Weak Lensing Mass Map Reconstruction

Technical Report

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

DAPNIA/SEDI, CEA-Saclay, 91191 Gif-sur-Yvette, France

July 2004 - Version 1.0

Contents

1 Presentation of the problem

1.1 Introduction

Weak Gravitational Lensing provides a unique method to map directly the distribution of dark matter in the universe. 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. This method is now widely used to map the mass of clusters and superclusters of galaxies.

Ongoing efforts are made to improve the detection of cosmic shear on existing telescopes and future instruments dedicated to survey cosmic shear are being 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 this shear map is converted into a map of the projected mass κ, the mass map obtained is blurred by the noise.

In this report, we have tested different methods of denoising in order to filter the noisy mass map.

1.2 The weak shear maps and the mass map

The relation between the weak shear maps γ_1, γ_2 and the mass map κ are given by :

$$\begin{aligned} \gamma_1 &= (\partial_1 - \partial_2)\psi \\ \gamma_2 &= 2\partial_1\partial_2\psi \end{aligned} \tag{1}$$

where ψ is the potential function defined by $\kappa = \nabla^2\psi$.

Therefore, we have in the Fourier space:

$$\begin{aligned} \hat{\kappa}(k_1,k_2) &= k^2\hat{\psi}(k_1,k_2) \\ \hat{\gamma_1}(k_1,k_2) &= \frac{k_1^2-k_2^2}{k^2}\hat{\kappa}(k_1,k_2) \\ \hat{\gamma_2}(k_1,k_2) &= \frac{2k_1k_2}{k^2}\hat{\kappa}(k_1,k_2) \end{aligned} \tag{2}$$

with $k^2 = k_1^2 + k_2^2$.

There is a degeneracy in the first equation when $k_1^2 = k_2^2$ and in the second when $k_1 = 0$ or $k_2 = 0$. Using both equations, there is a degeneracy only when $k_1 = k_2 = 0$. The mean value of κ cannot be recovered from the γ_1 and γ_2 maps. Noting $\hat{P_1}(k_1,k_2) = \frac{k_1^2-k_2^2}{k^2}$ (with $\hat{P_1}(k_1,k_2) = 0$ when $k_1^2 = k_2^2$) and $\hat{P_2}(k_1,k_2) = \frac{2k_1k_2}{k^2}$ (with $\hat{P_2}(k_1,k_2) = 0$ when $k_1 = 0$ or $k_2 = 0$), the mass reconstruction consists in searching κ such that :

$$\begin{aligned} \gamma_1 &= P_1 * \kappa \\ \gamma_2 &= P_2 * \kappa \end{aligned} \tag{3}$$

Furthermore, the data are noisy, and the relation between the data γ_{1b}, γ_{2b} and κ_b are given by:

$$\begin{aligned} \gamma_{1b} &= P_1 * \kappa_b + N_1 \\ \gamma_{2b} &= P_2 * \kappa_b + N_2 \end{aligned} \tag{4}$$

where N_1 and N_2 can be considered as Gaussian noise with a standard deviation $\sigma_n = \frac{\sigma_\epsilon}{N_g}$, where $\sigma_\epsilon = 0.3$, and $N_g = n_g A$. A is the surface of the pixel and n_g is the number of galaxies per arcmin^2.

In this report, we have simulated the noise only in two cases :

- for spatial observations where $n_g = 100$ gal/arcmin^2,
- for ground observations where $n_g = 20$ gal/arcmin^2.

Then we can easily derive the noisy mass map κ_b by inverse filtering by noticing that $\hat{P_1^2} + \hat{P_2^2} = 1$. Then we have in the Fourier domain :

$$\hat{\kappa_b} = \hat{P_1}.\hat{\gamma_{1b}} + \hat{P_2}.\hat{\gamma_{2b}} \tag{5}$$

1.3 Our Data

1.3.1 Simulated Data

In this survey, we have at our disposal a simulated mass map κ given by Jain Bhuvnesh from the university of Pennsylvania. The field is 3.3*3.3 degrees containing 2048*2048 pixels We can visualize a sector of this mass map in Fig.1.

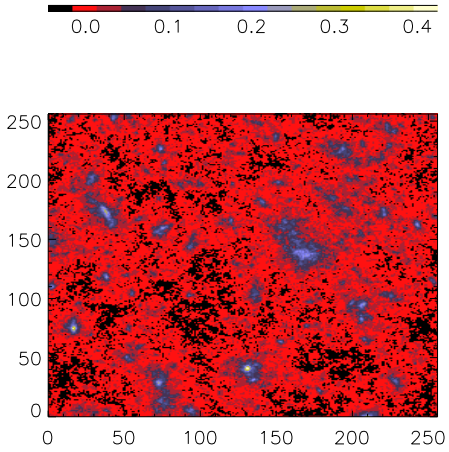

Figure 1: Simulated kappa map without noise

In order to test our methods we have collected simulated Weak Lensing mass maps on Internet. These maps derived from ray tracing through large N-body simulations of the formation and evolution of large-scale struture. The field is 2*2 degrees containing 2048*2048 pixels. Fig.2 we have a sector of one of these mass maps.

Using the equation (3), we can derive the γ_1 and γ_2 maps from bhuvnesh. And we can derive the shear maps, by creating the vectors defined by the amplitude $\sqrt{\gamma_1^2 + \gamma_2^2}$ and the angle $0.5 * \arctan \frac{\gamma_2}{\gamma_1}$.
Fig. 3 shows the weak shear maps obtained from the simulated mass map.

Then we have added a Gaussian noise to these shear maps in order to simulate the shear maps obtained with the telescopes.

By inverse filtering, using the equation (5) the mass map with noise κ_b is obtained in the Fourier domain. The solution obtained by inverse Fourier transform is presented in Fig. 4.

In this report, we present the different filterings that we have tested on the mass map with noise κ_b in order to recover the real mass map κ. Thanks to the simulated mass map (without noise) κ, we can easily compare the results of those filterings. For each filtered image, we have

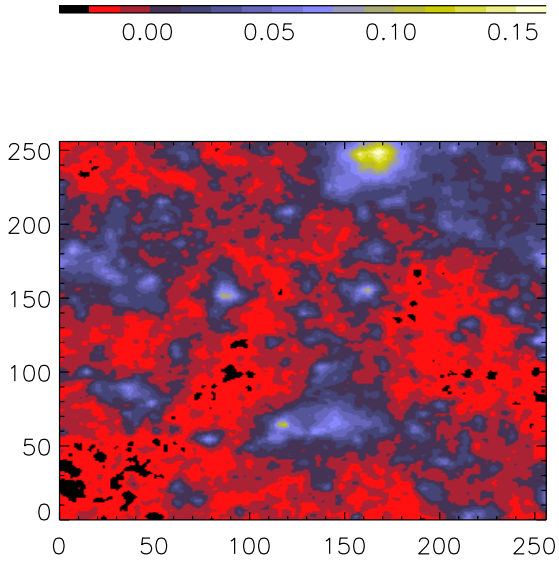

Figure 2: Another simulated mass map without noise obtained on Internet

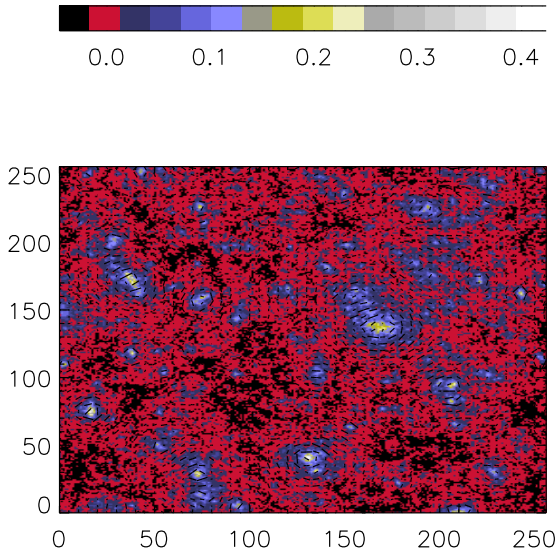

Figure 3: Shear map without noise

calculated the error as the standard deviation between the mass map κ and the filtered mass map κ_f :

$$\varepsilon = \sqrt{< (\kappa - \kappa_f)^2 >}$$

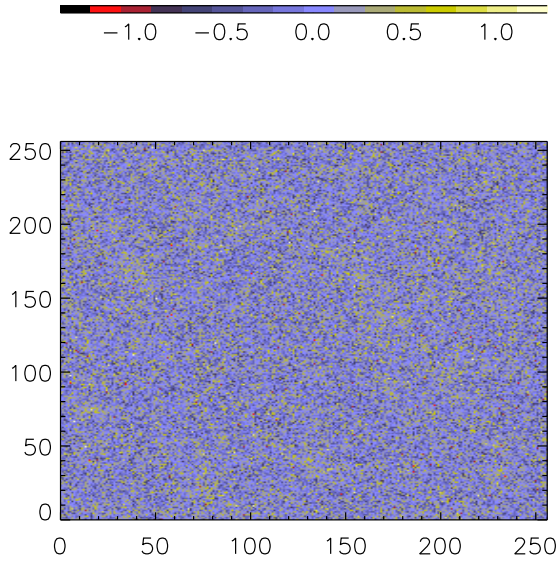

Figure 4: Noisy mass map κ_b with $n_g = 100$

1.3.2 Real ones

We have at our disposal two galaxies catalogues obtained by space observations of the HST (Hubble Space Telescope). The field is 0.17*0.28 degrees. The number of galaxies per $arcmin^2$ is rather weak, only $n_g = 65$ because exposure time is not sufficient. Fig.5, we can visualise a galaxies catalogue (the north sector). From shear catalogue, we have to make a pixelised map with a rather large number of galaxies per pixel. Then we have to adapt all the filterings at these real noisy mass maps.

1.4 Isophots

In order to make sure that the structures which we detect are not only noise, we have developped a procedure to plot isophots (level lines) on level 3σ, 4σ and 5σ over the noisy mass map or after filterings.
This procedure consists in seeking in the noisy mass map the maximum of detection for each pixel in each scale and keeping the maximum value for each pixel among the scales. We plot then level lines on level 3σ, 4σ and 5σ.
After the filtering of the noisy mass map, we will overplot the curve of the isophots on the filtered mass map. We know then which confidence we must give to the structures detected. Fig.6, we can visualise the level lines of noisy mass map ($n_g = 100$) overplot on the mass map (without noise).

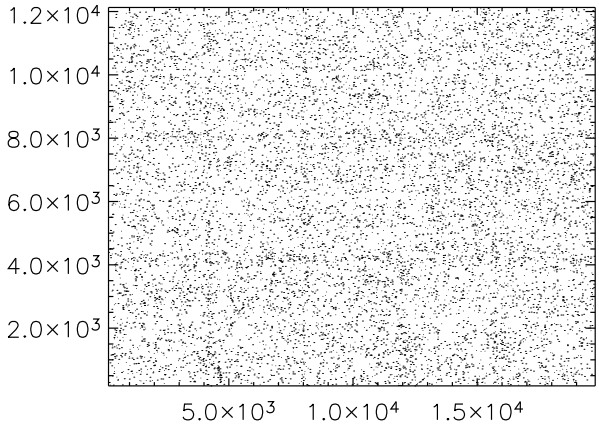

Figure 5: Galaxies catalogue - north sector

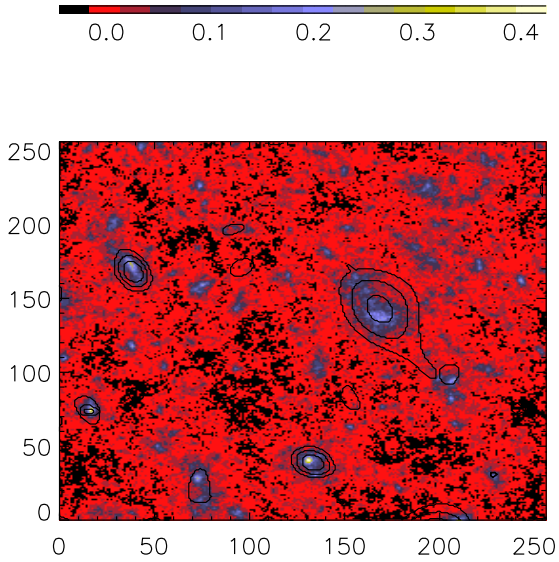

Figure 6: Isophots of the noisy mass map ($n_g = 100$) on the mass map (without noise)

2 Filterings of the noisy mass map

2.1 Filterings in Fourier space

2.1.1 The Gaussian filter

The standard method consists in convolving the noisy mass map κ_b with a Gaussian window Z :

$$\kappa_f = Z \overset{*}{,} \kappa_b \tag{6}$$

Then, it is necessary to specify the standard deviation of the Gaussian (σ). This corresponds to the width of the filter. The result of the filtering is a function of this standard deviation.

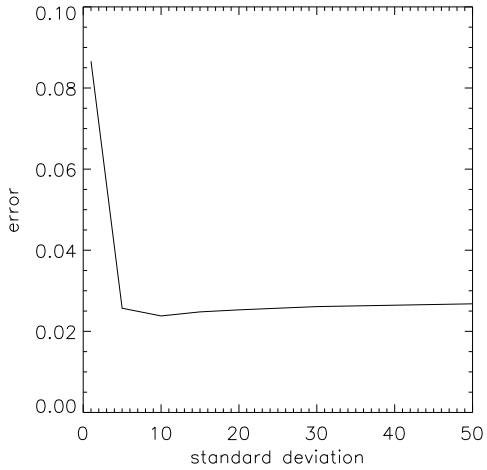

Figure 7: Error as a function of standard deviation σ with $n_g = 100$

The Fig. 7 shows the variation of the error (as it defined earlier) between the mass map κ and the filtered mass map κ_f with σ. The optimal value of σ lies between 5 and 10 for space observations ($n_g = 100$). The Fig. 8 shows the result after such processing using a Gaussian filter with $\sigma = 25$ ((a) for observations on the ground and (b) and with $\sigma = 10$ for space observations).

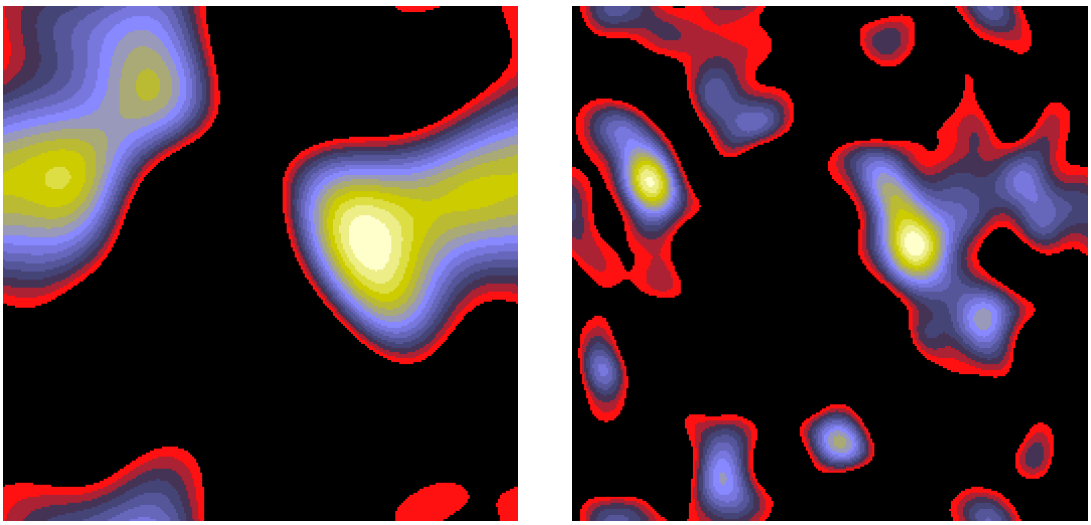

Figure 8: Gaussian filtering (left) with $\sigma = 25$ for $n_g = 20$ and with $\sigma = 10$ for (right) $n_g = 100$

The filtering is not sufficient, especially for the low frequencies. The smoothed image shows not only some structures, but also residual noise. Consequently, it is difficult to attribute any significance to these structures.

In order to filter the low frequencies too, we have tested another filter in Fourier space : the Wiener filter.

2.1.2 The Wiener filter

The Wiener filter is an optimal filter used for the removal of Gaussian noise from a noisy image. The filtering is done by multiplying the Fourier transform of the noisy image $\hat{\kappa_b}$ by the Fourier transform of the filter function $\hat{W}$. The output image κ_f is then created by taking the inverse Fourier transform of the product :

$$\kappa_f = W * \kappa_b \tag{7}$$

The first method consists in calculating the Fourier transform of the filter function $\hat{W}$ in each pixel of the image, which is given by :

$$\hat{W}(u,v) = \frac{|\hat{S}(u,v)|^2}{|\hat{S}(u,v)|^2 + |\hat{N}(u,v)|^2} \tag{8}$$

where $\hat{S}$ is the Fourier transform of the signal without noise and $\hat{N}$ that of the noise .

The filter depends on the model of the noise in the given image. This application assumes that the noise is 'white' and is constant across the image.

The second method consists in calculating the variance of the signal and that of the noise on concentric rings in the Fourier space. The Fourier transform of the filter function $\hat{W}$ is calculated in each ring and not in each pixel as in the first method. And it is given by :

$$\hat{W}(k) = \frac{< |\hat{S}(u,v)|^2 >_k}{< |\hat{S}(u,v)|^2 >_k + < |\hat{N}(u,v)|^2 >_k} \tag{9}$$

where k is the index of the rings.

The equation (9) use the signal without noise. However, the image observed by the telescope is blurred by the noise. These two variables being decorrelated the variance $< |S(u,v) + N(u,v)|^2 >$ is simply equal to the sum of the variance of the signal and that of the noise $< |S(u,v)|^2 > + < |N(u,v)|^2 >$. We then make the assumption that the noise is Gaussian. So, $< |N(u,v)|^2 >$ is known and by simple subtraction, we derive $< |S(u,v)|^2 >$.

Initially, we calculated $\hat{W}$ for circles of which the size increased linearly. But we have noticed, by plotting the curves representing the variance of the signal and that of the noise as a function of the index of the ring, that the noise is always dominating. So the result of filtering is not good. The main information (low frequencies) is in the circles of small radius.

We then considered circles of which the size increased logarithmically. We can thus have small rings in the center and rings increasingly larger as we move away from the center of the image (in Fourier space).

Fig. 9, we have plotted the variance of the signal and that of the noise as a function of the ring. We can notice that the noise is 'white' because its power spectrum is quasi-constant.

Fig. 10 shows the result of this filtering.

For comparison, we have plotted (Fig.11) the Wiener function and the Gaussian functions used in the previous Gaussian filtering, 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.

Actually, the Wiener filter is the optimal filter if the image and the noise follow a Gaussian distribution. However, this is not the case. So, in the following paragraph, we have tried another filtering method using the wavelet transform in order to improve the quality of the filtering.

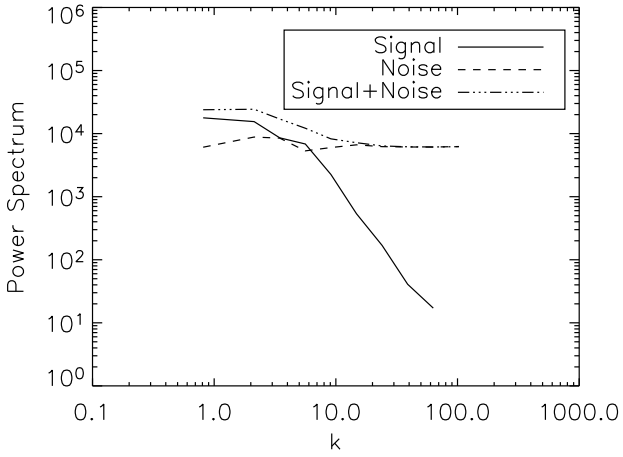

Figure 9: Power spectrum of the signal and that of the noise in each ring ($n_g = 100$)

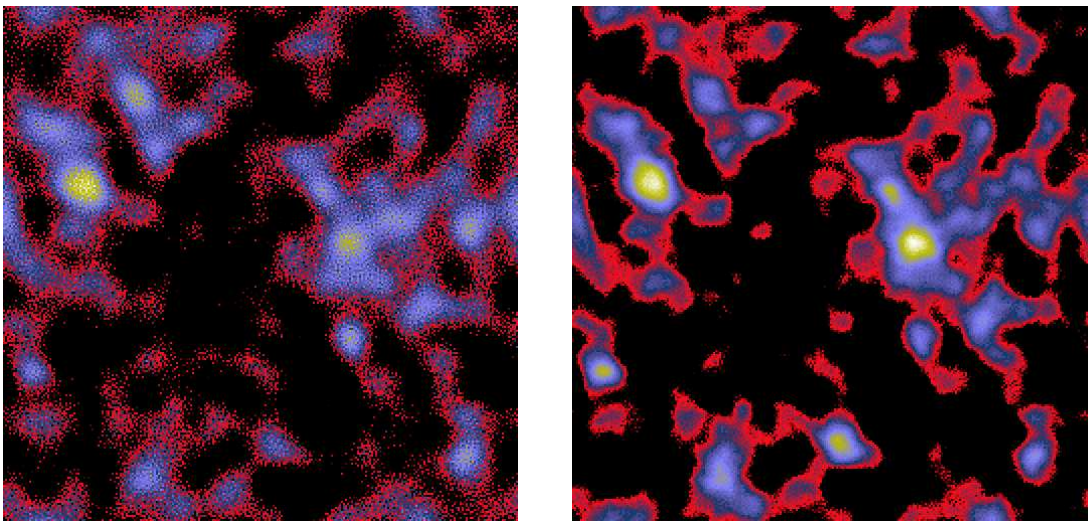

Figure 10: Filtered mass map by Wiener filter with rings increasing in a logarithmic way (left) $n_g = 20$ and (right) $n_g = 100$

2.2 Filterings in Wavelet space

2.2.1 The "à trous" isotropic Wavelet Transform

The "à trous" isotropic wavelet transform algorithm decomposes an image d of nxn pixels as a superposition of the form :

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

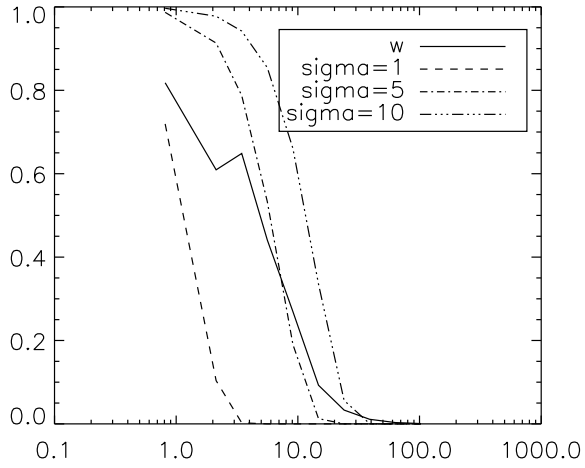

Figure 11: Comparison between the Gaussian filters and the Wiener filter ($n_g = 100$)

where c_J is a coarse or smooth version of the original signal d and w_j represents 'the details of d' at scale 2^{-j}, see [4, 3] for more information. Thus, the algorithm outputs $j+1$ subband arrays of size n. (The indexing is such that, here, $j = 1$ 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.

2.2.2 The multiresolution support and wavelet coefficient thresholding

1. Thresholding : A multiresolution support of an image describes in a logical or Boolean way if the data d contains information at a given scale j and at a given position (x, y). If $M^{(d)}(j, x, y) = 1$ (or $= true$), then d contains information at scale j and at the position (x, y).

   The multiresolution support will be obtained by detecting at each scale the significant coefficients :

$$M(j,x,y) = \begin{cases} 1 & \text{if } w_{j,k} \text{ is significant} \\ 0 & \text{if } w_{j,k} \text{ is not significant} \end{cases} \tag{10}$$

   In the case of Gaussian noise, a coefficient $w_{j,x,y}$ is significant if $\mid w_{j,x,y} \mid > k\sigma_j$, where σ_j is the noise at the scale j and k is a constant generally chosen equal to 3.

   One possible thresholding of the noisy mass map κ_b consists in setting to 0 all non significant wavelet coefficients:

$$\kappa_{ond} = \mathcal{W}_r.M.\mathcal{W}_t.\kappa_b \tag{11}$$

   where $\mathcal{W}_t$ and $\mathcal{W}_r$ are respectively the wavelet transform and the wavelet reconstruction operators.

k is a constant generally chosen equal to 3, but actually, if we keep this value in each scale, we obtain a filtered image with white spots. Indeed, the small scales (high frequencies) contain only noise discontinuities of which are higher than $3.\sigma_j$. So, for the small scales, k is chosen equal to 4 or 5. The multiresolution support M is then equal to 0 thus making it possible to remove the high frequency noise. Fig. 12 shows the result after such processing.

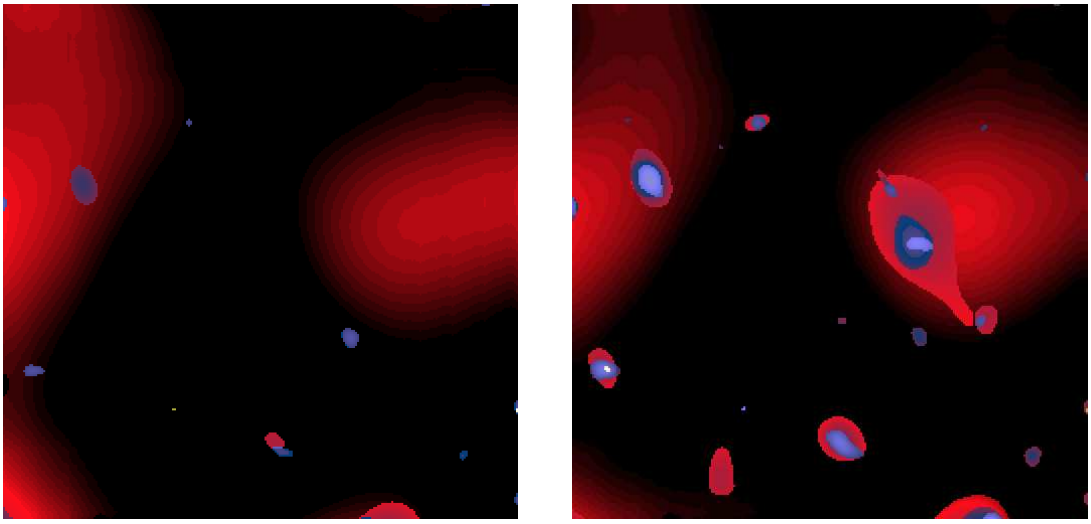

Figure 12: Mass map filtered by the multiresolution support in wavelet space ((left) $n_g = 20$ and (right) $n_g = 100$)

2. Iterative thresholding : The solution can be improved by the Van Cittert iteration [4]. This can be expressed in the following way :

$$\begin{aligned} \kappa_{ond}^{n+1} &= \kappa_{ond}^{n} + \mathcal{W}_r(M.\mathcal{W}_t.\kappa_b - M.\mathcal{W}_t.\kappa_{ond}^{n}) \\ &= \kappa_{ond}^{n} + \mathcal{W}_r(M.\mathcal{W}_t.R^n) \end{aligned} \quad (12)$$

where $R^n = d - s^n$ is the residual at the iteration n.

The Van Cittert iteration does nothing but amplify the amplitude of the already detected peaks.

In the mass map filtered by the multiresolution support in the wavelet space (fig. 12), we notice that the peaks come out well. But smoothing is too strong in the low frequencies, generating kinds of steps. This is due to the multiresolution support M which is equal either to 0 or to 1. So we have to find a support that would be smoother.

2.2.3 Wiener-like filtering in Wavelet space

After having computed the wavelet transform of the noisy mass map, we consider a wavelet coefficient $w_{j,k}$ on the scale j and we assume that its value, at a given scale and a given position, results from a noisy process, based on a Gaussian distribution. We can then use the previous Wiener function in the wavelet space. But instead of calculating the filter function for each ring, we calculate it for each scale. With a simple multiplication of the coefficient by the Wiener filter, we get a linear filter. In Fig. 13, we can visualize the power spectrum of the signal and that of the noise for each scale. We can notice that the noise is dominating at the small scales (high frequencies). It is only from 5th scale that the signal becomes more important than the noise.

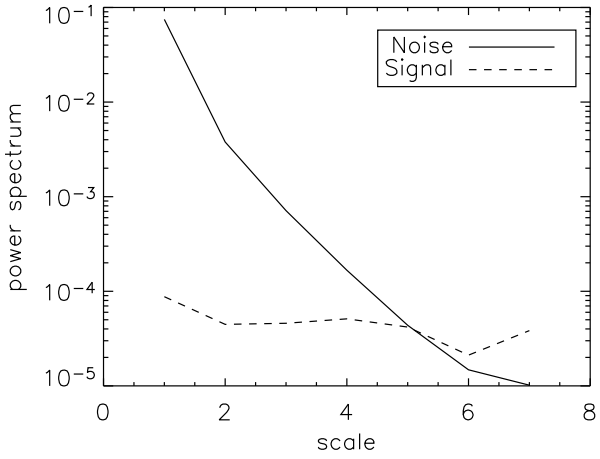

Figure 13: Power spectrum of the signal and that of the noise for each scale ($n_g = 100$)

In Fig.14, we have represented the Wiener weight as a function of the scale. It is in accordance with the multiresolution support M (seen earlier) where the first few scales are equal to 0 in order to remove the high frequency noise.

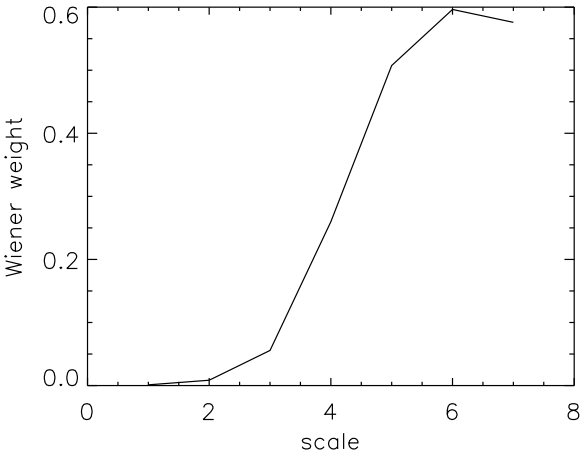

Figure 14: Wiener weight as a function of the scale ($n_g = 100$)

Fig.15 shows the result of this filtering which is close to the result of the classical Wiener filtering.

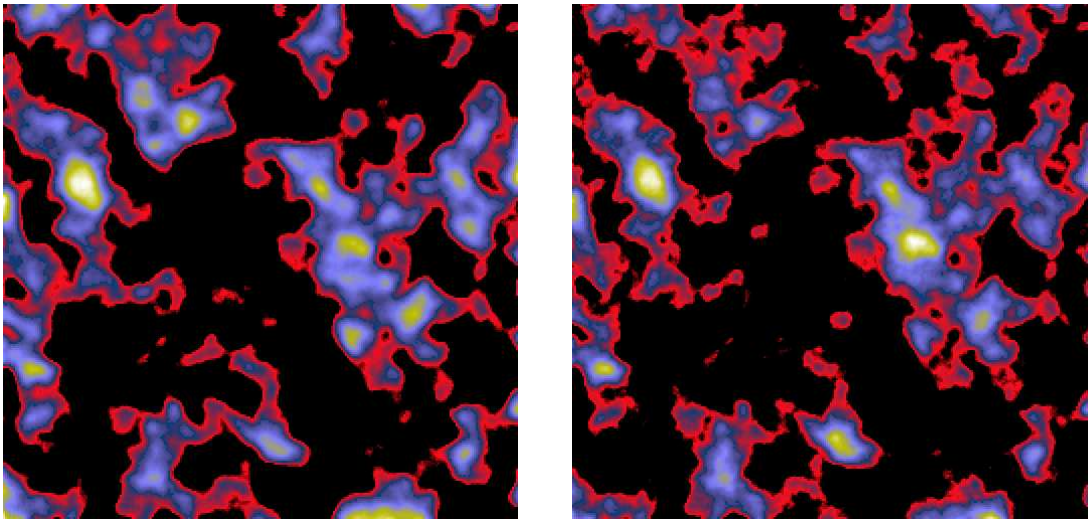

Figure 15: Mass map filtered by Wiener in the wavelet space ((left) $n_g = 20$ and (right) $n_g = 100$)

The Wiener-like filter in the wavelet space does not enable us to detect the main structures such as stars or galaxies. Furthermore , low frequencies structures are too smoothed.

This is because the Wiener filter is an optimal filter to remove the noise in a Gaussian image. But the mass map is not perfectly Gaussian. If we plot the histogram for each scale which is obtained from the wavelet transform of the mass map (Fig. 16), we notice that the histogram looks like a Gaussian function only for the first scale. Then, the more we consider a larger scale, the more the histogram looks like a Gaussian function which is less and less symmetrical.

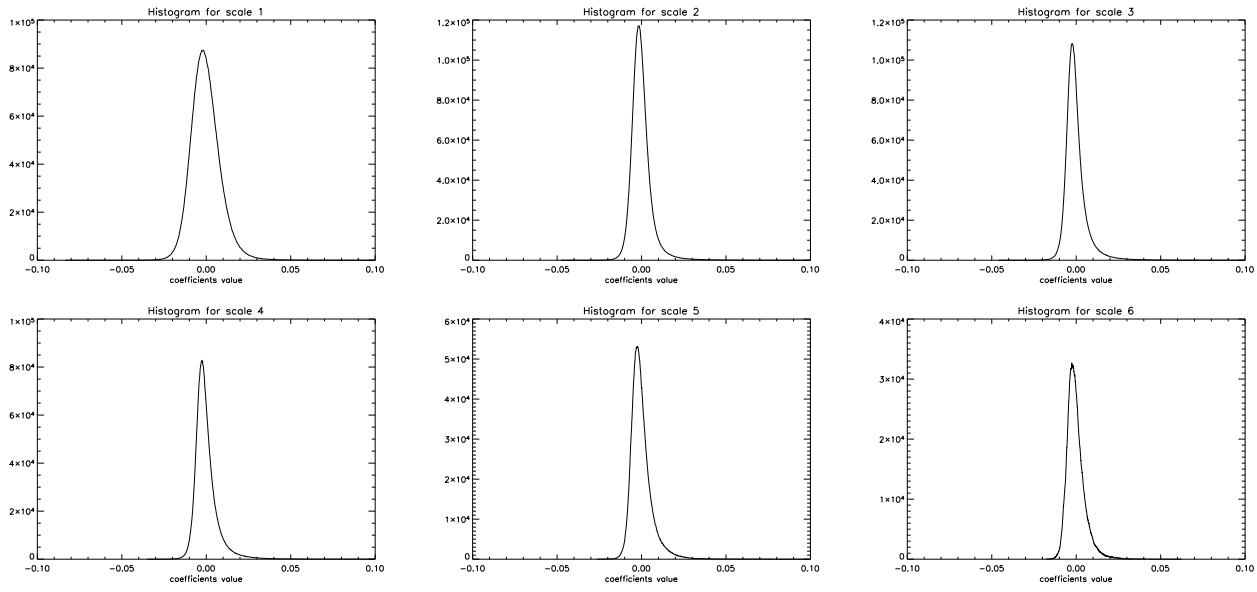

Figure 16: Histogram for scales of the wavelet transform of κ

In order to increase the amplitudes of some of the peaks of galaxies, an additive thresholding has been tested on the image obtained by Wiener-like filtering in the wavelet space.

2.2.4 Adding a thresholding

1. The Hard Thresholding : Let $\tilde{w}_{j,x,y}$ be the new wavelet coefficients at the scale j, $w^w_{j,x,y}$ the wavelet coefficients of the filtered image by Wiener in the wavelet space and $w_{j,x,y}$ the wavelet coefficients of the noisy mass map κ_b.

$$\tilde{w}_{j,x,y} = w^w_{j,x,y} + HT(w_{j,x,y} - w^w_{j,x,y}, k\sigma_j) \tag{13}$$

with :

$$HT(x,y) = \begin{cases} x & \text{if } \mid x \mid > y \\ 0 & \text{otherwise} \end{cases}$$

The Hard Thresholding consists in considering the difference between the noisy mass map and the filtered (by Wiener in the wavelet space) mass map. If the difference is higher than a threshold T (in the case of Gaussian noise, $T = k.\sigma_j$), it means that we have forgotten a main structure and this difference is added to the new image. otherwise nothing is done. Fig. 17 shows the result of this thresholding.

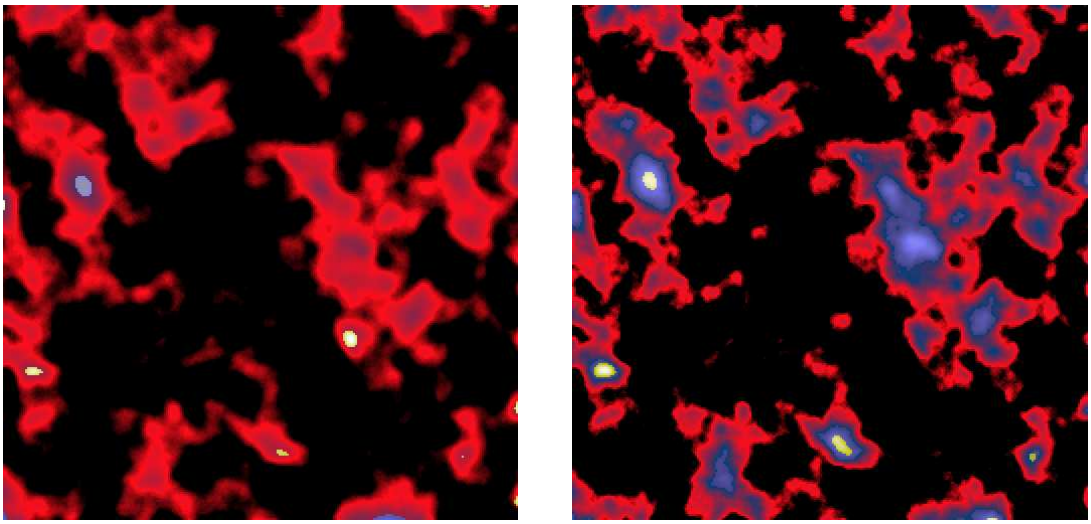

Figure 17: Filtered mass map by Wiener in wavelet space with an additional Hard Thresholding ((left) $n_g = 20$ and (right) $n_g = 100$)

2. The Soft Thresholding : The previous formula is still valid but with the ST function :

$$\tilde{w}_{j,x,y} = w^w_{j,x,y} + ST(w_{j,x,y} - w^w_{j,x,y}, k\sigma_j) \tag{14}$$

with :

$$ST(x,y) = \begin{cases} sgn(x)(\mid x \mid -y) & \text{if } \mid x \mid > y \\ 0 & \text{otherwise} \end{cases}$$

As previously, the Soft Thresholding consists in considering the difference between the noisy mass map and the filtered (by Wiener in the wavelet space) mass map. But this

time, instead of adding the mass map value to the filtered mass map, we add the difference between the two images. Furthermore , in this case, k is chosen twice as small as in the Hard Thresholding in order not to add distortions to the image. Because k is directly used for the estimation of the new wavelet coefficients.
After this thresholding, new structures which can be either noise or signal can emerge. Fig. 18 shows the result of this processing.

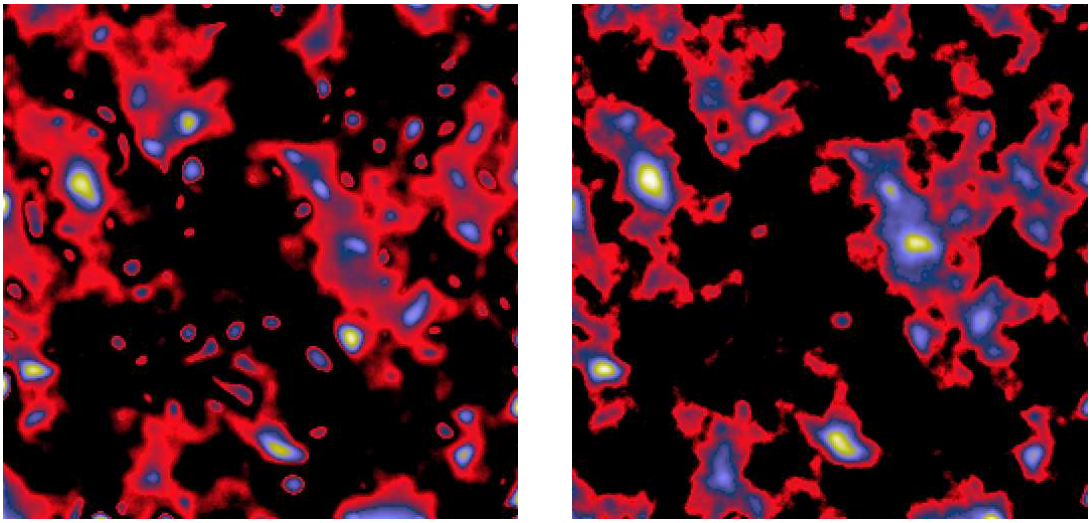

Figure 18: Filtered mass map by Wiener in wavelet space with an additional Soft Thresholding ((left) $n_g = 20$ and (right) $n_g = 100$)

Thanks to the Hard Thresholding, the brighter structures can be isolated but their boundaries are too sharp. Whereas, the Soft Thresholding leads to smoother boundaries. In order to improve the quality of the filtering a bit more, we have tested another method of filtering in the wavelet space, using the concept of entropy.

2.2.5 Multiscale Entropy Method (MEM)

The entropy can be used for the general problem of data filtering. A possibility 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 $H(X)$ the information relative to the signal and $h(w_{j,k})$ the information relative to a single wavelet coefficient, we define :

$$H(X) = \sum_{j=1}^{l} \sum_{k=1}^{N_j} h(w_{j,k}) \tag{15}$$

where l is the number of scales and N_j is the number of samples in the band (scale) j. For Gaussian noise, we get :

$$h(w_{j,k}) = \frac{w_{j,k}^2}{2.\sigma_j^2} + Cte \tag{16}$$

where σ_j is the standard deviation of the noise in the scale j.

1. Signal and noise information : The mass map is corrupted by noise so 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 is not informative for us. For each wavelet coefficient $w_{j,k}$, we have to estimate the fractions h_n and h_s of h (with $h(w_{j,k}) = h_n(w_{j,k}) + h_s(w_{j,k})$). These fractions h_n and h_s contribute respectively to the calculation of H_n and H_s.
   Hence, the signal information and the noise information are defined by :

$$H_s(X) = \sum_{j=1}^{l} \sum_{k=1}^{N_j} h_s(w_{j,k})$$
$$H_n(X) = \sum_{j=1}^{l} \sum_{k=1}^{N_j} h_n(w_{j,k}) \quad (17)$$

2. Filtering : The problem of filtering κ_b can be expressed as follows. We look for a solution κ_f such that the difference between κ_f and κ_b minimizes the information due to the signal (we want recover all the signal) and such that κ_f minimizes the information due to the noise (we want no noise). In practice, we minimize for each wavelet coefficient $w_{j,k}$:

$$l(\tilde{w}_{j,x,y}) = h_s(w_{j,x,y} - \tilde{w}_{j,x,y}) + \alpha.h_n(\tilde{w}_{j,x,y}) \quad (18)$$

   where $w_{j,x,y}$ are the wavelet coefficients of κ_b and $\tilde{w}_{j,x,y}$ the wavelet coefficients of the image filtered by MEM.
   Thanks to the parameter α, we can control the smoothness of the solution. The higher the value of α, the more the corrected wavelet coefficients are reduced. There are several kinds of regularization methods for the parameter α. To carry out this regularization, we have used an existing program (*mw_filter* in mr2 package) which we have adapted to the characteristics of the noisy mass map. In Fig. 19 we can visualize the result of this processing.

   For the observations on the ground, we obtain a filtered image with white spots like with the multiresolution support. Hence, if we compare visually the mass map filtered by MEM with the real mass map, we can say that the multiscale entropy filtering is the best filtering that we have tested.

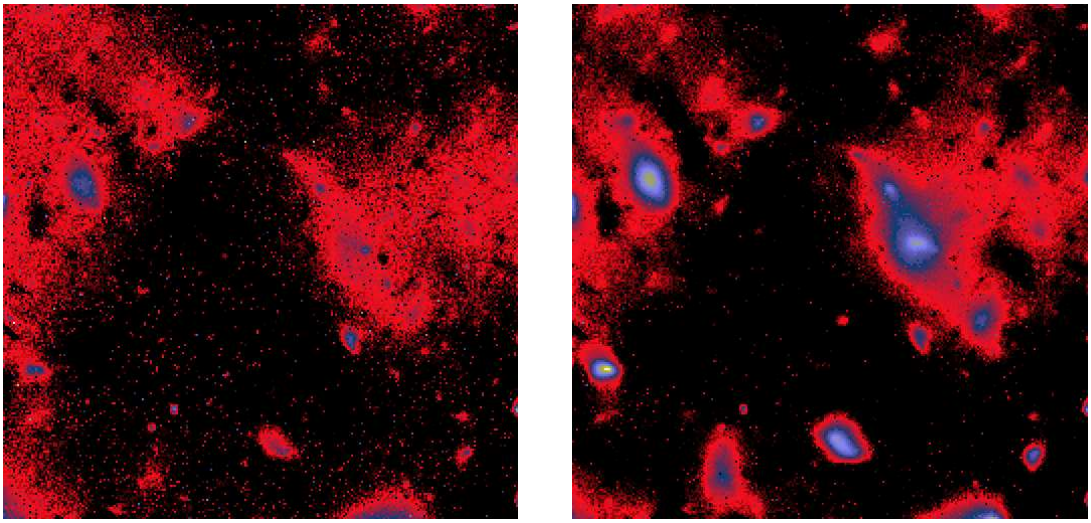

Figure 19: Mass map filtered by the Multiscale Entropy Method ((left) $n_g = 20$ and (right) $n_g = 100$)

3. Removing of the first two scales : The MEM Method always tries to keep a few coefficients even in the first scales. However, in the small scales, there are only noise. So, we have chosen to remove the first two scales. Fig. 20, shows the result of the filtering with or without the first two scales. Fig. 21, shows the difference between these two images. No structure is distinguished, there are only noise in this image. This processing gives good results.

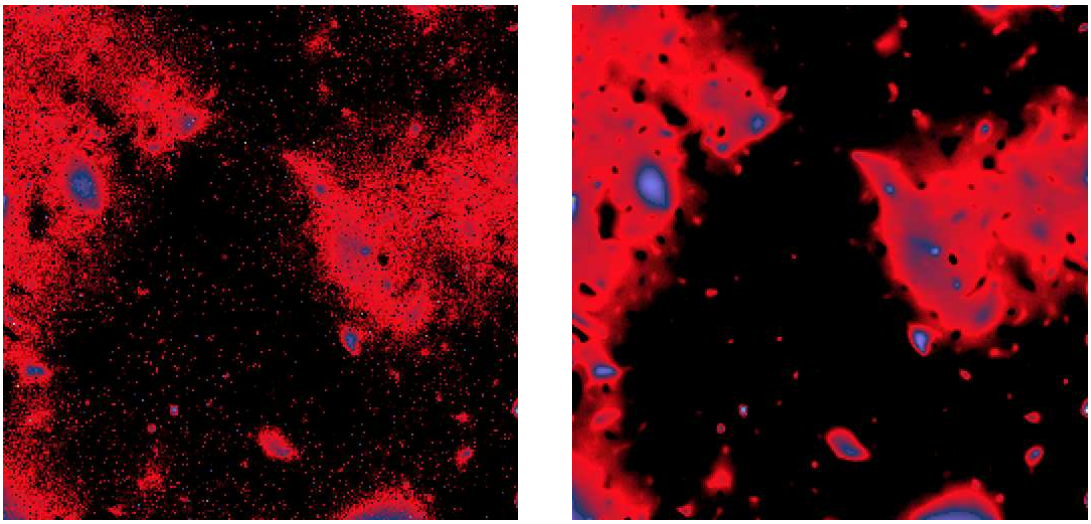

Figure 20: Mass map filtered by the Multiscale Entropy Method $n_g = 20$ ((left) with the first two scales and (b) without)

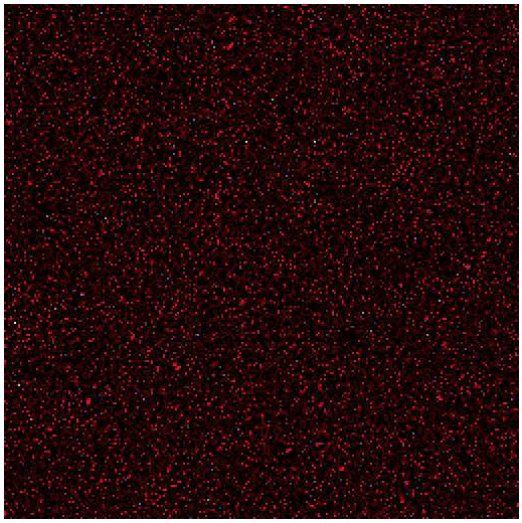

Figure 21: Difference between the previous images with and without the first two scales

4. The choice of the threshold and the FDR-Thresholding

   The threshold $k.\sigma$ is selected so that all the wavelet coefficients higher than $k.\sigma$ are untouched. Initially, we used a fixed value of k (typically defaulted to 3) for all the scales. But we have plotted the curves representing the standard deviation between the filtered mass map κ_f and the real mass map κ as a function of the number of scale, for three fixed value $k = 3$, $k = 4$ and $k = 5$ (Fig. 22) and we have noticed that according to the scale there are a given threshold value that is better than others. For example, the noise have greater variations for the larger scale.

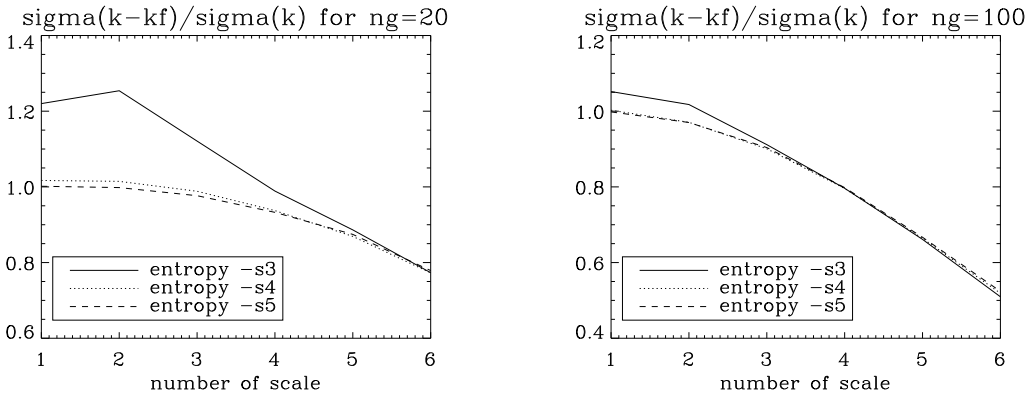

Figure 22: (left) $n_g = 20$ and (right) $n_g = 100$

Consequently, we have decided to take a different threshold value for each scale. The result seems to be good. We have plotted again the curves representing the standard deviation between the filtered mass map κ_f and the real mass map κ as a function of the number of scale for two fixed value $k = 3$ and $k = 4$ and for k variable (Fig. 23). We noticed that

the result obtained with a variable threshold is almost always better.

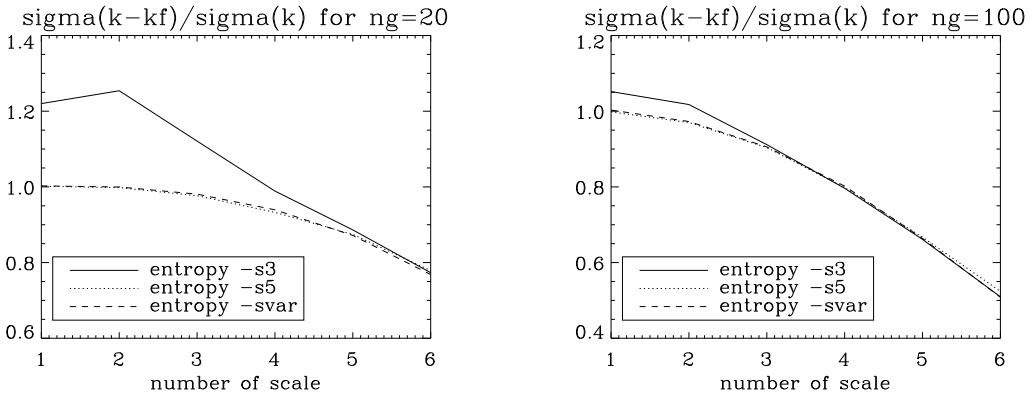

Figure 23: (left) $n_g = 20$ and (right) $n_g = 100$

But a basic problem remains : choosing the suitable threshold for a given scale for all kinds of data. Without an objective method for selecting these thresholds we have to adjust these thresholds to give desirable results each time.
The FDR (False Discovery Rate) method has been developped recently and allows us to control the average fraction of false detections made over the total number of detections performed and also offers an effective way to select thresholds that is automatically adaptative across data.
The FDR is given by the ratio :

$$FDR = \frac{V_{ia}}{D_a} \tag{19}$$

where V_{ia} are the number of pixels truly inactive declared active
and D_a are the number of pixels declared active

This procedure controlling the FDR specifies a rate α between 0 and 1 and ensures that *on average* FDR is no bigger than α.

$$E(FDR) \leq \frac{T_i}{V}.\alpha \leq \alpha \tag{20}$$

The unknown factor $\frac{T_i}{V}$ is the proportion of truly inactive pixels. For analyses of smaller interest regions, it might be useful to estimate $\frac{T_i}{V}$ and choose α accordingly.
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 :

$$d = max\{j : P_j < \frac{j.\alpha}{c_N.N}\} \tag{21}$$

where $c_N = 1$, if p_values are statistically independants.
Now, declare actived all the pixels whose p_values are less than or equal to P_d.

Graphically, this procedure corresponds to plotting the P_j versus $\frac{j}{N}$, superposing the line through the origin of slope $\frac{\alpha}{c_N}$, and finding the last point at which P_j falls below the line.

As previously, Fig. 24 shows the curves representing the standard deviation for each scale between the filtered mass map κ_f and the real mass map κ. The result obtained with the mass map filtered with arbitrary thresholds per scale is in solid line. And we have plotted in dotted lines these obtained with the fdr-thresholds per scale for different α values.
We can notice that for ground observations ($n_g = 20$), the FDR method don't really improved the standard deviation per scale but enables us to adapt the thresholds automatically. For space observations ($n_g = 100$), on the contrary, the FDR method enables us to improve considerably the standard deviation per scale and the thresholds are also automatically adaptative.

For space observations, as we can see it on Fig. 25, there is a noticeable difference between the mass map filtered with the fdr-thresholds and arbitrary variable thresholds. In fact, Fig. 26, we have plotted over the real mass map (slightly smoothed) the isophots of the two filtering results. The fdr-thresholding allows us to detect more clusters.

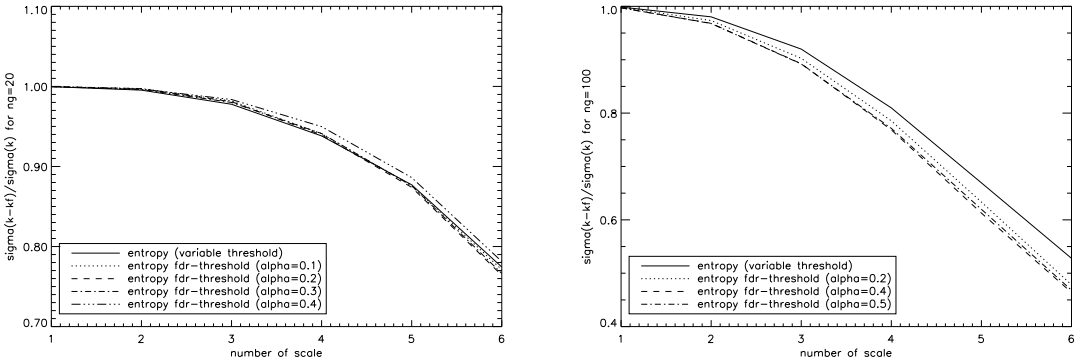

Figure 24: (left) $n_g = 20$ and (right) $n_g = 100$

5. The addition of a loop to raise the level of the filtered image

   The result of the filtering by multiresolution Entropy can be improved by an iterating process. Its goal is to recover the information lost during the reconstruction by the opposite Wavelet Transform because the Wavelet Transform is not reversible.
   This process consists in adding to the map I_i, for each iteration i, a Residual obtained by the opposite 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.
   Fig.27 on the left, we have the mass map reconstructed by MEM without an iterative process and on the right, we have the mass map reconstructed by MEM with 5 iterations for $n_g = 100$. The levels of the image of right-hand side are recovered better.

2.3 Conclusion about the filterings of the noisy mass map

We have summarized the results of each filtering in the following array, by calculating for each filtering the standard deviation between the filtered mass map κ_f and the real mass map κ.

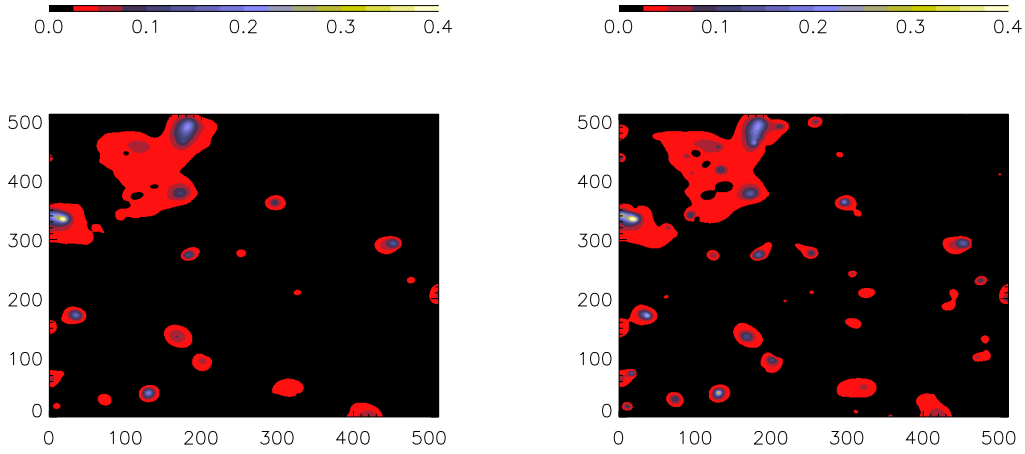

Figure 25: Kappa filtered (left) with arbitrary thresholds per scale and (right) with the fdr-thresholds per scale

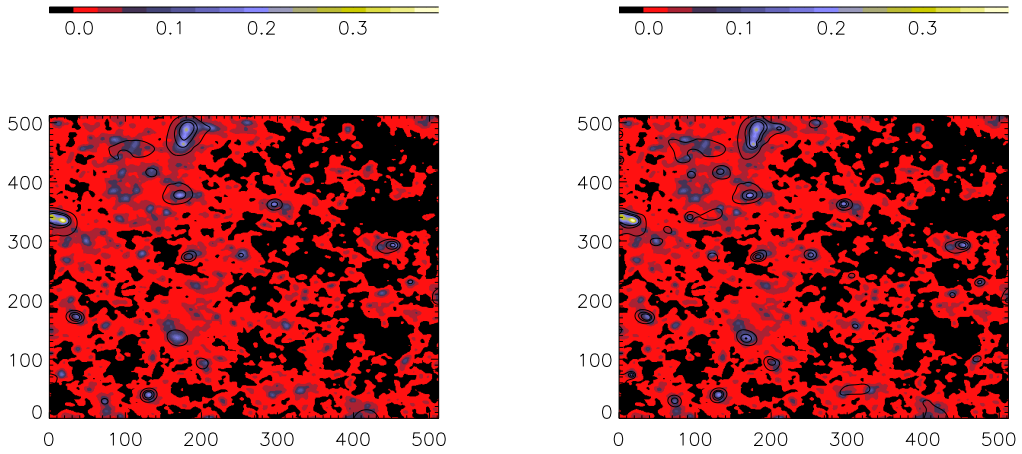

Figure 26: Isophots of the noisy mass map filtered (left) with arbitrary thresholds per scale and (right) with the fdr-thresholds per scale

filter / n_g	20	100
Gaussian Filter ($\sigma = 10$)	0.0302	0.0239
Gaussian Filter ($\sigma = 25$)	0.0278	0.0259
Wiener Filter (1D) in Fourier space	0.0261	0.0234
Multiresolution Support Filtering	0.0276	0.0236
Wiener Filter in Wavelet space	0.0260	0.0235
Additional Hard Thresholding	0.0263	0.0231
Additional Soft Thresholding	0.0258	0.0228
MEM Filter (k=5)	0.0284	0.0229
MEM Filter (k=5, without the first two scales)	0.0267	0.0227
MEM Filter (k=variable, without the first two scales)	0.0268	0.0229
MEM Filter (k=variable, without the first two scales + i=5)	0.0259	0.0227

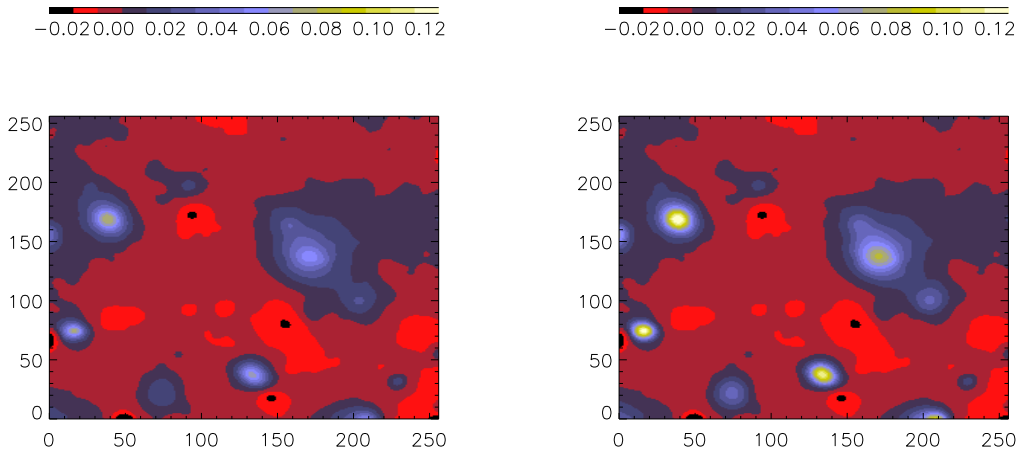

Figure 27: (left) $i = 0$ and (right) $i = 5$ for $n_g = 100$

We notice that the Wiener filter with an additional Soft Thresholding is the one that gives the filtered mass map κ_f nearest to the mass map κ according to the standard deviation estimator. Hence, the standard deviation does not seem to be the best estimator of the quality of the filtering. This is because, it makes an average on the entire image whereas we rather want to compare the images pixel by pixel. Indeed, when we estimate the quality of the filtering visually, it is the Multiscale Entropy Method which gives the best result.
The Gaussian filter is the filter that gives the coarsest result. The main peaks are found, but the edges remain vague.

The Wiener filter gives good results in the Fourier space but is better when applied in the wavelet space. In the Fourier space, the filtered image obtained by Wiener has some spots of noise. But, the additional Hard and Soft Thresholding improve the quality of the filtering. We notice that when we use the standard deviation estimator, for the observations on the ground, this improvement is not notable. This confirms what we have said earlier about the validity of the standard deviation estimator.

In order to improve the quality of the estimator, we have plotted the curves representing the standard deviation between the filtered mass map κ_f and the real mass map κ for each scale. In Fig. 28, we have compared the Gaussian filtering (σ=10), the Wiener filtering (1D), the Multiresolution Support filtering and the MEM filtering (k=variable).

According to the standard deviation per scale estimator, the Multiscale Entropy Method gives the best result. In Fig. 29, Fig. 30, Fig. 31 and Fig. 32, we have compared (image to image) the Wiener filter and the MEM filter with the mass map κ for each scale starting from the third scale. We can notice that the Wiener filter is optimal for the first scales because the signal is more or less Gaussian. For the others scales, there are more structures which arise with the Wiener filter than with the MEM filter but there are more false detections.

To conclude, we can say that the Multiscale Entropy Filtering (with a k variable and after having removed the first two scales) is the best method that we have tested as it enables us to detect all the main structures of the mass map without doing too much false detections.

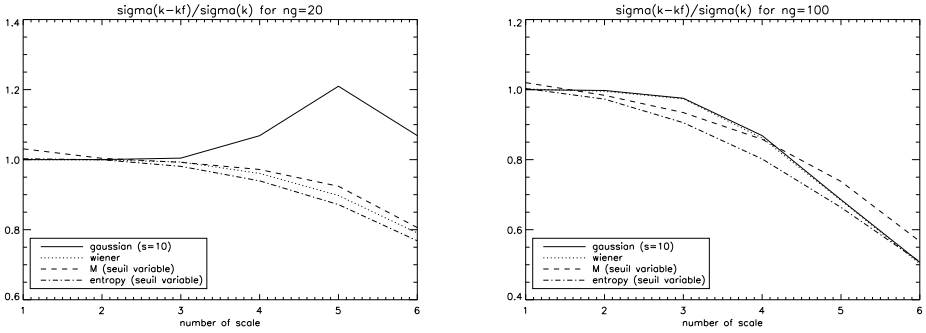

Figure 28: (left) $n_g = 20$ and (right) $n_g = 100$

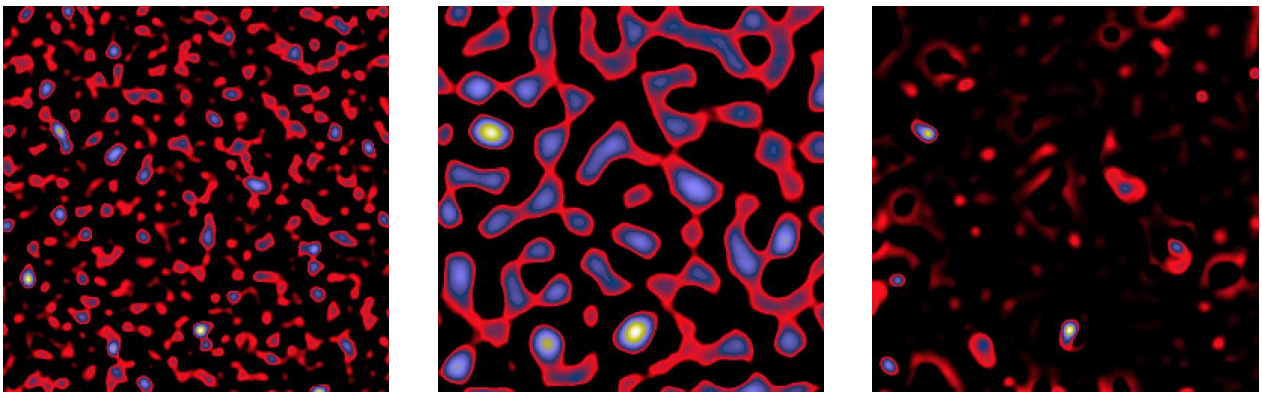

Figure 29: scale 3 - $n_g = 100$

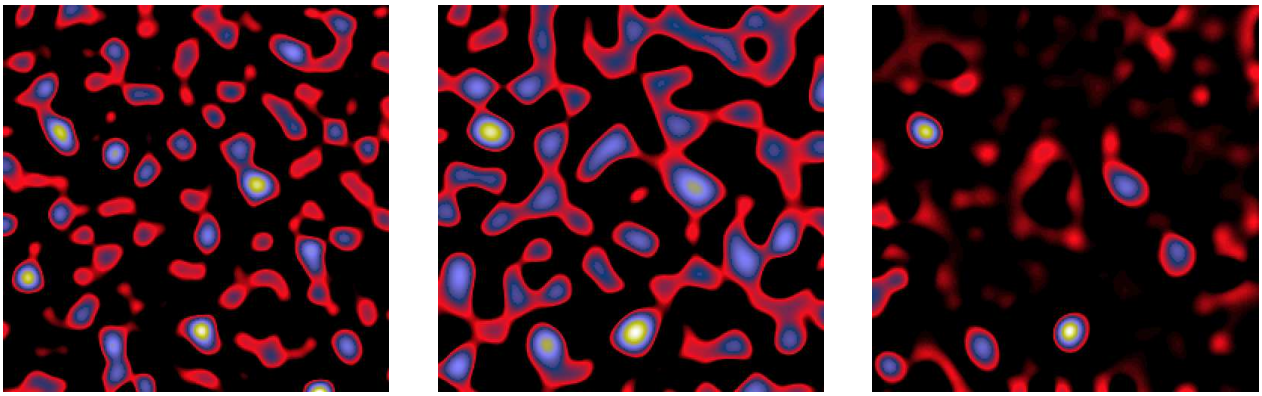

Figure 30: scale 4 - $n_g = 100$

3 Filterings of the noisy mass map with a lack of data

We have ended this survey by looking what would happen with the different filters, if some data were missing in the image.

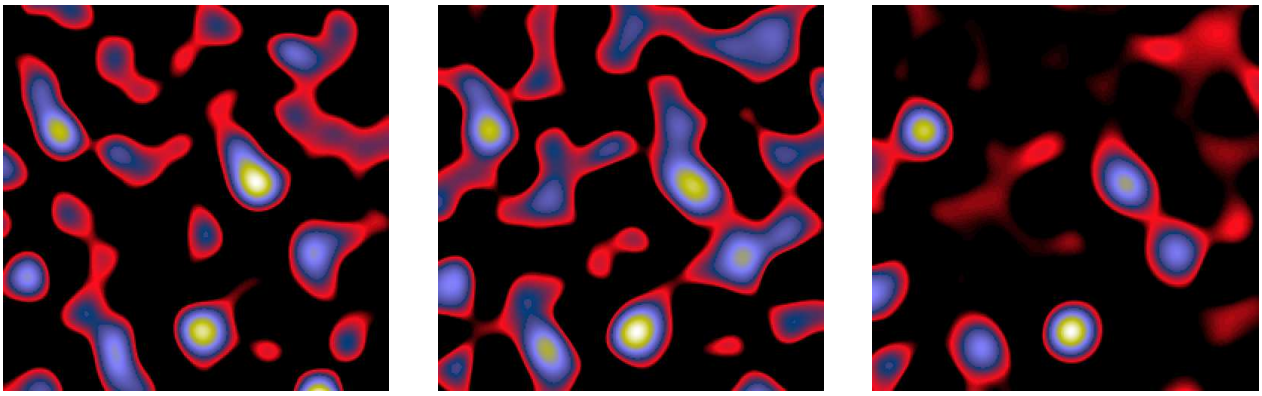

Figure 31: scale 5 - $n_g = 100$

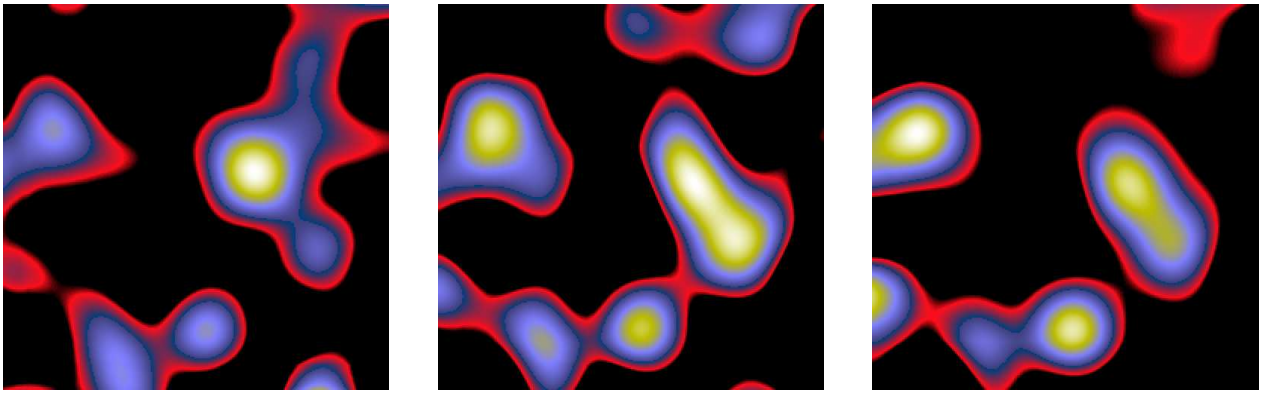

Figure 32: scale 6 - $n_g = 100$

3.1 Missing Data

Sometimes during the observations, an incident can cause a loss of data in the image. This can be due either to a defect of the camera CCD, 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 or something else,... In order to modelize this problem, we make a square zone take the value 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. 33). We were then able to test the different filterings seen earlier on this noisy mass map with a "hole".

3.2 Filterings in Fourier space

3.2.1 The Gaussian filter

In Fig. 34, we have represented the result of the Gaussian filtering on the noisy mass map with the "hole" (for observations on the ground $n_g = 20$ and for space observations $n_g = 100$). The result is quite good : we can notice that the square edges are smoothed.

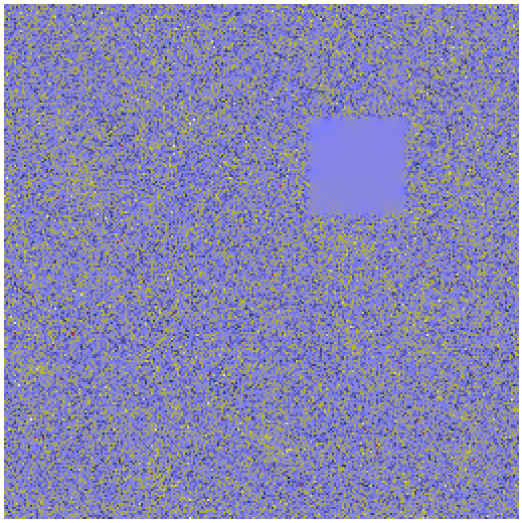

Figure 33: Noisy mass map with a "hole" ($n_g = 100$)

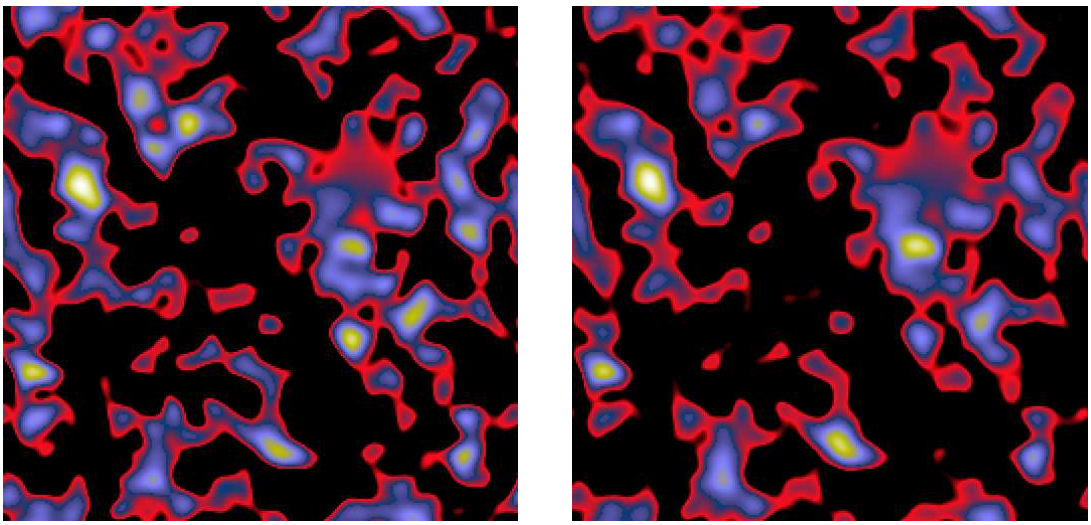

Figure 34: Noisy mass map with a "hole" filtered by a Gaussian $\sigma = 5$ ((left) $n_g = 20$ and (right) $n_g = 100$)

3.2.2 The Wiener filter

Fig. 35, shows the result of the Wiener filtering on the noisy mass map with the "hole" (for observations on the ground $n_g = 20$ and for space observations $n_g = 100$) in the Fourier space.

In this case, we notice that this processing gives a very bad result and that the "hole" appears in the filtered image. For $n_g = 20$, nothing is recovered. For $n_g = 100$, the image is slightly blurred. In order to understand this bad result, let us see how the Wiener weight is calculated. The pixel values of the "hole" are used to calculate the weight. Thus, the variance of the signal decreases on some of the rings compared to the value it would have without the "hole". The Wiener weight decreases in the same way and the structures are consequently affected.

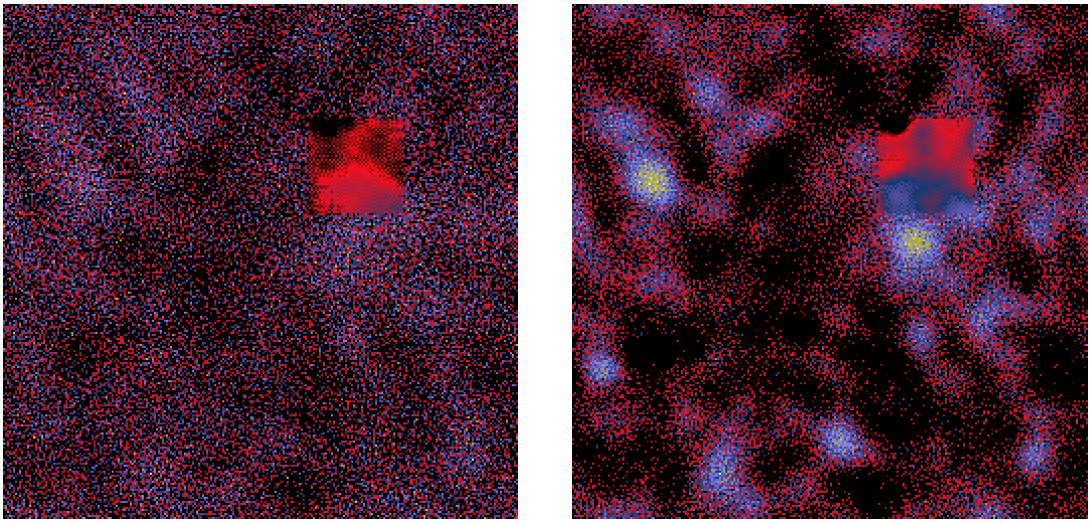

Figure 35: Noisy mass map with a "hole" filtered by the Wiener filter ((left) $n_g = 20$ and (right) $n_g = 100$)

3.3 Filterings in Wavelet space

As soon as we work in the wavelet space, the "hole" disappears except for the Wiener filtering in the wavelet space for the reason mentioned earlier. The "hole" disappears because it is only present in the small scales (high frequencies) and we have already seen that in the mutiresolution support, the first few scales are equal to 0 in order to suppress the high frequency noise. Thus, the "hole" is also removed.

3.3.1 The multiresolution support and wavelet coefficient thresholding

In Fig. 36, we have represented the result of the filtering by the wavelet coefficient thresholding on the noisy mass map with the "hole" (for observations on the ground $n_g = 20$ and for space observations $n_g = 100$). The presence of the "hole" in the data does not disturb the filtering of the other part of the image.

3.3.2 Wiener-like filtering in Wavelet space

The result of the Wiener-like filter in the wavelet space is as bad as the classical Wiener filter in the Fourier space.

3.3.3 Multiscale Entropy filtering

Fig. 37 shows the result of the Multiscale Entropy Filtering on the noisy mass map with the "hole" (for observations on the ground $n_g = 20$ and for space observations $n_g = 100$).

The image filtered by MEM is not much affected by the presence of the "hole". Indeed, when we minimize the information due to the noise in κ_f, we remove the high frequency noise and consequently the "hole" disappears because it is only present in the small scales. But we also minimize the information due to the signal of the difference between κ_f and κ_b and this can slightly disturb the result.

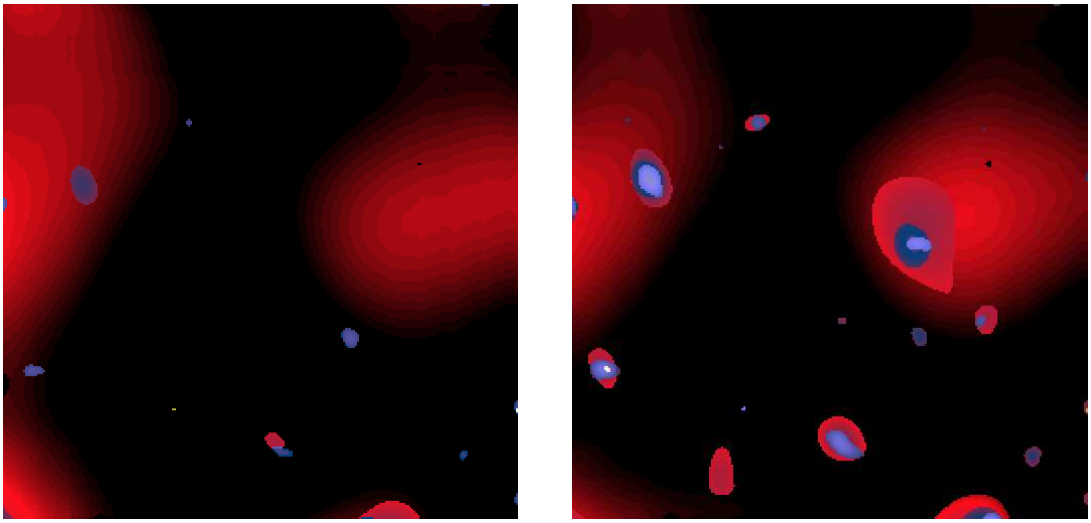

Figure 36: Noisy mass map with a "hole" filtered by the multiresolution support ((left) $n_g = 20$ and (right) $n_g = 100$)

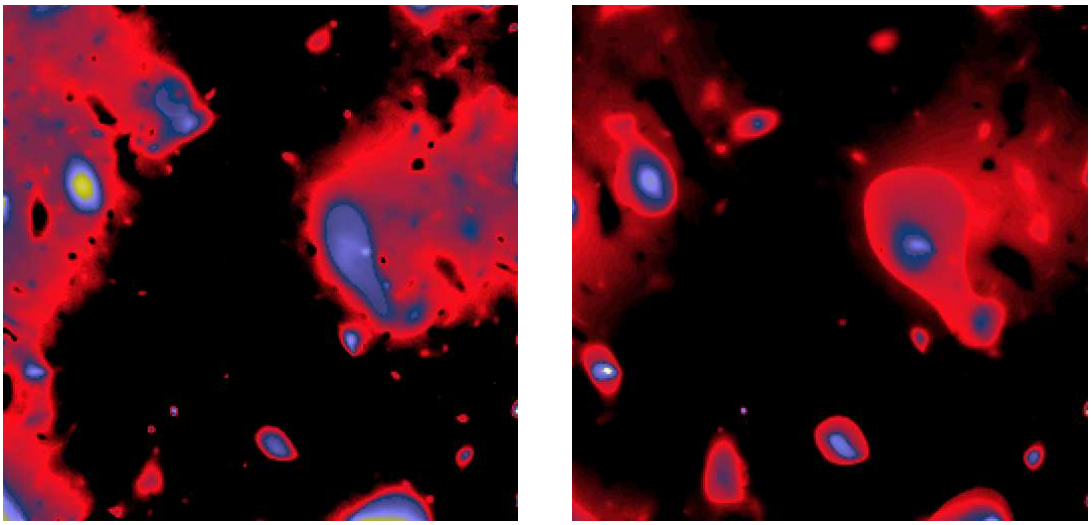

Figure 37: Noisy mass map with a "hole" filtered by the Multiscale Entropy Method without the first two scales ((left) $n_g = 20$ and (right) $n_g = 100$)

3.4 Conclusion about the filterings of the noisy mass map with a lack of data

The present study shows that it is a mistake to consider the pixels of the "hole" when we calculate the Wiener weight and to perform the Multiscale Entropy filtering. This is because the "hole" does not contain any information. We propose three solutions. The first one consists in calculating the weight or performing the MEM filtering without taking into account the pixels of the "hole". The second one, only for the Wiener filtering, is to create a noisy map with a "hole" to calculate the corresponding $< |\hat{N}(u,v)| >$. The third one, only for the MEM Method consists in removing the first two scales. The Fig. 38 shows the wavelet coefficients of the mass map with a "hole". we can notice that the hole is quite visible on the first two scales, those which we removed. It then starts to disappear and it is not visible anymore. Consequently, the

influence of the hole is rather weak when the first two scales are removed.

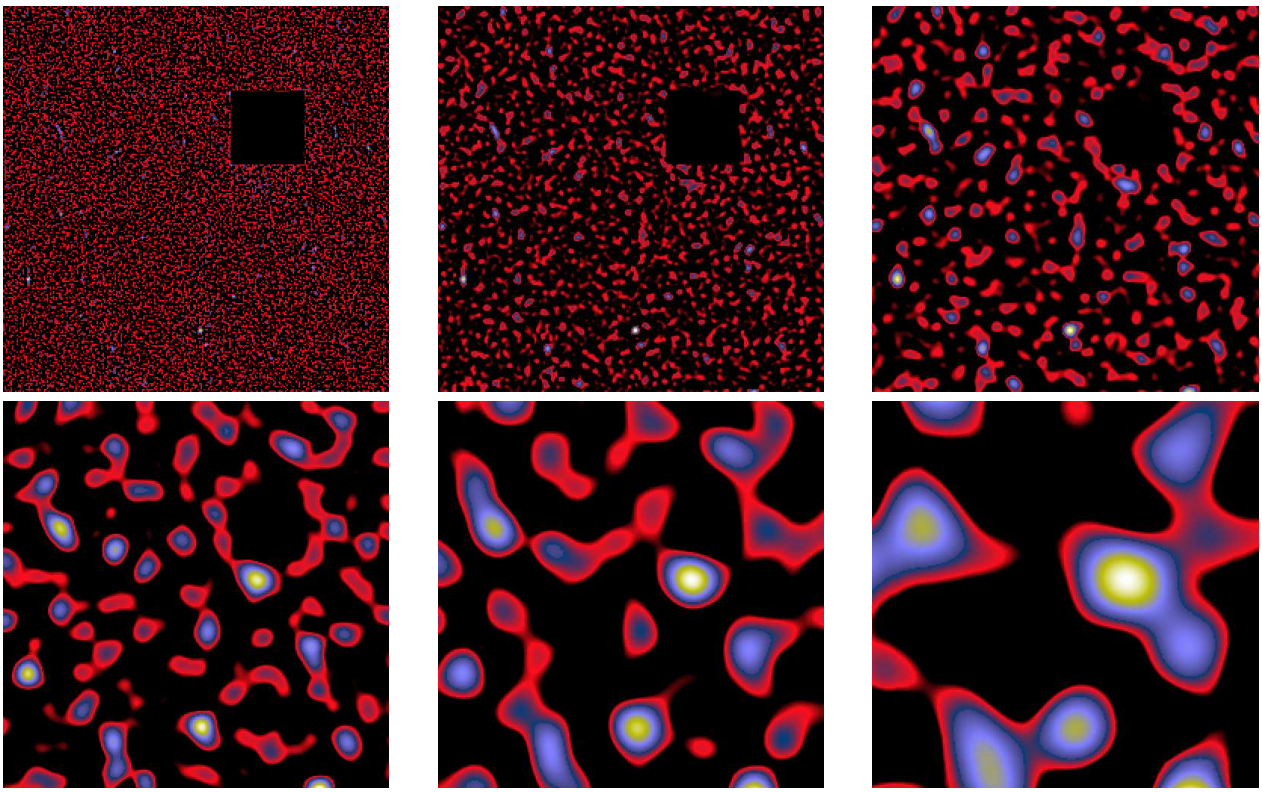

Figure 38: Wavelet coefficients for each scale of the wavelet transform of κ with a "hole"

4 Program

4.1 Organization in the folder *prog_WL*

In this folder, we have performed all filterings on the noisy mass map κ_b for the Weak Lensing.
This folder itself is divided into two folders. In the first, we gathered all filterings on simulated images and in the second one gathered all filterings on real images.
The SIMULATION folder itself is divided into three folders. In the first, are gathered all filterings performed on a mass map from bhuvnesh. The second one, gathered all filterings on a simulated mass map collected on Internet. And in the third one we have tried to simulate mass map from the caracteristics of some real images collected (noise level, pixel size, number of galaxies per $arcmin^2$,...) In all this sub-folders, we have a folder *data* in which we can find the simulated mass map, a folder *devl* in which we can find all the IDL routines used in this survey and a folder *images* in which we have saved the noisy mass map and all the filtered images (in the format .fits). There is a folder *save_ps*, in which we have saved a few images in the format .ps. and a folder *courbes_xdr*, in which we have saved a few interesting graphics in the format .xdr. A README file in each of these folders describes how each image and graphics have been generated.
The REAL folder itself is divided into two folders. In the first are all filterings on images from the observations on the ground and in the second one are all filterings on images from the space observations. In all this sub-folders, we have a folder *data* in which we can find the noisy mass map, a folder *devl* in which we can find all the IDL routines used in this survey and a folder *images* in which we have saved all the filtered images (in the format .fits). A README file in each of these folders describes how each image has been generated.

4.2 Organization in the folder *prog_hole_WL*

In this folder we have performed all the filterings on the noisy mass map with a lack of data for the Weak Lensing. All the file are quite similar to the files of the "prog_WL" folder. The main difference is in the procedure *kappa_hole.pro* that calculate the mask of the hole for all the scales which is used then by all the filtering procedures and in the procedure *rec_kap_hole.pro* that calculates the noisy mass map, it calls *bruit_gaus_hole* which puts a Gaussian noise on γ_1 and γ_2 (like *bruit_gaus* in the previous folder) and also adds a "hole" in the data.

4.3 IDL routines

4.3.1 IDL Process routine : *rec_kap* (or *rec_kap_hole* in the folder *prog_hole_WL*)

The program *rec_kap*
(or *rec_kap_hole* in the folder *prog_hole_header*) calculates the noisy mass map κ_b
from the simulate mass map.

REC_KAP, a, n_g, s, noi1, noi2, kb, mb

where

input :

a = the mass map (file structure)

n_g = the number of galaxies per $arcmin^2$

output :

s = the rms of the noise

noi1 = the array of the noise added to gamma1

noi2 = the array of the noise added to gamma2

kb = the noisy mass map

mb = the noisy mass map (file structure)

4.3.2 IDL Process routine : *rec_kap_gaus*

The program *rec_kap_gaus* calculates the convolution between the noisy mass map and a Gaussian window.

REC_KAP_GAUS, kb, sigma, k_gaus

where

input :

kb = the noisy mass map

sigma = the rms of the Gaussian window

output :

k_gaus = the filtered mass map

4.3.3 IDL Process routine : *rec_kap_wiener_1d*

The program *rec_kap_wiener_1d* calculates the solution given by the Wiener filtering (classical method 1D) by calculating the weight of the Wiener filter for each ring (the radius increasing logarithmically) of the image

REC_KAP_WIENER_1D, mb, sigmae, n_g, L, Vs, Vn, W, k_w1d

where

input :

mb = the noisy mass map (file structure)

sigmae = the rms of gamma1 and gamma2

n_g = the number of galaxies per $arcmin^2$

output :

L = the radius of the concentric circle

Vs = the standard deviation of the signal for each ring

Vn = the standard deviation of the noise for each ring

W = the weight of the wiener filter for each ring

k_w1d = the filtered mass map

4.3.4 IDL Process routine : *rec_kap_ond*

The program *rec_kap_ond* performs the filtering by the multiresolution method using wavelet transform

REC_KAP_OND, a, kb, n_g, ny, s, M, W, k_ond

where

input :

a = the mass map (file structure)

kb = the noisy mass map

n_g = the number of galaxies per $arcmin^2$

ny = the number of scales

s = the rms of the noise

output :

M = the mutiresolution support

W = the different scale plane

k_ond = the filtered mass map

4.3.5 IDL Process routine : *rec_kap_ond_iter*

The program *rec_kap_ond_iter* performs a multiresolution filtering using the wavelet transform and improved by the Van Cittert iteration

REC_KAP_OND_ITER, a, kb, n_g, ny, s, ni, M, W, k_ond1, k_ondi

where

input :

a = the mass map (file structure)

kb = the noisy mass map

n_g = the number of galaxies per $arcmin^2$

ny = the number of scales

s = the rms of the noise

ni = number of iteration

output :

M = the mutiresolution support

W = the different scale plane

k_ond1 = the filtered mass map for ni=1

k_ondi = the filtered mass map for ni=i

4.3.6 IDL Process routine : *rec_kap_ond_wl*

The program *rec_kap_ond_wl* performs a filtering using the wavelet transform and the Wiener filter

REC_KAP_OND_WL, a, kb, ny, s, noi1, noi2, Wk, Wn, Wkb, Wkrec, k_ond_wl

where

input :

a = the mass map (file structure)

kb = the noisy mass map

ny = the number of scales

s = the rms of the noise

noi1 = the array of the noise added to gamma1

noi2 = the array of the noise added to gamma2

output :

Wk = each scale of the wavelet transform for the mass map

Wn = each scale of the wavelet transform for the noise

Wkb = each scale of the wavelet transform for the noisy mass map

Wkrec = each scale of the wavelet transform for the map retraced

k_ond_wl = the array of the map retraced

4.3.7 IDL Process routine : *rec_kap_ond_wl_ht*

The program *rec_kap_ond_wl_ht* performs a filtering using the wavelet transform, the Wiener filter and a additional High Thresholding

REC_KAP_OND_WL_HT, a, kb, ny, s, noi1, noi2, k_ond_wl, k_ond_ht

where

input :

a = the mass map (file structure)

kb = the noisy mass map

ny = the number of scales

s = the rms of the noise

noi1 = the array of the noise added to gamma1

noi2 = the array of the noise added to gamma2

output :

k_ond_wl = the array of the map rebuilt without Hard Thresholding

k_ond_ht = the array of the map rebuilt

4.3.8 IDL Process routine : *rec_kap_ond_wl_st*

The program *rec_kap_ond_wl_st* performs a filtering using the wavelet transform, the Wiener filter and a additional Soft Thresholding

REC_KAP_OND_WL_ST, a, kb, ny, s, noi1, noi2, k_ond_wl, k_ond_st

where

input :

a = the mass map (file structure)

kb = the noisy mass map

ny = the number of scales

s = the rms of the noise

noi1 = the array of the noise added to gamma1

noi2 = the array of the noise added to gamma2

output :

k_ond_wl = the array of the map rebuilt without Soft Thresholding

k_ond_st = the array of the map rebuilt

4.3.9 IDL Process routine : *rec_kap_entropie*

The program *rec_kap_entropie* performs a filtering using the multiscale entropy method (for observations on the ground n_g = 20 or space observation n_g = 100)

REC_KAP_ENTROPIE, a, kb, n_g, k_entropie

where

input :

a = the mass map (file structure)

kb = the noisy mass map

n_g = the number of galaxies per $arcmin^2$

output :

k_entropie = the array of the map rebuilt by the multiscale entropy method

4.3.10 IDL Process sub-routine

my_kappa_to_gamma is called by *rec_kap* to calculate gamma1 and gamma2 from kappa

my_gamma_to_kappa is called by *rec_kap* to calculate kappa from gamma1 and gamma2

bruit_gaus (or *bruit_gaus_hole* in the folder *prog_hole_WL*) is called by *rec_kap* to put Gaussian noise on gamma1 and gamma2 (and in the folder *prog_hole_WL* in addition to the noise it simulates a "hole" in the data)

variance is called by *rec_kap_w1d* to calculate the standard deviation of each ring of the image

seuil_detec is called by *rec_kap_ond* to calculate the rms of noise for each scale

sim_bruit is called by *rec_kap_ond* to calculate the map of the simulated noise

mrs is called by *rec_kap_ond* to calculate the multiresolution support M of the noisy mass map

hard_t is called by *rec_kap_ond_wl_ht* to calculate the mutiresolution support by Hard Thresholding

soft_t is called by *rec_kap_ond_wl_st* to calculate the mutiresolution support by Soft Thresholding

4.3.11 IDL Process routines useful

pixelise enables us to make a pixelised map from another

mk_gamma enables us to make a pixelised map from shear catalogue

M_isophot enables us to calculate the value of maximum detection for each pixel of the map

histo enables us to plot the normalised histogram of a map

4.3.12 IDL Process routines of reading

rd_map enables us to read the mass map simulated from bhuvnesh

rd_map_net1 enables us to read the first simulated mass map collected on Internet

rd_map_net2 enables us to read the second simulated mass map collected on Internet

rd_gcat enables us to read the galaxies catalogue of the observations on the ground

rd_gcat_goods enables us to read the galaxies catalogue of the space observations

4.3.13 IDL Process routines of writing or saving

plt_colbar, *plt_evec*, *plt_image*, *plt_shear_kappa*, *plt_shear_gamma* , *plt_gamma*, *plt_gcat* are used to plot the images.

ops, *cps*, *opub*, *cpub* are used to save our images in the .ps format

mk_jpeg is used to save our images in the .jpeg format

References

[1] M. Bertero and P. Boccacci. *Introduction to Inverse Problems in Imaging.* Institute of Physics, 1998.

[2] F. Murtagh, J.-L. Starck, and A. Bijaoui. Image restoration with noise suppression using a multiresolution support. *Astronomy and Astrophysics, Supplement Series*, 112:179–189, 1995.

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

[4] J.-L. Starck, F. Murtagh, and A. Bijaoui. *Image Processing and Data Analysis: The Multiscale Approach.* Cambridge University Press, 1998.

[5] J.-L. Starck, M.K. Nguyen, and F. Murtagh. Wavelets and curvelets for image deconvolution: a combined approach. *Signal Processing*, 2002. in press.

[6] C.J. Miller, C. Genovese, R.C. Nichols, L. Wasserman, A. Connolly, D. Reichart, A. Moore. *Controlling The False-Discovery Rate in Astrophysical Data Analysis* The Astronomical Journal, 122:3492-3505, 2001 December

[7] C. Genovese, N.A. Lazar, T. Nichols *Thresholding of Statistical Maps in Functional Neuroimaging Using the False Discovery Rate*

[8] M.J. Fadili and E.T. Bullmore *A Comparative evaluation of Wavelet-based methods for multiple hypothesis testing of brain activation maps Submitted to NeuroImage*, 2004.