with the following fields :
- n1, n2 = (int) map size in pixels
- theta1, theta2 = (int) map size in radians
- delta1, delta2 = (int) pixel size in radians
- gamma1, gamma2 = (IDL array) shear maps γ_1 and γ_2

↬ **add_gnoise_gamma.pro** : Add a Gaussian noise to a shear maps structure : Add a Gaussian noise to the shear maps depending on the number of galaxies per pixel.

USAGE : add_gnoise_gamma, sgamma, ng, s, sgamman

INPUTS :

• sgamma = IDL structure of mass map with the following fields :
- n1, n2 = (int) map size in pixels
- theta1, theta2 = (int) map size in radians
- delta1, delta2 = (int) pixel size in radians
- gamma1, gamma2 = (IDL array) shear maps γ_1 and γ_2

• ng = (int) galaxies number per arcmin square

OUTPUTS :

• s = (int) the root mean square of the noise

• sgamman = IDL structure of shear maps embedded in a gaussian noise with the following fields :
- n1, n2 = (int) map size in pixels
- theta1, theta2 = (int) map size in radians
- delta1, delta2 = (int) pixel size in radians
- gamma1, gamma2 = (IDL array) noisy shear maps γ_{1b} and γ_{2b}

↬ **gamma_to_kappa.pro** : Derive a mass map structure from a shear maps structure: Derive a mass map structure from shear maps structure using the relation 2.9 in chapter 2.

USAGE : gamma_to_kappa, sgamma, smap

INPUTS :

• sgamma = IDL structure of shear maps with the following fields :
- n1, n2 = (int) map size in pixels
- theta1, theta2 = (int) map size in radians
- delta1, delta2 = (int) pixel size in radians
- gamma1, gamma2 = (IDL array) shear maps γ_1 and γ_2

KEYWORD :

- cat = (string) if set, some fields (specific to real data) are added to the output mass map structure (ng, wtot, mask, gamma_err, kappa_err, ng_eff, sigma_gamma,x1_ran, x2_ran, x1_m, x2_m)
- bmode = (string) if set, the magnetic component of the mass map is computed, otherwise is the electric component that is computed

OUTPUTS :

- smap = IDL structure of mass map with the following fields :
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = (IDL array) mass map
  - gamma1, gamma2 = (IDL array) shear maps γ_1 and γ_2

  if keyword_set(cat) :

- smap = IDL structure of mass map with the following fields :
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = (IDL array) mass map
  - gamma1, gamma2 = (IDL array) shear maps
  - ng = (IDL array) number of galaxies per pixel
  - wtot = (IDL array) weight per pixel
  - mask = (IDL array) mask of the missing data
  - gamma_err = (IDL array) measurement error in shear maps per pixel taking into account the weight of each galaxy
  - kappa_err = (IDL array) measurement error in mass map per pixel (kappa_err=gamma_err/sqrt(2.))
  - ng_eff = (IDL array) effective number of galaxies per pixel taking into account the weight of each galaxy
  - sigma_gamma = (int) measurement error in shear maps per pixel
  - x1_ran, x2_ran = (int) exact range in degree
  - x1_m, x2_m = IDL array with the exact middle position (in deg) of each pixel

7.2.2 IDL routines for real data

► Build a catalogue structure

↬ **mk_gcat.pro** : Build a shear catalogue structure :

USAGE : mk_gcat, x, y, pixscale, weight, g1, g2, gcat

INPUTS :

- x, y = (1D IDL array) coordinates in pixels of each galaxy

- pixscale = (int) pixel size [in rad]
- weight = (1D IDL array) weight of each galaxy
- g1, g2 = (1D IDL array) shear γ_1 and γ_2 of each galaxy

OUTPUTS :

- gcat = IDL structure of catalogue with the following fields :
  - x, y = (1D IDL array) coordinates in pixels of each galaxy
  - pixscale = (int) pixel size [in deg]
  - weight = (1D IDL array) weight of each galaxy
  - gamma1, gamma2 = (1D IDL array) shear γ_1 and γ_2 of each galaxy

► Build a shear maps structure

↬ **gcat_to_gamma.pro** : Build a (pixelised) shear maps structure from a shear catalogue structure :

USAGE : gcat_to_gamma, gcat, gamma, delta=delta

INPUTS :

- gcat = IDL structure of catalogue with the following fields :
  - x, y = (1D IDL array) galaxies position in pixels of each galaxy
  - pixscale = (int) pixel size [in rad]
  - weight = (1D IDL array) weight of each galaxy
  - gamma1, gamma2 = (1D IDL array) shear γ_1 and γ_2 of each galaxy

KEYWORD :

- delta = (string) pixel size in *arcmin*, default is 1

OUTPUTS :

- gamma = IDL structure of shear maps with the following fields :
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = (IDL array) mass map
  - gamma1, gamma2 = (IDL array) shear maps γ_1 and γ_2
  - ng = (IDL array) number of galaxies per pixel
  - wtot = (IDL array) weight per pixel
  - mask = (IDL array) mask of the missing data
  - gamma_err = (IDL array) measurement error in shear maps per pixel taking into account the weight of each galaxy
  - kappa_err = (IDL array) measurement error in mass map per pixel (kappa_err=gamma_err/sqrt(2.))
  - ng_eff = (IDL array) effective number of galaxies per pixel taking into account the weight of each galaxy
  - sigma_gamma = (int) measurement error in shear maps per pixel
  - x1_ran, x2_ran = (int) exact range in degree
  - x1_m, x2_m = IDL array with the exact middle position (in deg) of each pixel

7.3 Relations between the distortion field and the projected (Electric) mass concentration

7.3.1 From shear maps γ_1, γ_2 to mass map κ

↬ **gamma_to_kappa.pro** : See description in previous subsection.

7.3.2 From mass map κ to shear maps γ_1, γ_2

↬ **kappa_to_gamma.pro** : Derive a shear maps structure from a mass map structure using the relation 2.7 in chapter 2.

USAGE : kappa_to_gamma, smap, sgamma

INPUTS :

- smap = IDL structure of mass map with the following fields :
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = (IDL array) mass map κ

KEYWORD :

- cat = (string) if set, some fields (specific to real data) are added to the output mass map structure (ng, wtot, mask, gamma_err, kappa_err, ng_eff, sigma_gamma,x1_ran, x2_ran, x1_m, x2_m)

OUTPUTS :

- sgamma = IDL structure of shear maps with the following fields :
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = (IDL array) mass map
  - gamma1, gamma2 = (IDL array) shear maps γ_1 and γ_2

if keyword_set(cat) :

- sgamma = IDL structure of mass map with the following fields :
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = (IDL array) mass map
  - gamma1, gamma2 = (IDL array) shear maps
  - ng = (IDL array) number of galaxies per pixel
  - wtot = (IDL array) weight per pixel
  - mask = (IDL array) mask of the missing data
  - gamma_err = (IDL array) measurement error in shear maps per pixel taking into

account the weight of each galaxy
- kappa_err = (IDL array) measurement error in mass map per pixel (kappa_err=gamma_err/sqrt(2.))
- ng_eff = (IDL array) effective number of galaxies per pixel taking into account the weight of each galaxy
- sigma_gamma = (int) measurement error in shear maps per pixel
- x1_ran, x2_ran = (int) exact range in degree
- x1_m, x2_m = IDL array with the exact middle position (in deg) of each pixel

7.4 Electric and Magnetic mass maps

↬ **geb_to_ke_kb.pro** : Compute the Electric and the Magnetic mass map from the shear maps : Perform a decomposition of the shear field into its 2 components : the Electric (E) component and the Magnetic (B) one. The decomposition is based on a rotation of the shear by 45° to obtain the Magnetic component. The presence of B-modes is used to test the presence of systematic errors in Weak Lensing shear maps. Indeed, Weak Lensing only produces E-modes.

USAGE : geb_to_ke_kb, gEB, mE, mB, cat= cat

INPUTS :

- gEB = IDL structure of shear maps with the following fields :
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = IDL array of the mass map
  - gamma1, gamma2 = (IDL array) shear maps γ_1 and γ_2

KEYWORD :

- cat = (string) if set, some fields (specific to real data) are added to the output mass map structure (ng, wtot, mask, gamma_err, kappa_err, ng_eff, sigma_gamma,x1_ran, x2_ran, x1_m, x2_m)

OUTPUTS :

- mE= IDL structure of the magnetic component mass map with the following fields :
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = (IDL array) mass map with simulated missing data
  - gamma1, gamma2 = (IDL array) shear maps γ_1 and γ_2

  if keyword_set(cat) :

- mE = IDL structure of the magnetic component mass map with the following fields:
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians

- delta1, delta2 = (int) pixel size in radians
- kappa = (IDL array) mass map
- gamma1, gamma2 = (IDL array) shear maps
- ng = (IDL array) number of galaxies per pixel
- wtot = (IDL array) weight per pixel
- mask = (IDL array) mask of the missing data
- gamma_err = (IDL array) measurement error in shear maps per pixel taking into account the weight of each galaxy
- kappa_err = (IDL array) measurement error in mass map per pixel (kappa_err=gamma_err/sqrt(2.))
- ng_eff = (IDL array) effective number of galaxies per pixel taking into account the weight of each galaxy
- sigma_gamma = (int) measurement error in shear maps per pixel
- x1_ran, x2_ran = (int) exact range in degree
- x1_m, x2_m = IDL array with the exact middle position (in deg) of each pixel

- mB = IDL structure of the magnetic component mass map with the following fields:
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = (IDL array) mass map with simulated missing data
  - gamma1, gamma2 = (IDL array) shear maps γ_1 and γ_2

  if keyword_set(cat) :

- mB = IDL structure of the magnetic component mass map with the following fields:
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = (IDL array) mass map
  - gamma1, gamma2 = (IDL array) shear maps
  - ng = (IDL array) number of galaxies per pixel
  - wtot = (IDL array) weight per pixel
  - mask = (IDL array) mask of the missing data
  - gamma_err = (IDL array) measurement error in shear maps per pixel taking into account the weight of each galaxy
  - kappa_err = (IDL array) measurement error in mass map per pixel (kappa_err=gamma_err/sqrt(2.))
  - ng_eff = (IDL array) effective number of galaxies per pixel taking into account the weight of each galaxy
  - sigma_gamma = (int) measurement error in shear maps per pixel
  - x1_ran, x2_ran = (int) exact range in degree
  - x1_m, x2_m = IDL array with the exact middle position (in deg) of each pixel

7.5 Missing data

7.5.1 Add a hole in order to simulate missing data

↬ **add_hole.pro** : Add a hole to the mass map : Add a square hole in the data (pixels are set to zero) to simulate missing data. To add more than one hole, we can use the procedure in an iterative way, adding holes one by one.

USAGE : add_hole, smap, xh, yh, sh, mask,smaph

INPUTS :

- smap = IDL structure of mass map with the following fields :
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = IDL array of the mass map
- xh, yh = (int) hole position (center)
- sh = (int) square size (half side)

OUTPUTS :

- mask = (IDL array) mask to hide missing data in the mass map
- smaph = IDL structure of mass map with simulated missing data with the following fields :
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = (IDL array) mass map with simulated missing data

7.5.2 How to overcome this problem?

↬ **whole.pro** : Compute the multi-resolution mask : In some cases in multi-resolution filterings, a multiscale mask data mask can be required to remove all the impact of the missing data. All the pixels in the multi-resolution mask can be distorted by the presence of the hole or by the edge of the image.

USAGE : whole, mask, ny, wmask

INPUTS :

- mask = (IDL array) mask of the missing data in the mass map
- ny = (int) number of scales used in the wavelet transform

OUTPUTS :

- wmask = (3D IDL array) multi-resolution mask of the missing data in the mass map

7.6 Filtering

7.6.1 Gaussian Filtering

↬ **rec_kap_gaus.pro** : Perform a Gaussian Filtering to filter a noisy mass map : Perform a Gaussian filtering by calculating the convolution between the noisy mass map and a gaussian window.

USAGE : rec_kap_gaus, map, sigma, mapg

INPUTS :

- map = IDL array of a noisy mass map
- sigma = (int) the width (σ) of the Gaussian window

OUTPUTS :

- mapg = IDL array of a filtered mass map by Gaussian filtering

7.6.2 Wiener Filtering

↬ **rec_kap_wiener.pro** : Perform a Wiener Filtering to filter a noisy mass map : Perform a Wiener filtering (classical 1D method). Build the Wiener weight function by computing a weight for each ring (7) of the image.

USAGE : rec_kap_wiener, smap, sigmae, ng, mapw

INPUTS :

- smap = IDL structure of a noisy mass map with the following fields :
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = IDL array of a noisy mass map
- sigmae = (int) shear error measurement (a common value is 0.3)
- ng = (int) number of galaxies per pixel (ng is equal to 20 $gal/amin^2$ for ground observations and 100 $gal/amin^2$ for space observations)

OUTPUTS :

- mapw = IDL array of a filtered mass map by Wiener filtering

7.6.3 Multiscale Entropy Filtering

↬ **rec_kap_wl.pro** : Perform a Multiscale Entropy Filtering described in chapter 5 to filter a noisy mass map : Apply the Multi-Resolution Filtering using the Multiscale Entropy concept and the False Discovery Rate (FDR) to derive robust detection levels in wavelet space.

USAGE : rec_kap_wl, map, mapwl, Opt='-n7 -k -I5 -C2 -c2. -s0.05 -F3 -K

INPUTS :

- map = (IDL array) noisy mass map

KEYWORDS :
Opt: string which contains the different options. Options are:

[-n number_of_scales]
Number of scales used in the multiresolution transform
default is 4.

[-F first_detection_scale]
First scale used for the detection
default is 1.

[-g sigma]
sigma = noise standard deviation assuming a Gaussian noise
by default, the standard deviation is automatically estimated.

[-k]
Suppress isolated pixels in the support. Default is no.

[-K]
Remove the smoothed plane. Default is no.

[-I NbIter]
Number of iterations in an iterative process of reconstruction
default is 10.

[-s NSigma]
Thresolding at NSigma * SigmaNoise at each scale
default is 3.
FDR-Thresolding NSigma = $alpha_0$
default value is 0.05.

[-C Thresh_Type]
Thresh_Type = 1 : Use a NSigma * SigmaNoise thresholding
Thresh_Type = 2 : Use a FDR Thresholding

default is 1.

[-c Alpha_Variation]
If Thresh_Type = 2, use a different alpha per band.
Choose a Alpha_Variation value range between 1.7 and 2.
[-P]
Apply the positivity constraint
default is no.

[-R RMS_Map_File_Name]
RMS Map

[-r]
rms map is automatically calculated

[-v]
Verbose.
default is no.

OUTPUTS :

- mapwl = IDL array of the filtered mass map by Multiscale Entropy filtering

EXTERNAL CALL :

- wl_t2_filter (C++ program)

7.7 Tools

7.7.1 Characterization

► Error per scale
↬ **run_sigma.pro** : Compute an error per scale : Compute the error between the original mass map and the filtered one for each scale taking into account the edges.

USAGE : run_sigma, w, ny, wn, sigma_n

INPUTS :

- w = (3D IDL array) wavelet transform of the original image
- ny = (int) number of scales used in the wavelet transform
- wn = (3D IDL array) wavelet transform of the filtered image

OUTPUTS :

- sigma_n= error per scale

↬ **run_sigma_hole.pro** : Compute an error per scale for a map with missing data : Compute the error between the original mass map and the filtered one for each scale taking into account the edges and the missing data.

USAGE : run_sigma_hole, wmask, w, ny, wn, sigma_n

INPUTS :

- wmask = (3D IDL array) multi-resolution hole mask
- w = (3D IDL array) wavelet transform of the original image
- ny = (int) number of scales used in the wavelet transform
- wn = (3D IDL array) wavelet transform of the filtered image

OUTPUTS :

- sigma_n= error per scale

► Cluster detection

Another important aspect of the weak shear mass reconstruction is the possibility to detect clusters and to build a catalog. Thanks to isophote map overplotted on the true mass map as contours, we can check visually the false detections.

↬ **isophot_gaus_rms.pro** : Compute an isophote map for Gaussian filtering : In the Gaussian isophote maps, the isophotes corresponds to a $k\sigma$ detection level where k = 3, 4, 5.

USAGE : isophot_gaus_rms, map, sigma, rms_map, mapg, isog

INPUTS :

- map = IDL array of a noisy mass map
- sigma = (int) the width (σ) of the Gaussian window
- rms_map = (IDL array) rms map, if $rms = cte$ map of constant value

OUTPUTS :

- mapg = IDL array of a filtered mass map by Gaussian filtering
- isog = (IDL array) isophote map for Gaussian filtering

↬ **isophot_wiener_rms.pro** : Compute an isophote map for Wiener filtering : In the Wiener isophote maps, the isophotes corresponds to a $k\sigma$ detection level where k = 3, 4, 5.

USAGE : isophot_wiener_rms, smap, rms_map, sigmae, ng, mapw, isow

INPUTS :

- smap = IDL structure of a noisy mass map
- rms_map = (IDL array) rms map, if $rms = cte$ map of constant value

- sigmae = (int) measurement error in $\gamma_{i,j}$ per pixel
- ng= (int) number of galaxies per pixel

OUTPUTS :

- mapw = IDL array of a filtered mass map by Wiener filtering
- isow = (IDL array) isophote map for Wiener filtering

↬ **isophot_fdrl.pro** : Compute an isophote map for Multiscale Entropy filtering : Using the FDR-thresholding in wavelet space, we detect as significant a set of wavelet coefficients. Then, we built an isophote map, where each isophote level corresponds to the detection level in a given scale. A cluster surrounded by two isophotes means that it has been detected at two scales.

USAGE : isophot_fdrl, smap, ny, ground=ground, space = space, min_scale, iso_fdr

INPUTS :

- smap = IDL structure of a noisy mass map
- ny = (int) number of scales used in the wavelet transform

KEYWORD :

- ground = (string) if set, we compute the fdr-threshold for ground observations
- space = (string) if set, we compute the fdr-threshold for space observations

OUTPUTS :

- min_scale = (IDL array) map with the minimum scale of detection for each pixel
- iso_fdr = (IDL array) isophote map with the maximum detection for each pixel for Multiscale Entropy filtering

7.7.2 Plots

▶ Plot the shear map

↬ **plt_shear.pro** : Plot a shear map from a mass map or shear maps. Overplot the shear on the corresponding mass map.

USAGE : plt_shear, smap, kappa = kappa, gamma = gamma

INPUTS :

- smap = IDL structure of a mass map or shear maps

KEYWORD :

- kappa = (string) is set if mass map input

- gamma = (string) is set if shear maps input

OUTPUTS :

- overplot the shear field over the mass map

▶ Plot galaxies position from a shear catalogue

↬ **plt_xy_gcat.pro** : Plot for each galaxy of the catalogue a cross to give the position.

USAGE : plt_xy_gcat, gcat

INPUTS :

- gcat = IDL structure of a shear catalogue with the following fields :
  - x, y = (IDL array) coordinates in pixel of each galaxy
  - gamma1, gamma2 = (IDL array) shear maps γ_1 and γ_2
  - pixscale = (int) pixel size [rad]

OUTPUTS :

- plot the galaxies position of the catalogue

▶ Plot a mass map field (specific to real data)

↬ **plt_kappa.pro** : Plot the mass map field using the real dimensions [in rad] and overplot the snr contours.

USAGE : plt_kappa, smap, contours = contours

INPUTS :

- smap = IDL structure of a mass map with the following fields :
  - n1, n2 = (int) map size in pixels
  - theta1, theta2 = (int) map size in radians
  - delta1, delta2 = (int) pixel size in radians
  - kappa = (IDL array) mass map
  - gamma1, gamma2 = (IDL array) shear maps
  - ng = (IDL array) number of galaxies per pixel
  - wtot = (IDL array) weight per pixel
  - mask = (IDL array) mask of the missing data
  - gamma_err = (IDL array) measurement error in shear maps per pixel taking into account the weight per galaxy
  - kappa_err = (IDL array) measurement error in mass map per pixel (kappa_err=gamma_err/sqrt(2.))
  - ng_eff = (IDL array) effective number of galaxies per pixel taking into account the weight of each galaxy
  - sigma_gamma = (int) measurement error in shear maps of each pixel
  - x1_ran, x2_ran = (int) exact range in degree
  - x1_m, x2_m = IDL array with the exact middle position (in deg) of each pixel

KEYWORD :

- contours = (string) if set we overplot the snr contours

OUTPUTS :

- plot the κ field using the real dimensions and overplot the snr contours (if contours is set)

▶ Plot an image scaled to the current window

↬ **plt_image.pro** : Plot an image scaled to the current window. Further plotting (such as contours) can be performed over the resulting plot. Optionally, an annotated color bar can be drawn at the top. The 'scalable' keyword must be invoked when outputing to a postscript file.

USAGE : plt_image, map, frame = frame, colbar=colbar, cran=cran, title=title, xtitle=xtitle, ytitle=ytitle, ctitle=ctitle, inverse=inverse, scalable=scalable, csize=csize

INPUTS :

- map = (IDL array) an image

KEYWORD :

- frame = (string) if set, draw a frame with pixel index limits
- colbar = (string) if set, draw color bar
- cran = (string) change the range (default: [min(map),max(map)])
- title = (string) set a title for the plot
- x,ytitle = (string) set titles for the frame
- ctitle = (string) set a title for the colorbar
- inverse = (string) if set, invert the color coding
- scalable = (string) if set, use scalable pixels which is to be ps devices (default: nonscalable to be used with x-term device)
- csize = (string) vertical size of the color bar (0-1, default:.12)

OUTPUTS :

- plot of the scaled image with, optionally, a coordinate frame and a color bar.

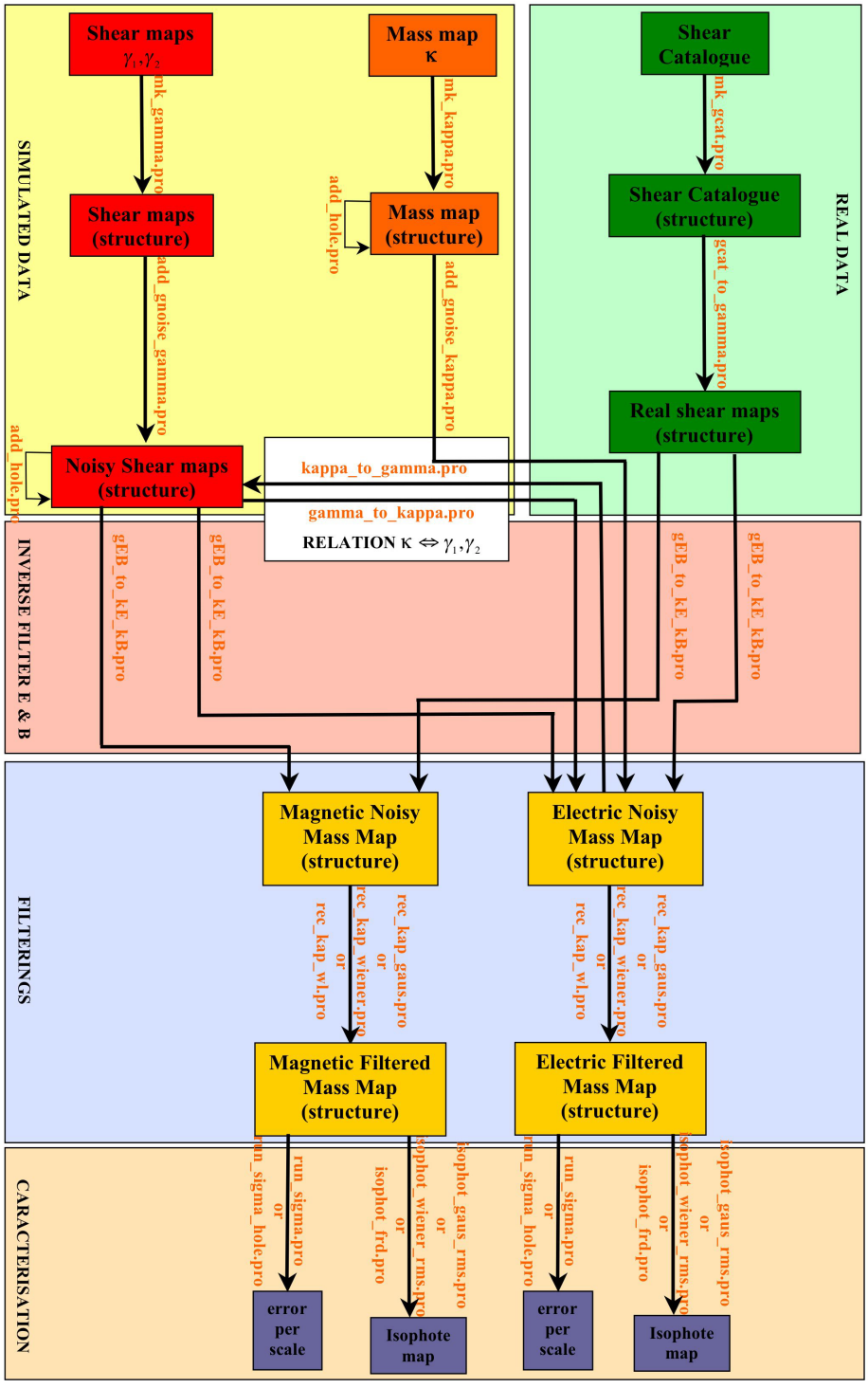


Figure 7.1: software MRL flowchart

7.8 Conclusion

We have now at your disposal all the tools to process Weak Lensing data. In the flowchart in Fig.7.1, the different areas stand for the different main processings. All the routines write down in this flowchart are described in the IDL routines section.

In the yellow area, we have the specific routines to simulated data and in the green area, we have the ones that are specific to real data. The red area is dedicated to the building of the Electric and Magnetic mass map. The white area stands for the relation and the inverse relation between the mass map and the shear maps. Finally the blue area represents the filtering step (Gaussian filtering, Wiener Filtering, Multiscale Entropy Filtering). Optionally, we can add a characterization step.

Depending on the kind of data at your disposal, one can take a different path in the flowchart. And then you run one by one the different routines.

Bibliography

Bacon, D., Refregier, A., and Ellis, R.: 2000, *MNRAS* pp 318–625

Bartelmann, M.: 1995, *Astronomy and Astrophysics* **303**, 643

Bartelmann, M. and Schneider, P.: 1999, *ArXiv Astrophysics e-prints*

Benjamini, Y. and Hochberg, Y.: 1995, *J. R. Stat. Soc. B* **57**, 289

Berge, J.: 2005, *An introduction to shapelets based - weak lensing image processing (Manual) - http://www.astro.caltech.edu/ jberge/shapelets*

Bridle, S. L., Hobson, M. P., Lasenby, A. N., and Saunders, R.: 1998, *MNRAS* **299**, 895

Donoho, D. L. and Elad, M.: 2003, *the Proc. Nat. Aca. Sci.* **100**, 2197

Gull, S. and Skilling, J.: 1991, *MEMSYS5 Quantified Maximum Entropy User's Manual*, Royston,England

Holschneider, M., Kronland-Martinet, R., Morlet, J., and Tchamitchian, P.: 1989, *Wavelets : Time-Frequency Methods and Phase-Space* pp 286–297

Hopkins, A. M., Miller, C. J., Connolly, A. J., Genovese, C., Nichol, R. C., and Wasserman, L.: 2002, *Astronomical Journal* **123**, 1086

Kaiser, N. and Squires, G.: 1993, *ApJ* **404**, 441

Kaiser, N., Wilson, G., and Luppino, G.: 2000, *Astrophysical Journal* pp 318–625

Maisinger, K., Hobson, M. P., and Lasenby, A. N.: 2004, *MNRAS* **347**, 339

Marshall, P. J., Hobson, M. P., Gull, S. F., and Bridle, S. L.: 2002, *MNRAS* **335**, 1037

Mellier, Y.: 1999, *Annual Review of Astronomy and Astrophysics* **37**, 127

Miller, C. J., Genovese, C., Nichol, R. C., Wasserman, L., Connolly, A., Reichart, D., Hopkins, A., Schneider, J., and Moore, A.: 2001, *Astronomical Journal* **122**, 3492

Murtagh, F., Starck, J.-L., and Bijaoui, A.: 1995, *Astronomy and Astrophysics, Supplement Series* **112**, 179

Narayan, R. and Nityananda, R.: 1986, *Annual Review of Astronomy and Astrophysics* **24**, 127

Pantin, E. and Starck, J.-L.: 1996, *Astronomy and Astrophysics, Supplement Series* **315**, 575

Pires, S., Juin, J.-B., Yvon, D., Moudden, Y., Anthoine, S., and Pierpaoli, E.: 2005, *Astronomy and Astrophysics*

Refregier, A.: 2003a, *MNRAS* **338**, 35

Refregier, A.: 2003b, *Annual Review of Astronomy and Astrophysics* **41**, 645

Refregier, A. and Bacon, D.: 2003, *MNRAS* **338**, 48

Starck, J.-L. and Murtagh, F.: 2002, *Astronomical Image and Data Analysis*, Springer-Verlag

Starck, J.-L., Murtagh, F., Querre, P., and Bonnarel, F.: 2001, *Astronomy and Astrophysics* **368**, 730

Starck, J.-L., Pires, S., and Refregier, A.: 2005, *Astronomy and Astrophysics*

Vale, C. and White, M.: 2003, *ApJ* **592**, 699

Van Waerbeke, L., Mellier, Y., Erben, T., Cuillandre, J., and Bernardeau, F. e. a.: 2000, *Astronomy and Astrophysics* p. 318:30

Walsh, D., Carswell, R., and Weymann, R.: 1979, **279**, 381

Wittman, D., Tyson, J., Kirkman, D.and DellAntonio, I., and Bernstein, G.: 2000, p. 405:143