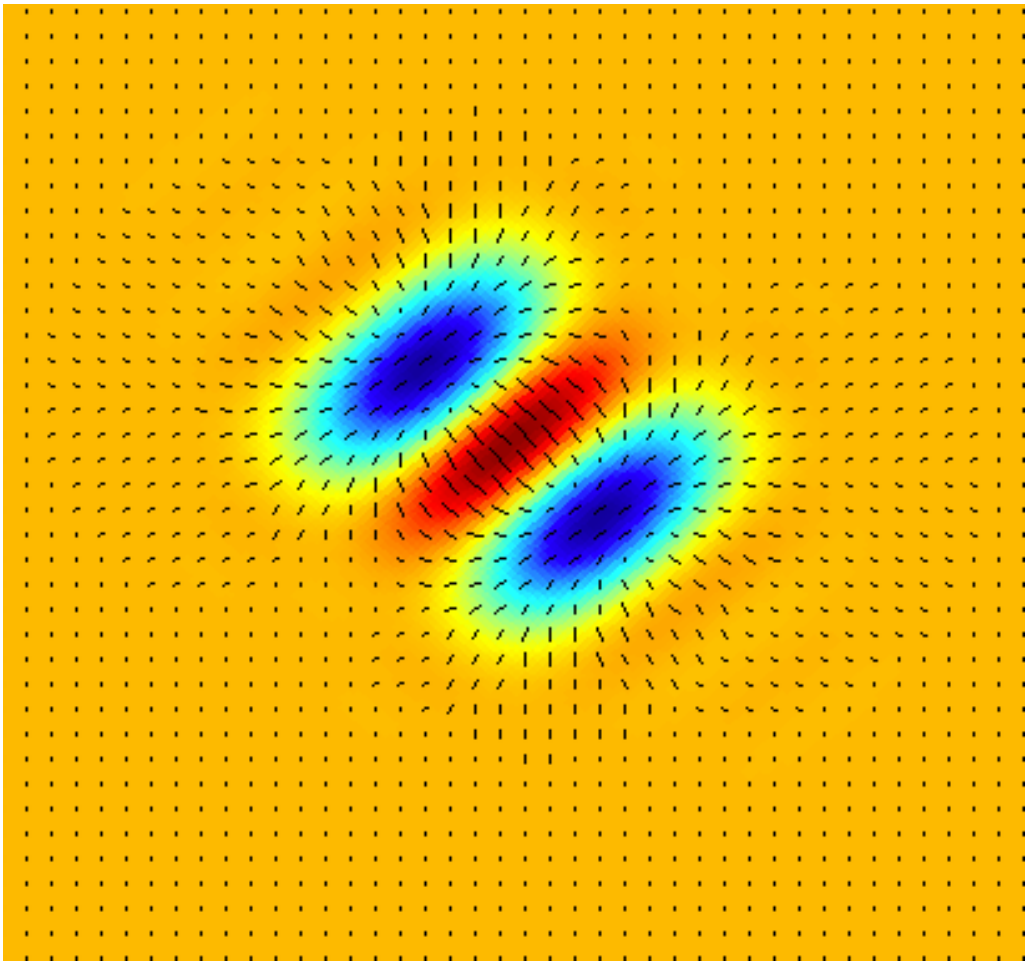
Interactive Sparse astronomical data Analysis Packages: iSAP V3.0

CEA-SACLAY/IRFU-AIM

CEA Technical Report

http://www.cosmostat.org



Contents

Acknowledgments

iSAP

iSAP is a collection of packages, in IDL and C++. The iSAP environment has been designed by F. Sureau, F. Lanusse and J.-L. Starck. The authors of the different packages are:

Sparse2D/V1.0 package

Main developers for this package are J.-L. Starck, A. Woiselle, J. Fadili, J. Bobin, and F. Sureau.

This package is based on algorithms and methods described in:

- **Sparse Image & Signal Processing: wavelets, curvelets, morphological diversity**, J.-L. Starck , F. Murtagh and J. Fadili, *Cambridge University Press*, 2010.

MSVST/V1.0 package

Main developers for this package are B. Zhang, J. Fadili and J.-L. Starck.

This package is based on methods described in:

- **Wavelets, Ridgelets and Curvelets for Poisson Noise Removal**, B. Zhang, M.J. Fadili and J.-L. Starck, *IEEE Transactions on Image Processing*, Vol 17, No 7, pp 1093–1108, 2008. .

MRS/V3.2 package

Main developers for this package are F. Sureau, J.-L. Starck, J. Bobin. Developers of previous versions are O. Fourt, P. Abrial and Y. Moudden. The authors of this software package would like to express their gratitude to all of their colleagues who have made this work possible through various contributions. We are especially thankful to our direct collaborators with whom we have been working for years on problems in statistics and data analysis in astrophysics and cosmology: Nabila Aghanim, Fabio Noviello, Jacques Delabrouille, David Donoho, Jalal Fadili, Olivier Forni, Jiashun Jin and Mai Nguyen. The original paper describing wavelets, ridgelets and curvelets on the sphere is:

- **Wavelets, Ridgelets and Curvelets on the Sphere**, J.-L. Starck, P. Abrial, Y. Moudden and M. Nguyen, *Astronomy and Astrophysics*, vol. 446, pp 1191-1204, 2, 2006.

A more detailed description can be found in:

- **Sparse Image & Signal Processing: wavelets, curvelets, morphological diversity**, J.-L. Starck , F. Murtagh and J. Fadili, *Cambridge University Press*, 2010.

Other colleagues we would like to acknowledge include: Bedros Afeyan, Emmanuel Candes, Jean-François Cardoso, Pierre-François Honoré, Guillaume Patanchon, Ludovic Poupard, Philippe Querre, Sandrine Pires, Ryad Sehil and Patricio Vielva.

This work was partially supported by the French National Agency for Research (ANR -08-EMER-009-01) and by the European Research Council (ERC-228261).

SparsePOL/V1.1 package

Main developers for this package are F. Sureau, J.-L. Starck, J. Bobin. Developers of previous versions are O. Fourt, P. Abrial and Y. Moudden. The authors of this software package would like to express their gratitude to David Donoho, Morteza Shahram, and Inam Ur-Raman

This package is based on algorithms and methods which were published in:

- **Polarized Wavelets and Curvelets on the Sphere**, J.-L. Starck, Y. Moudden and J. Bobin, *Astronomy and Astrophysics*, Vol 497, pp 931–943, 2009.

This work was partially supported by the European Research Council (ERC-228261).

MRS-MSVST/V1.1 package

Main developers for this package are J. Schmitt , J.-L. Starck and J. Fadili.

This package is based on algorithms and methods which were developed and/or used successfully in a number of applications reported in the following publications:

- **Poisson Denoising on the Sphere: Application to the Fermi Gamma Ray Space Telescope**, J. Schmitt, J.L. Starck, J.M. Casandjian, J. Fadili, I. Grenier, *Astronomy and Astrophysics*, 517, A26, 2010.
- **Multichannel Poisson Denoising and Deconvolution on the Sphere : Application to the Fermi Gamma Ray Space Telescope**, J. Schmitt, J.L. Starck, J.M. Casandjian, J. Fadili, I. Grenier,, *Astronomy and Astrophysics*, 546, id.A114, pp10, 2012.

This work was partially supported by the European Research Council (ERC-228261).

SparseGal/V1.0 package

Main developers for this package are F.X. Dupé, A. Rassat, A. Leonard, D. Machado and J.-L. Starck. This package is based on algorithms and methods which were published in:

- **Darth Fader: Using wavelets to obtain accurate redshifts of spectra at very low signal-to-noise**, D. Machado, A. Leonard, J-L. Starck, and F. Abdalla, *Astronomy and Astrophysics*, 560, id.A83, pp 20, 2013.
- **Measuring the Integrated Sachs-Wolfe Effect", Astronomy and Astrophysics**, F. -X Dupe, A. Rassat, A., J.-L. Starck, M.J. Fadili, *Astronomy and Astrophysics*, 534, A51+, 2011.

This work was partially supported by the European Research Council (ERC-228261).

SparseCMB/V1.0 package

Main developers for this package are P. Paykari, F. Lanusse, F. Sureeau, J. Bobin, A. Rassat, J. Fadili and J.-L. Starck. This package is based on algorithms and methods which were published in:

- **On Preferred Axes in WMAP Cosmic Microwave Background Data after Subtraction of the Integrated Sachs-Wolfe Effect**, A. Rassat and J-L. Starck, *Astronomy and Astrophysics* , 557, id.L1, pp 7, 2013.
- **Removal of two large scale Cosmic Microwave Background anomalies after subtraction of the Integrated Sachs Wolfe effect**, A. Rassat, J-L. Starck, and F.X. Dupe, *Astronomy and Astrophysics* , 557, id.A32, pp 15, 2013.
- **WMAP 9-year CMB estimation using sparsity**, *Astronomy and Astrophysics* , J. Bobin, F. Sureau, P. Paykari, A. Rassat, S. Basak and J.-L. Starck, 553, L4, pp 10, 2013.
- **Sparsity and the Bayesian Perspective**, J.-L. Starck, D.L. Donoho, M.J. Fadili and A. Rassat, *Astronomy and Astrophysics* , 552, A133, 2013.
- **Sparse component separation for accurate CMB map estimation** , J. Bobin, J.-L. Starck, F. Sureau and S. Basak, *Astronomy and Astrophysics* , 550, A73, 2013.
- **Low-l CMB Analysis and Inpainting**, *Astronomy and Astrophysics* , J.-L. Starck, A. Rassat, and M.J. Fadili, 550, A15, 2013.
- **True CMB Power Spectrum Estimation**, P. Paykari, J.L. Starck, and J. Fadili, *Astronomy and Astrophysics* , 541, A74, 2012.
- **CMB map restoration**, J. Bobin, J.-L. Starck, F. Sureau, and J. Fadili, , *Advances in Astronomy* , 2012, Id703217, 2012.

This work was partially supported by the European Research Council (ERC-228261).

Chapter 1

iSAP Introduction and Installation

1.1 Introduction to iSAP

iSAP is a collection of packages, in IDL and C++, related to sparsity and its application in astronomical data analysis (the IDL software (`http://www.idl-envi.com`) is analogous to Matlab and is very widely used in astrophysics and in medical imaging). The C++ routines can be used independently of IDL. The library is available via the web site:

`http://www.cosmostat.org/isap.html`

It contains the following packages:

- **Sparse2D V1.0: Sparsity for 1D and 2D data set.**
  IDL and C++ code, allowing sparse decomposition, denoising and deconvolution.
- **MSVST V1.0: Multi-Scale Variance Stabilizing Transform (MSVST) for 1D and 2D data set.**
  IDL and C++ code for Poisson noise removal.
- **MRS V3.2: MultiResolution on the Sphere.**
  IDL and C++ code for sparse representation on the sphere.
- **SparsePOL V1.1: Polarized Spherical Wavelets and Curvelets.**
  IDL code for sparse representation of polarized data on the sphere.
- **MRS-MSVSTS V1.1: Multi-Scale Variance Stabilizing Transform on the Sphere.**
  IDL code for Poisson noise removal and deconvolution on mono-channel and multi-channel spherical data.
- **SparseGal V1.0: Sparsity for galaxies survey analysis.**
  IDL code, with two subpackages:
  - ISW V1.0: Integrated Sachs-Wolfe effect detection.
  - DarthFader V1.0: Spectroscopic Redshift Estimation using sparsity.

- **SparseCMB V1.0: Sparsity for CMB data analysis.**
  IDL and C++ code.

1.2 IDL Installation

A set of routines has been developed in IDL. Starting IDL using the script program *isap.pro* allows the user to get the sparse IDL astronomical data analysis environment, and all routines described in the following can be called. An online help facility is also available by invoking the *isaph* program under IDL.

Then, installing the iSAP package simply requires adding some lines in your environment profile:

- define the environment variable **ISAP**

  ```
  setenv ISAP /home/user/ISAP
  ```

- define the alias **isap**

  ```
  alias isap 'idl $ISAP/idl/isap.pro  $ISAP/idl/compile_healpixfile'
  ```

  If the Healpix package is not installed, then replace the previous command with

  ```
  alias isap 'idl $ISAP/idl/isap.pro'
  ```

  In this case, packages MRS, MRSP, MRS-MSVST will not be active.

Then the command "isap" will start the IDL session using the iSAP environment.

1.3 MRS/V3.2 package

The MRS package, included in iSAP, requires IDL (version 6.0 or later) and HEALPix (version 2.0 or later) to be installed. The HEALPix environment variable **HEALPix** is expected to be defined. HEALPix is available at:

```
http://sourceforge.net/projects/healpix
```

HEALPix binaries must be in the user path.

MRS/V3.2 contains also C++ programs that can be used from IDL or directly from a terminal session. It has been tested using gcc-4.4.1. C++ is not necessary, but for some applications such as inpainting, C++ routines are much faster. To compile C++ routine, do the following commands:

- go the **iSAP** directory

  ```
  cd  $ISAP/cxx/mrs/build
  ```

- build the makefile

```
./cmake ..
```

- run the makefile

```
make
```

- copy the created binaries to the **iSAP** binary directory

```
make install
```

- set the IDL global variable ISAPCXX equal to 1 in the file **$iSAP**/isap.pro or on the user command line after being in the IDL-iSAP environment.

```
ISAP>  ISAPCXX=1
```

Simular installation procedures must be done for each C++ package available in **$iSAP/cxx**.

1.3.1 Input Data

Most of the functions of MRS package are working with spherical maps either as input or as output variables. The maps could be in two possible kind of formats: the first one which will be recognized by all functions is the HEALPix format. This format allows two kind of pixel order in the data array called NESTED and RING. Unless stated, MRS functions will work only with NESTED maps. Conversions between NESTED and RING schemes could be done with the function "REORDER" from HEALPix package.

The second format which is recognized by some functions of MRS package, but not all, is the GLESP format. In IDL, the variable for an image in GLESP format is an IDL structure. The MRS package includes the file "mrs_glesp.pro" which contains several functions for working with GLESP images especially the functions "healpix2glesp" and "glesp2healpix" which are used for the conversions between Healpix and GLESP format. To use the GLESP image format, GLESP library, available at `http://www.glesp.nbi.dk`, must be installed, and the environment variable **GLESP** must be initialized to the correct path.

1.3.2 Global IDL MRS Variable

Four global MRS variables, defined in the file "isap.pro", are available, and can be changed by the user.

- HealpixCXX: default is 1. By default, MRS uses HEALPix C++ programs to compute the spherical harmonic coefficients. If HealpixCXX is set to 0, MRS will call the HEALPix Fortran programs. We recommend to keep HealpixCXX equal to 1. Fortran option will not be supported in the future.

- DEF_ALM_FAST: default is 1. By default, spherical harmonic coefficients are calculated with floating values. This is faster and requires less memory, but is not as accurate than using double. Set DEF_ALM_FAST to zero to make all calculation with double.

- DEF_ALM_NITER: default is 10. This variable is only used when DEF_ALM_FAST is equal to zero. DEF_ALM_NITER is the number of iterations used by HEALPix to compute the spherical harmonic coefficients. In the fast mode (default mode), there is no iteration.
- DEF_NORM_POWSPEC: default is 0. If DEF_NORM_POWSPEC is set to 1, the command "mrs_powspec" will return a normalized power spectrum, such that a map containing a Gaussian noise with variance 1 will have a power spectrum equal to 1.
- ISAPCXX: Default is 0: If the **iSAP** C++ code has been compiled, it is recommended to set it to 1. When it is equal to 1, then spherical harmonic transform and recontruction are done using the **MRScxx** binaries instead of Healpix binaries.

1.4 SparsePOL/V1.1 package

The SparsePOL package, included in iSAP, requires IDL (version 6.0 or later), HEALPix (version 2.0 or later), and MRS/V3.2 to be installed.

Part I

MRS/V3.2 : MultiResolution on the Sphere

Chapter 2

Data on the Sphere

2.1 Introduction

In a number of areas of scientific activity, data is gathered which naturally maps to the sphere. For instance, remote sensing of the Earth's surface and atmosphere, e.g. with POLDER[1], generates spherical data maps which are crucial for global and local geophysical studies such as understanding climate change, geodynamics or monitoring human-environment interactions. More examples can be found in medical imaging or computer graphics. In astronomy and astrophysics, recent and upcoming ground based and satellite borne experiments such as WMAP[2] or Planck-Surveyor[3] for the observation of the Cosmic Microwave Background radiation field over the whole celestial sphere, produce full-sky maps in a wide range of wavelengths. These maps are necessarily digitized and hence distributed as a finite set of pixel values on some grid. The properties of this grid will affect the subsequent analysis of the data, and a good choice will make standard computations, such as the spherical harmonics transform, much faster and accurate. Considerable work has been dedicated to the development of pixelization schemes on the sphere. In particular, Healpix (Górski et al. 2005) is a sampling scheme which has some attractive geometrical features profitably used in this spherical data analysis software package.

Processing spherical data maps requires specific tools or somehow adapting traditional methods used on flat images to the spherical topology, such as multiscale transforms for image processing. Among these, Wavelets and related representations are by now successfully used in all areas of signal and image processing. Their recent inclusion in JPEG 2000 – the new still-picture compression standard – is an illustration of this lasting and significant impact. Wavelets are also very popular tools in astronomy (Starck and Murtagh 2002) which have led to very impressive results in denoising and detection applications. For instance, both the Chandra and the XMM data centers use wavelets for the detection of extended sources in X-ray images. For denoising and deconvolution, wavelets have also demonstrated how powerful they are for discriminating signal from noise (Starck et al. 2002b). In cosmology, wavelets have been used in many studies such as for analyzing the spatial distribution of galaxies (Slezak et al. 1993; Escalera and MacGillivray 1995;

[1] http://polder.cnes.fr
[2] http://map.gsfc.nasa.gov
[3] http://astro.estec.esa.nl/Planck

Starck et al. 2005; Martínez et al. 2005), determining the topology of the universe (Rocha et al. 2004), detecting non-Gaussianity in the CMB maps (Aghanim and Forni 1999; Barreiro et al. 2001; Vielva et al. 2004; Starck et al. 2004a), reconstructing the primordial power spectrum (Mukherjee and Wang 2003), measuring the galaxy power spectrum (Fang and Feng 2000) or reconstructing weak lensing mass maps (Starck et al. 2006). It has also been shown that noise is a problem of major concern for N-body simulations of structure formation in the early Universe and that using wavelets for removing noise from N-body simulations is equivalent to simulations with two orders of magnitude more particles (Romeo et al. 2003, 2004).

This technical report includes an overview of multiscale transforms on the sphere and introduces their uses in several algorithms for signal processing on the sphere. In this chapter, the Section 2.2 overviews the problem of pixelization on the sphere and introduces the two solutions used by MRS package: the HEALPix pixelization scheme 2.2.1 and Gauss-Legendre Sky Pixelization (GLESP) 2.2.2. The Section 2.3 introduces the spherical harmonics transform which could be considered as an extension of Fourier's transform to the sphere. Section 2.4 is a short introduction to multiscale methods on the sphere and to their applications. These methods are fully presented in Chapter 3 and algorithm on the sphere bases on them are given in Chapter 4 for data restoration, in Chapter 5 for sparse component analysis and in Chapter 6 for blind source separation. The Chapter 7 is dedicated to statistics on the sphere which includes the detection of non-gaussianities. Sections 5.2.3 and 5.3.2 present how these new tools can help us to analyze data in two real applications, in physics and in cosmology. Finally, a guided numerical experiments together with a toolbox written in IDL and dedicated to multiscale transforms on the sphere (MR/S) are described in Chapter 8.

2.2 Pixelization

Various pixelization schemes for data on the sphere exist in the literature. These include the Equidistant Coordinate Partition (ECP), the Icosahedron method (Tegmark 1996), the Quad Cube (White and Stemwedel 1992), IGLOO (Crittenden 2000), HEALPix (Górski et al. 2005), Hierarchical Triangular Mesh (HTM) (Kunszt et al. 2001) or Gauss-Legendre Sky Pixelization (GLESP) (Doroshkevich et al. 2005). Important properties to decide which one is the best for a given application include the number of pixels and their size, fast computation of the spherical harmonics transform, equal surface area for all pixels, pixel shape regularity, separability of variables with respect to latitude and longitude, availability of efficient software libraries including parallel implementation, etc. Each of these properties has advantages and drawbacks. In this section, we present the HEALPix and the GLESP representation which have several useful properties.

2.2.1 HEALPix

The HEALPix representation (Hierarchical Equal Area isoLatitude Pixelization of a sphere) (Górski et al. 2005)[4] is a curvilinear hierarchical partition of the sphere into quadrilateral pixels of exactly equal area but with varying shape. The base resolution divides the sphere into 12 quadrilateral faces of equal area placed on three rings around

[4] http://healpix.jpl.nasa.gov.

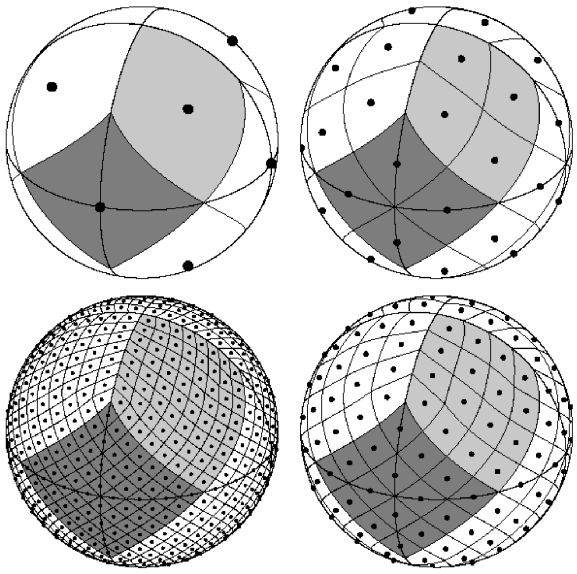


Figure 2.1: The HEALPix sampling grid for four different resolutions.

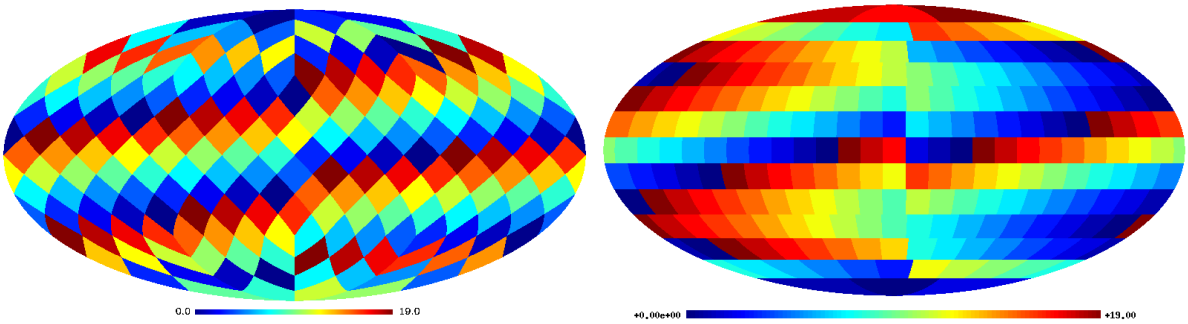


Figure 2.2: Comparison of pixelization schemes for a low resolution map (880 arcmin) with Healpix pixelization (on the left) and Glesp pixelization (on the right).

the poles and equator. Each face is subsequently divided into N_{side}^2 pixels following a quadrilateral multiscale tree structure (see Fig. 2.1). The pixel centers are located on iso-latitude rings, and pixels from the same ring are equispaced in azimuth. This is critical for computational speed of all operations involving the evaluation of the spherical harmonics coefficients, including standard operations such as convolution, power spectrum estimation, and so on. HEALPix is a standard pixelization scheme in astronomy.

2.2.2 Gauss-Legendre Sky Pixelization (GLESP)

The principle of GLESP (Doroshkevich et al. 2005) is to select pixels centers on points whose latitude is on the zeros of the associated Legendre's polynomial. In order to fix the longitude, two solutions have been built. The first one seeks for pixel with surfaces approximately equal. The equator is split in $2 * l + 1$ pixels and other pixels bands are split in order to have pixel surfaces as closed as possible to those of the equatorial band.

The second one chooses pixels with an equally spaced longitude. Pixels don't have the same surface, but the accuracy on spherical harmonics calculation is better.

Figure 2.2 shows the shape and localization of pixels with both Healpix and GLESP scheme for a low resolution map. The difference in the shape of the pixels is clearly shown. There are 16 rings and 192 pixels for the Healpix scheme, and 13 rings and 220 pixels for the GLESP one.

2.3 Spherical Harmonics

In the following, the $\hat{}$ notation will be used to denote the spherical harmonics coefficients of a function. Any function $f(\theta, \vartheta) \in L_2(S^2)$ on the sphere S^2 in $\mathbb{R}^3$ can be decomposed into spherical harmonics:

$$f(\theta, \vartheta) = \sum_{l=0}^{+\infty} \sum_{m=-l}^{l} \hat{f}_{lm} Y_{lm}(\theta, \vartheta), \tag{2.1}$$

where Y_{lm} are the spherical harmonics defined by:

$$Y_{lm}(\theta, \vartheta) = \sqrt{\frac{2l+1}{4\pi} \frac{(l-|m|)!}{(l+|m|)!}} P_{lm}(\cos\vartheta) e^{im\theta}, \tag{2.2}$$

P_{lm} are the associated Legendre functions (or polynomials) defined by the following differential equation:

$$\frac{d}{dt}\left[(1-t^2)\frac{d}{dt}P_{lm}\right] + \left(l(l+1) - \frac{m^2}{1-t^2}\right)P_{lm} = 0. \tag{2.3}$$

These functions are related to the Legendre polynomials P_l by

$$P_{lm}(t) = (-1)^m (1-t^2)^{m/2} \frac{d^m}{dt^m} P_l(t), \tag{2.4}$$

where P_l is:

$$P_l(t) = \frac{1}{2^l l!} \frac{d^l}{dt^l} (t^2-1)^l. \tag{2.5}$$

Furthermore, an important property of the Legendre polynomials is that they are orthogonal:

$$\sum_{l \in \mathbb{N}} \sum_{|m| \leqslant l} Y^*_{lm}(\boldsymbol{\omega}')\ Y_{lm}(\boldsymbol{\omega}) = \delta(\boldsymbol{\omega}' - \boldsymbol{\omega}). \tag{2.6}$$

with $\boldsymbol{\omega} = (\theta, \vartheta)$ et $\boldsymbol{\omega}' = (\theta', \vartheta')$

2.4 Multiscale methods on the sphere

2.4.1 Wavelets on the sphere

Many wavelet transforms on the sphere have been proposed in the past years. Using the lifting scheme (Schröder and Sweldens 1995) developed an orthogonal Haar wavelet transform on any surface, which can be directly applied on the sphere. Its interest is however relatively limited because of the poor properties of the Haar function and the problems inherent to orthogonal transforms. More interestingly, many papers have presented new continuous wavelet transforms (Antoine 1999; Tenorio et al. 1999; Cayón et al. 2001a; Holschneider 1996). These works have been extended to directional wavelet transforms (Antoine et al. 2002; McEwen et al. 2005). All these continuous wavelet decompositions are useful for data analysis, but cannot be used for restoration purposes because of the lack of an inverse transform. (Freeden and Windheuser 1997) and (Freeden and Schneider 1998) proposed the first redundant wavelet transform, based on the spherical harmonics transform, which presents an inverse transform. (Starck et al. 2006) proposed an invertible isotropic undecimated wavelet transform (UWT) on the sphere, also based on spherical harmonics, which has the same property as the isotropic undecimated wavelet transform, i.e. the sum of the wavelet scales reproduces the original image. A similar wavelet construction (Marinucci et al. 2008; Faÿ and Guilloux 2008; Faÿ et al. 2008) used the so-called needlet filters. (Wiaux et al. 2008) also proposed an algorithm which permits to reconstruct an image from its steerable wavelet transform. Since reconstruction algorithms are available, these new tools can be used for many applications such as denoising, deconvolution, component separation (Moudden et al. 2005; Bobin et al. 2008; Delabrouille et al. 2008) or inpainting (Abrial et al. 2007; Abrial et al. 2008).

The MRS package offers an implementation of an isotropic wavelet transform on the sphere. Its properties are similar to those of the "à trous" algorithm and therefore should be very useful for data denoising and deconvolution. This algorithm, described in chapter 3, is directly derived from the FFT-based wavelet transform proposed in (Starck et al. 1994) for aperture synthesis image restoration. It is relatively close to the Freeden and Maier (Freeden and Schneider 1998) method, except that the reconstruction process is as straightforward as in the "à trous" algorithm (i.e. the sum of the scales reproduces the original data). This wavelet transform can also be easily extended to a pyramidal wavelet transform, which may be very important for larger data sets such such as from the Planck experiment.

2.4.2 Ridgelets and Curvelets on the sphere

In this area, further insight will come from the analysis of full-sky data mapped to the sphere thus requiring the development of a curvelet transform on the sphere. The MRS package offers an implementation of ridgelet and curvelet transforms for spherical maps. Those implementations are derived as extensions of the digital ridgelet and curvelet transforms described in (Starck et al. 2002a). The implemented undecimated isotropic wavelet transform on the sphere and the specific geometry of the Healpix sampling grid are important components of the present implementation of curvelets on the sphere.

Further motivation for developing these multiscale methods on the sphere follows from

the results obtained in different data processing applications. As described in chapter 4, the MRS package provides the necessary tools to experiment with these spherical multiscale transforms in denoising applications, for instance using the Combined Filtering Method, which allows us to filter data on the sphere using both the Wavelet and the Curvelet transforms. The analysis of multichannel data mapped to the sphere, a problem encountered for instance in the processing of WMAP and Planck observations, is another issue that is shown to benefit from the developed multiscale representations on the sphere. This is reported in chapter 6 which is dedicated to describing some methods in multichannel data analysis extended to spherical maps which are implemented in the MRS package.

Chapter 3

Multiscale Methods on the Sphere

In this chapter, many multiscale decompositions will be built based on the spherical harmonics and/or the HEALPix representation.

3.1 Orthogonal Haar Wavelets on the Sphere

The Haar wavelet transform on the sphere (Schröder and Sweldens 1995) at each resolution j and pixel $\mathbf{k} = (k_x, k_y)$ on the sphere is based on a scaling function $\phi_{j,\mathbf{k}}$ ($\phi_{j,\mathbf{k}}(\mathbf{x}) = \phi\left(2^{-j}(\mathbf{x} - \mathbf{k})\right)$, where $\mathbf{x}$ is the vector of Cartesian coordinates on the sphere, and ϕ is the Haar scaling function) and three Haar wavelet functions $\psi^d_{j,\mathbf{k}}$ (see (3.1)) with $d \in \{1, 2, 3\}$. It uses the idea that a given pixel on the sphere at a given resolution j in the HEALPix representation is directly related to four pixels at the next resolution $j - 1$.

Noting, $\mathbf{k}_0$, the pixel. The scaling function ϕ and three wavelet functions ψ are defined by:

$$\begin{aligned}
\phi_{j,\mathbf{k}}(\mathbf{x}) &= \begin{cases} 1 & \text{if } t \in S_{j,\mathbf{k}} \\ 0 & \text{otherwise} \end{cases} \\
\psi_{1,j+1,\mathbf{k}} &= \frac{\phi_{j,\mathbf{k}_0} + \phi_{j,\mathbf{k}_2} - \phi_{j,\mathbf{k}_1} - \phi_{j,\mathbf{k}_3}}{4} \\
\psi_{2,j+1,\mathbf{k}} &= \frac{\phi_{j,\mathbf{k}_0} + \phi_{j,\mathbf{k}_1} - \phi_{j,\mathbf{k}_2} - \phi_{j,\mathbf{k}_3}}{4} \\
\psi_{3,j+1,\mathbf{k}} &= \frac{\phi_{j,\mathbf{k}_0} + \phi_{j,\mathbf{k}_3} - \phi_{j,\mathbf{k}_1} - \phi_{j,\mathbf{k}_2}}{4}
\end{aligned} \tag{3.1}$$

Denoting $\mathbf{k}_0, \mathbf{k}_1, \mathbf{k}_2, \mathbf{k}_3$ the four pixels at scale j, hierarchically related to the pixel $\mathbf{k}$ at scale $j + 1$, scaling coefficients $c_{j+1,\mathbf{k}}$ at scale $j + 1$ are derived from those at scale j by:

$$c_{j+1}[\mathbf{k}] = \frac{1}{4} \sum_{d=0}^{3} c_j[\mathbf{k}_d], \tag{3.2}$$

and wavelet coefficients at scale $j+1$ from coefficients at scale j by:

$$\begin{aligned} w^1_{j+1}[\mathbf{k}] &= \frac{1}{4}(c_j[\mathbf{k}_0] + c_j[\mathbf{k}_2] - c_j[\mathbf{k}_1] - c_j[\mathbf{k}_3]) \\ w^2_{j+1}[\mathbf{k}] &= \frac{1}{4}(c_j[\mathbf{k}_0] + c_j[\mathbf{k}_1] - c_j[\mathbf{k}_2] - c_j[\mathbf{k}_3]) \\ w^3_{j+1}[\mathbf{k}] &= \frac{1}{4}(c_j[\mathbf{k}_0] + c_j[\mathbf{k}_3] - c_j[\mathbf{k}_1] - c_j[\mathbf{k}_2]). \end{aligned} \tag{3.3}$$

The Haar wavelet transform on the sphere is orthogonal and its reconstruction is exact. The inverse transformation is obtained by:

$$c_0[\mathbf{x}] = \sum_{\mathbf{k}} c_J[\mathbf{k}]\phi_{J,\mathbf{k}}(\mathbf{x}) + \sum_{j=1}^{J}\sum_{d=1}^{3}\sum_{\mathbf{k}} w^d_j[\mathbf{k}]\psi^d_j(\mathbf{x}). \tag{3.4}$$

This transform is very fast but its interest is relatively limited. Indeed, it is not rotation invariant, and more importantly the Haar wavelet shape is not well adapted for most applications, because of the non-regular shape of the wavelet function.

3.2 Continuous Wavelets on the Sphere

3.2.1 Stereoscopic Projection

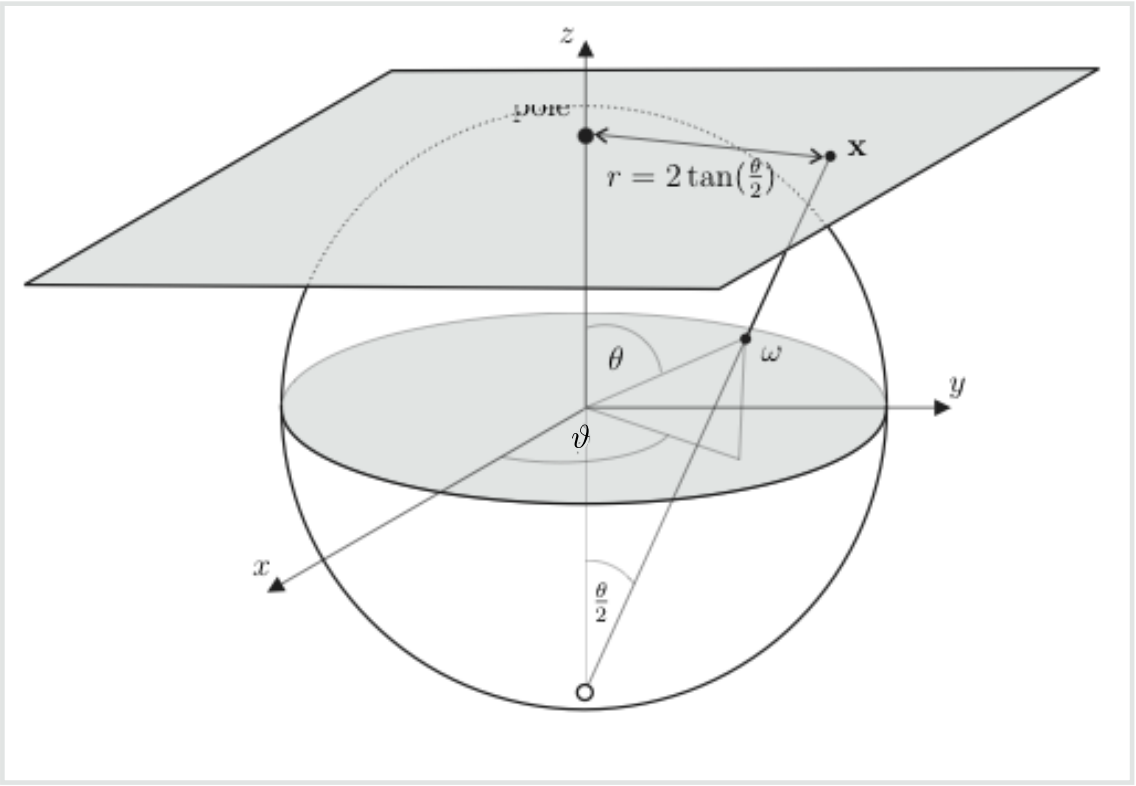


Figure 3.1: Inverse stereographic projections of a radial function from plane to the sphere.

In order to have more choice to design the wavelet function, we may want to use wavelets defined for regular 2D images to the sphere. This is possible by using inverse stereographic projections of radial wavelet functions such the Mexican hat (Cayón et al. 2001a). Defining the stereographic projection operator, $\mathbf{R} : \mathbf{t} \mapsto \boldsymbol{\omega}$, with $\boldsymbol{\omega} = (\theta(r), \vartheta)$, $\theta(r) = 2\arctan(r/2)$, the radial wavelets ψ_{plane} can be projected on the sphere by a unique rotation, $\boldsymbol{\omega}_0 = (\theta_0, \vartheta_0)$, respectively around the two axes O_y and O_z. Fig. 3.1 shows the projection of radial functions from the plane to the sphere.

The convolution on the sphere between a radial wavelet function $\psi(\theta)$ and a function $f(\boldsymbol{\omega})$ is:

$$(\psi * f)(\theta, \vartheta) = \int_{S^2} \psi^*_{\text{plane}}(\mathbf{R}^{-1}\boldsymbol{\omega}) f(\boldsymbol{\omega}) d\boldsymbol{\omega}. \tag{3.5}$$

Such wavelets are axisymmetric by construction. This property can be used to derive fast transformation algorithms using spherical harmonics. Indeed spherical harmonics coefficients $\hat{\psi}[l, m]$ of the wavelet function ψ on the sphere are equal to zero when $m \neq 0$, and by the Funk-Hecke theorem, the convolution can be written using spherical harmonics by:

$$(\psi * f)(\theta, \vartheta) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \sqrt{\frac{2l+1}{4\pi}} \hat{f}[l, m] \hat{\psi}[l, 0] Y_{lm}(\theta, \vartheta) \ . \tag{3.6}$$

where $\hat{f}[l, m]$ are the spherical harmonics coefficients of the function f,
i.e. $f = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \hat{f}[l, m] Y_{lm}$ and similary for $\hat{\psi}$.

Classical wavelet dilations can also be derived on the sphere using the dilation operator $\mathscr{D}_a$ by a factor $a > 0$ (Wiaux et al. 2007):

$$\mathscr{D}_a(f)(\boldsymbol{\omega}) = \chi_a^{1/2}(a, \theta) f(D_a^{-1}\boldsymbol{\omega}), \tag{3.7}$$

where $D_a(\theta, \vartheta) = (\theta_a(\theta), \vartheta)$ with the linear relation $\tan \theta_a(\theta)/2 = a \tan \theta/2$, and D_a is the dilation operator that maps a sphere without its South pole on itself. $\chi_a^{1/2}(a, \theta)$ is a norm preservation term (i.e. $\mathscr{D}_a$ is unitary):

$$\chi_a^{1/2}(a, \theta) = a^{-1}[1 + \tan^2(\theta/2)]/[1 + a^{-2} \tan^2(\theta/2)] \ . \tag{3.8}$$

3.2.2 Mexican Hat Wavelet

The 2D Mexican hat wavelet transform is the second derivative of a Gaussian:

$$\psi(r) = \frac{1}{\sqrt{2\pi}} \frac{1}{a} \Big(2 - \Big(\frac{r}{a}\Big)^2\Big) e^{-\frac{r^2}{2a^2}} \tag{3.9}$$

where a is a scale factor parameter and r the distance to the wavelet center.

Using the inverse stereographic projection, it is possible to extend the Mexican hat wavelet on the sphere (Antoine 1999; Tenorio et al. 1999; Cayón et al. 2001a; Holschneider 1996; Vielva et al. 2004):

$$\psi_a(r) = \frac{1}{\sqrt{2\pi} C_a} \Big(1 + \Big(\frac{r}{2}\Big)^2\Big)^2 \Big(2 - \Big(\frac{r}{a}\Big)^2\Big) e^{-\frac{r^2}{2a^2}} \tag{3.10}$$

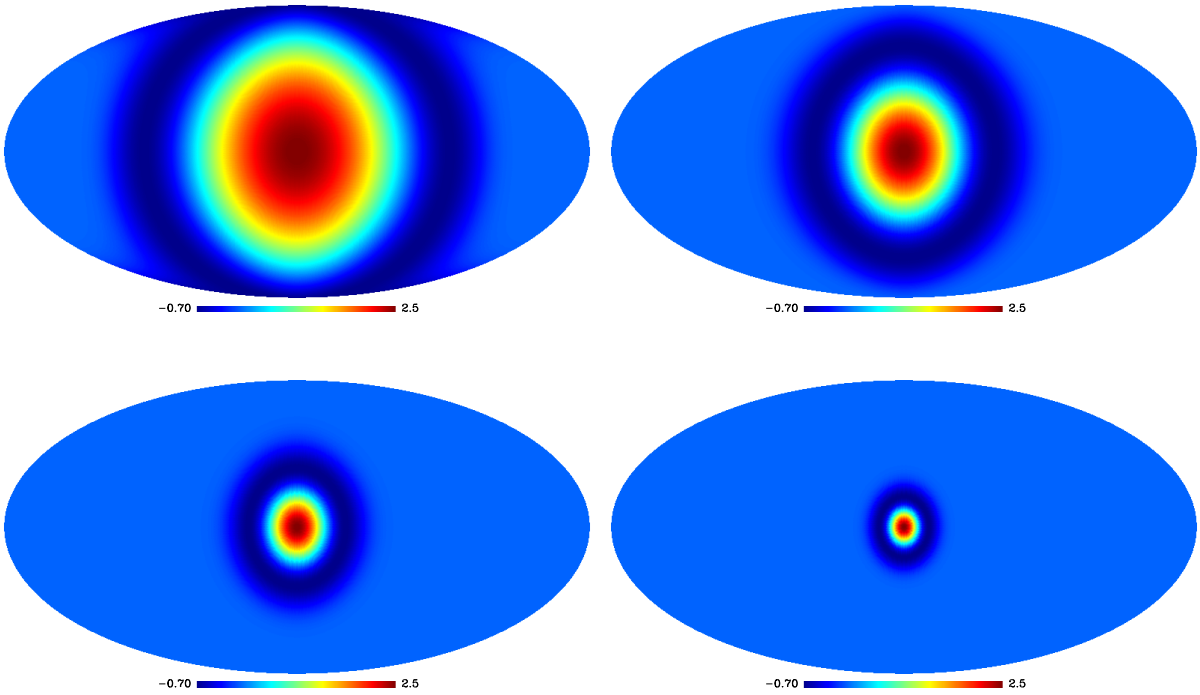


Figure 3.2: Mexican hat on the sphere for the dilation parameter equal to $a = \{1, 2, 4, 8\}$.

where a is a scale factor, C_a is a normalization term $C_a = a\left(1 + \frac{a^2}{2} + \frac{a^4}{4}\right)^{\frac{1}{2}}$, and r is the distance on the tangent plane, which is related to the polar angle θ through $r = 2\tan\frac{\theta}{2}$. This transform may be useful to analyze the data, but it does not have a reconstruction operator, and can therefore not be used for restoration applications.

Fig. 3.2 shows the Mexican hat wavelet on the sphere for four different scales.

3.2.3 Directional Wavelets

To study anisotropic structures, the previously described continuous wavelet transform can be extended to directional wavelets (Antoine et al. 2002; Vielva et al. 2006; McEwen et al. 2007). Fig. 3.3 shows the projection of an elliptic function from the plane to the sphere.

Elongated Mexican Hat Wavelet

The elongated Mexican hat wavelet can be written as:

$$\psi_{a_x,a_y}(\theta,\vartheta) = \sqrt{\frac{2}{\pi}} C(a_x,a_y)\left(1+\tan^2\frac{\theta}{2}\right)\left(1-\frac{4\tan^2\theta/2}{a_x^2+a_y^2}\left(\frac{a_y^2}{a_x^2}\cos^2\vartheta + \frac{a_x^2}{a_y^2}\sin^2\vartheta\right)\right)e^{-2\tan\frac{\theta}{2}\left(\cos^2\vartheta/a_x^2+\sin^2\vartheta/a_y^2\right)} \quad (3.11)$$

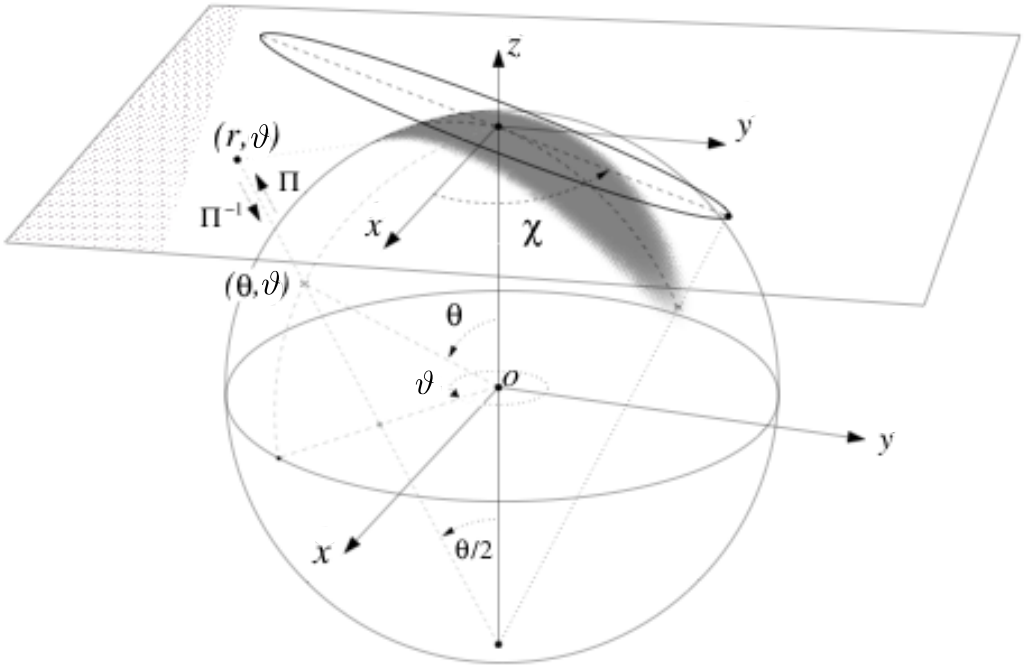


Figure 3.3: Inverse stereographic projections of a directional wavelet on the sphere.

where a_x and a_y are the dilation factors along the two axes O_x and O_y, $C(a_x, a_y)$ is a normalization constant defined as

$$C(a_x, a_y) \quad = \quad (a_x^2 + a_y^2) \left(a_x a_y (3a_x^4 + 3a_y^4 + 2a_x a_y)\right)^{-1/2} \tag{3.12}$$

Fig. 3.4 shows the wavelet functions for different dilation parameters a_x and a_y.

Morlet Wavelet

The Morlet wavelet on the sphere, derived from the stereographic projection of the 2D function on the plane, is:

$$\psi_{a_x,a_y,\mathbf{k}}(\theta,\vartheta) = \sqrt{\frac{2}{\pi}} C(\mathbf{k})(1+\tan^2\frac{\theta}{2})\left(\cos\frac{\mathbf{k}\cdot R^{-1}\mathbf{x}}{\sqrt{2}} - e^{-|\mathbf{k}|^2/4}\right) e^{-2\tan^2(\theta/2)} \tag{3.13}$$

with $R^{-1}\mathbf{x} = (2\tan(\theta/2)\cos\vartheta, 2\tan(\theta/2)\sin\vartheta)$, $\mathbf{k} = (k_x, k_y)$, $|\mathbf{k}|^2 = k_x^2 + k_y^2$, and $C(\mathbf{k}) = (1 + 3e^{-|\mathbf{k}|^2/2} - 4e^{-3|\mathbf{k}|^2/8})^{-1/2}$. $\mathbf{k}$ allows us to control the oscillations of the wavelet functions.

Fig. 3.5 shows the Morlet wavelet for $\mathbf{k}$ equal respectively to $(2,0)$, $(4,0)$, $(6,6)$ and $(9,1)$.

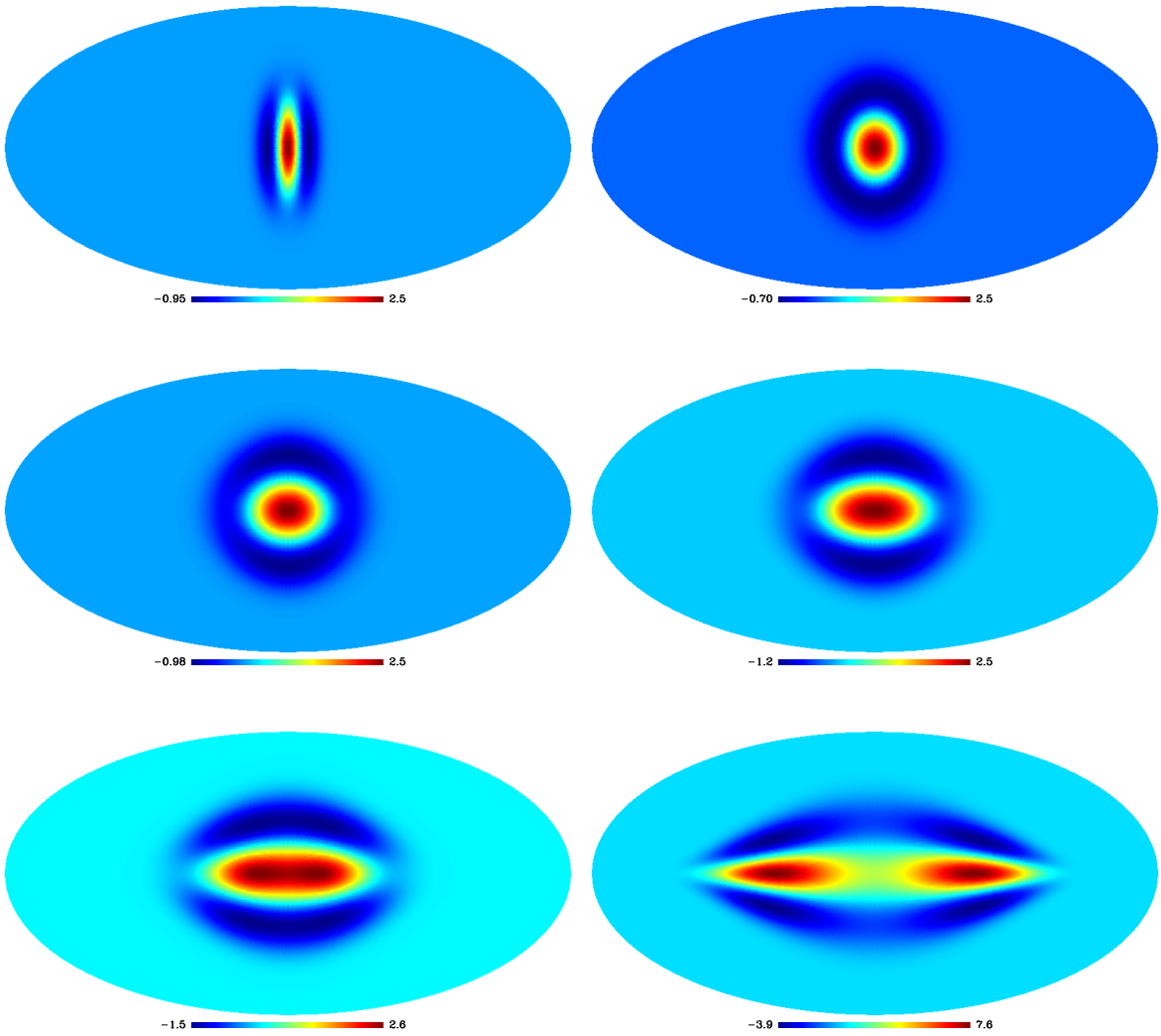


Figure 3.4: Elongated Mexican hat on the sphere for the dilation parameter equal to $a_x = 1$ and $a_y = \{0.5, 1., 1.25, 1.5, 2, 4\}$.

Continuous wavelet transforms have been intensively used in astrophysics, mainly to analyze the Cosmic Microwave Background (Vielva et al. 2004). Directional wavelets based on steerable filters were also proposed in (Wiaux et al. 2006; McEwen et al. 2007). We present in the following a set of multiscale decompositions on the sphere which have a fast exact inverse transform, and are therefore suitable for many applications such as restoration.

3.3 Redundant Wavelet Transform on the Sphere with Exact Reconstruction

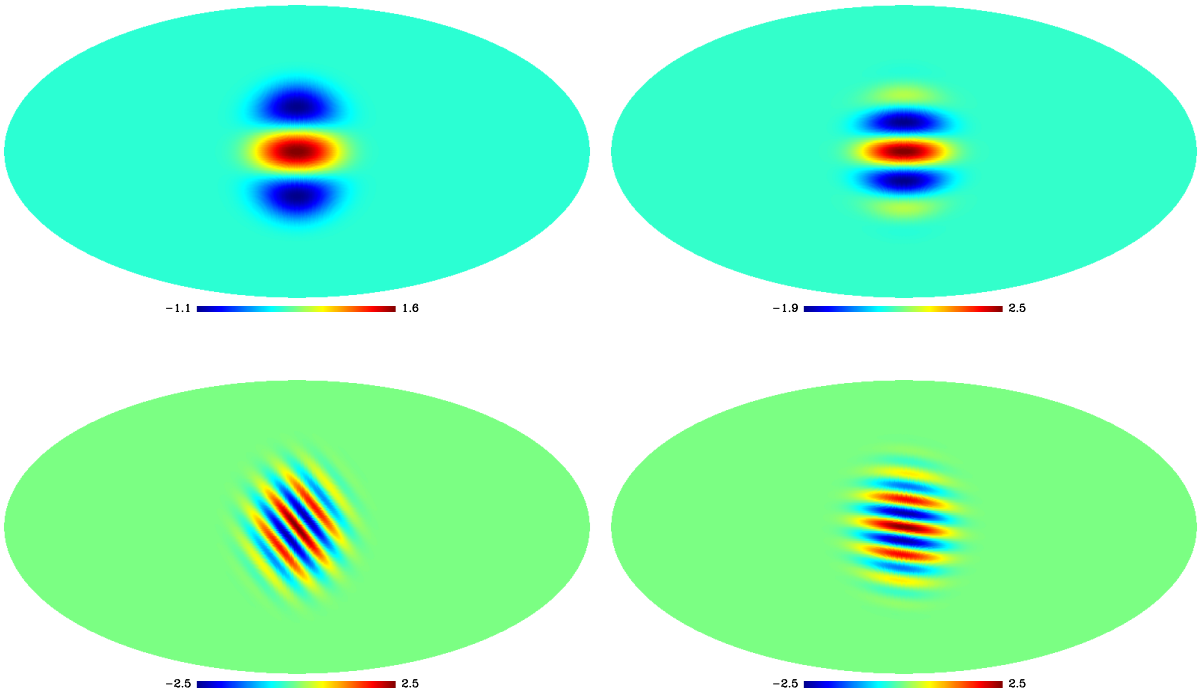


Figure 3.5: Morlet wavelets on the sphere for the parameter **k** equal to (2,0), (4,0), (6,6) et (9,1).

3.3.1 Isotropic Undecimated Wavelet Transform on the Sphere

Here an undecimated isotropic transform (UWTS) is described which is similar in many respects to the undecimated isotropic transform (Starck et al. 2011, 2007a), and will therefore be a good candidate for restoration applications. Its isotropy is a favorable property when analyzing isotropic features. This isotropic transform is obtained using a scaling function $\phi_{l_c}(\theta, \vartheta)$ with cut-off frequency l_c and azimuthal symmetry, meaning that ϕ_{l_c} does not depend on the azimuth ϑ. Hence the spherical harmonics coefficients $\hat{\phi}_{l_c}[l, m]$ of ϕ_{l_c} vanish when $m \neq 0$ so that:

$$\phi_{l_c}(\theta, \vartheta) = \phi_{l_c}(\theta) = \sum_{l=0}^{l=l_c} \hat{\phi}_{l_c}[l, 0] Y_{l0}(\theta, \vartheta). \tag{3.14}$$

Then, convolving a function $f(\theta, \vartheta) \in L_2(S^2)$ with ϕ_{l_c} is greatly simplified and the spherical harmonics coefficients of the resulting map c_0 are readily given by

$$\hat{c}_0[l, m] = \widehat{\phi_{l_c} * f}[l, m] = \sqrt{\frac{2l+1}{4\pi}} \hat{\phi}_{l_c}[l, 0] \hat{f}[l, m]. \tag{3.15}$$

From One Resolution to the Next

A sequence of smoother approximations of f on a dyadic resolution scale can be obtained using the scaling function ϕ_{l_c} as follows:

$$\begin{aligned} c_0 &= \phi_{l_c} * f \\ c_1 &= \phi_{2^{-1}l_c} * f \\ &\dots \\ c_j &= \phi_{2^{-j}l_c} * f, \end{aligned} \tag{3.16}$$

where $\phi_{2^{-j}l_c}$ is a rescaled version of ϕ_{l_c}. The above multiresolution sequence can actually be obtained recursively.

Define a low pass filter h_j for each scale j by:

$$\begin{aligned} \widehat{H}_j[l,m] &= \sqrt{\frac{4\pi}{2l+1}}\hat{h}_j[l,m] \\ &= \begin{cases} \frac{\hat{\phi}_{\frac{l_c}{2^{j+1}}}[l,m]}{\hat{\phi}_{\frac{l_c}{2^j}}[l,m]} & \text{if } l < \frac{l_c}{2^{j+1}} \quad \text{and} \quad m = 0, \\ 0 & \text{otherwise .} \end{cases} \end{aligned} \tag{3.17}$$

It is then easily shown that c_{j+1} derives from c_j by convolution on the sphere with h_j: $c_{j+1} = c_j * h_j$.

The Wavelet Coefficients

Given an axisymmetric wavelet function ψ_{l_c}, we can derive in the same way a high pass filter g_j on each scale j:

$$\begin{aligned} \widehat{G}_j[l,m] &= \sqrt{\frac{4\pi}{2l+1}}\hat{g}_j[l,m] \\ &= \begin{cases} \frac{\hat{\psi}_{\frac{l_c}{2^{j+1}}}[l,m]}{\hat{\phi}_{\frac{l_c}{2^j}}[l,m]} & \text{if } l < \frac{l_c}{2^{j+1}} \quad \text{and} \quad m = 0, \\ 1 & \text{if } l \geq \frac{l_c}{2^{j+1}} \quad \text{and} \quad m = 0, \\ 0 & \text{otherwise .} \end{cases} \end{aligned} \tag{3.18}$$

From this definition, the wavelet coefficients w_{j+1} at scale $j+1$ are obtained from the previous scaling coefficients c_j by a simple convolution on the sphere with g_j: $w_{j+1} = c_j * g_j$.

As in the standard 2D isotropic undecimated transform algorithm, the wavelet coefficients can be defined as the difference between two consecutive resolutions, $w_{j+1}(\theta,\vartheta) = c_j(\theta,\vartheta) - c_{j+1}(\theta,\vartheta)$. This defines a zonal wavelet function ψ_{l_c} as:

$$\hat{\psi}_{\frac{l_c}{2^j}}[l,m] = \hat{\phi}_{\frac{l_c}{2^{j-1}}}[l,m] - \hat{\phi}_{\frac{l_c}{2^j}}[l,m]. \tag{3.19}$$

The high pass filters g_j associated with this wavelet are expressed as:

$$\begin{aligned} \widehat{G}_j[l,m] &= \sqrt{\frac{4\pi}{2l+1}}\hat{g}_j[l,m] \\ &= 1 - \sqrt{\frac{4\pi}{2l+1}}\hat{h}_j[l,m] = 1 - \widehat{H}_j[l,m]. \end{aligned} \tag{3.20}$$

Obviously other wavelet functions could be used just as well. For example, we can define the wavelet function as the difference between the squares of the scaling functions Starck et al. (1998), i.e. $\hat{\psi}^2_{\frac{l_c}{2^j}}[l,m] = \hat{\phi}^2_{\frac{l_c}{2^{j-1}}}[l,m] - \hat{\phi}^2_{\frac{l_c}{2^j}}[l,m]$.

Choice of the Scaling Function

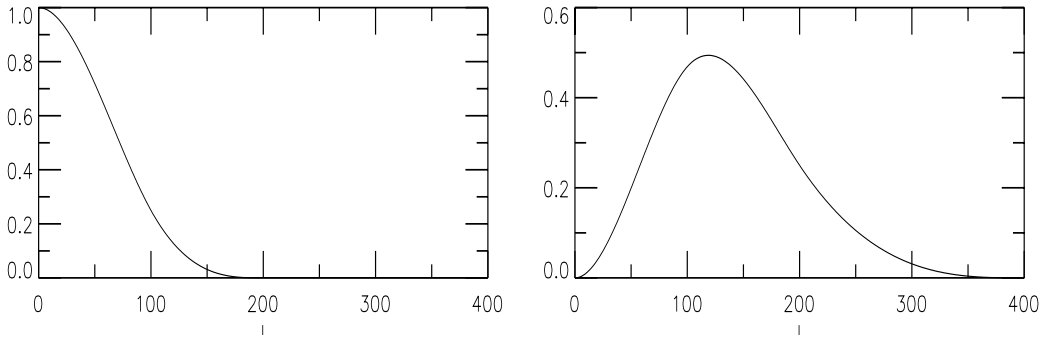


Figure 3.6: On the left, spherical harmonics coefficients $\hat{\phi}[l,0]$ of the the scaling function ϕ and, on the right, those of the wavelet function ψ.

Any function with a cut-off frequency is a possible candidate. The B-spline function of order 3 (Starck et al. 2011, 2007a) leads to :

$$\hat{\phi}_{l_c}[l,m] = \frac{3}{2} B_3\left(\frac{2l}{l_c}\right) \tag{3.21}$$

where $B_3(t)$ is the scaling function:

$$B_3(t) = \frac{1}{12}(\mid t-2 \mid^3 - 4 \mid t-1 \mid^3 + 6 \mid t \mid^3 - 4 \mid t+1 \mid^3 + \mid t+2 \mid^3) \tag{3.22}$$

In Fig. 3.6 the spherical harmonics coefficients of the scaling function derived from a B_3-spline, and those of the associated wavelet function (3.19), are plotted as a function of l. Other functions such as Meyer wavelets or the needlet function (Marinucci et al. 2008) can be used as well.

Meyer and needlet wavelet functions have both a much better frequency localization than the wavelet function derived from the B_3-spline, and, as nothing is perfect, the price to pay is more oscillations in the direct space. To illustrate this, we show in Fig. 3.7 different wavelet functions. Top left and right shows the respectively the spline and the needet wavelet function at a given scale. Fig. 3.7 middle left shows a cut of the healpix face with contains the previous functions. Middle right is the same functions, but we have plotted the absolute value in order to better visualize their respective ringing. As it can be seen, for wavelet functions with the same main lob, the needlet wavelet oscillate much more than the spline wavelet. Fig. 3.7 bottom left and right compares Meyer and needlet functions. They are relatively close and share the same property of good frequency localization, but with more oscillations than the spline wavelet function. Hence, the best wavelet choice certainly depends on the final applications. For statistical analysis, detection or restoration applications, we may prefer to use a wavelet which does not oscillate too much and with a smaller support, and the spline wavelet is clearly the correct choice. For spectral or bispectral analysis, where the frequency localization is fundamental, then Meyer or needlet shoud be preferred to the spline wavelet.

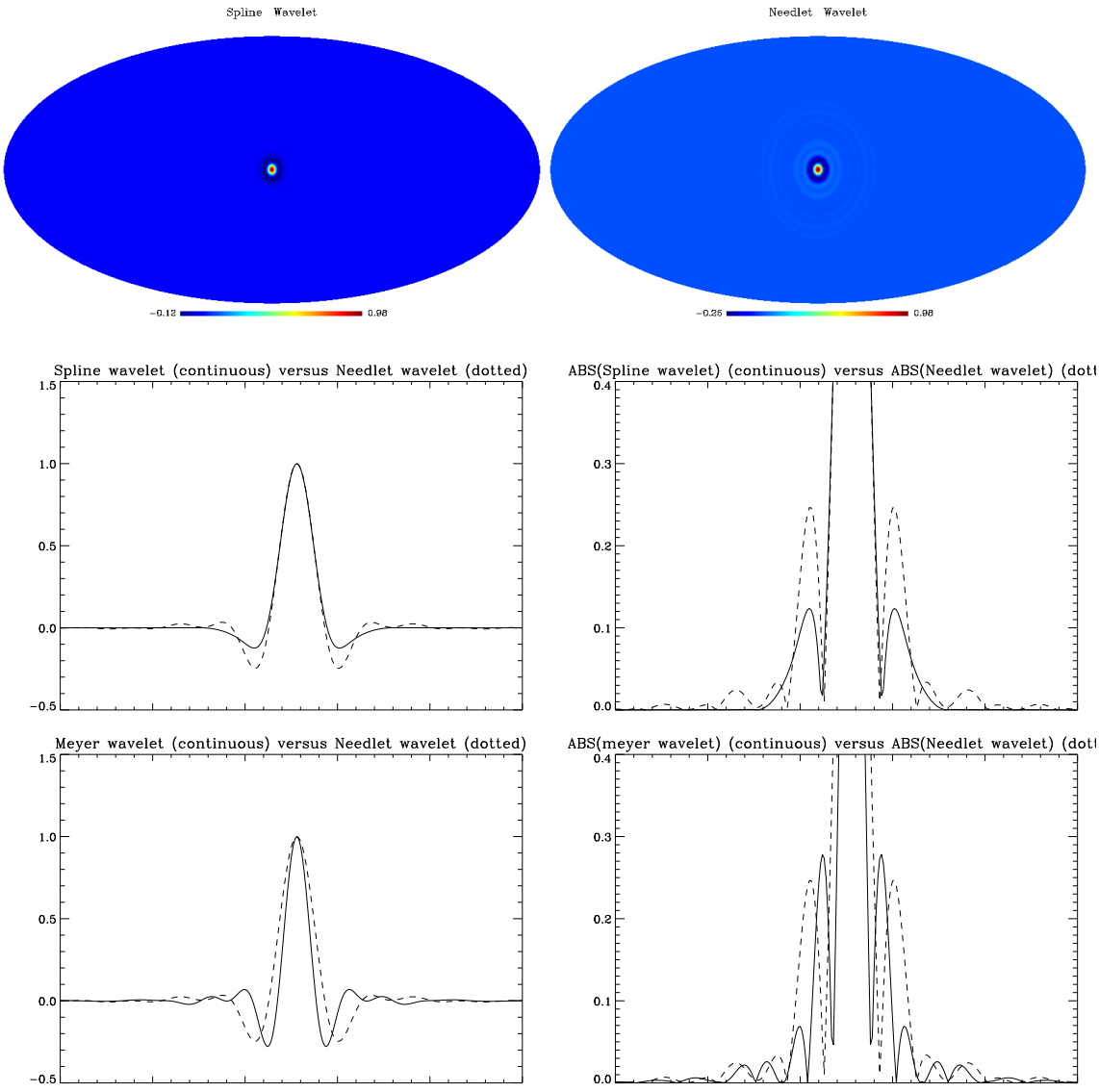


Figure 3.7: Comparaison between spline, needlet, and Meyer wavelet functions on the sphere.

The steps of the UWT on the sphere of a discrete image X sampled from f are summarized in Algorithm 1. If the wavelet function corresponds to the choice (3.19), Step 3 in this UWTS algorithm reduces to $w_{j+1} = c_j - c_{j+1}$.

Fig. 3.8 shows the Mars topographic map (top left) [1] and its wavelet transform, using five scales (four wavelet scales + coarse scale). The sum of the five scales reproduces exactly the original image.

Inverse Transform

[1]The Mars Orbiter Laser Altimeter (MOLA) generated altimetry profiles used to create global topographic maps. The MOLA instrument stopped acquiring altimetry data on June 30, 2001, and after that operated in passive radiometry mode until the end of the Mars Global Surveyor mission. MOLA data sets are produced by the MOLA Science Team and archived by the PDS Geosciences Node.

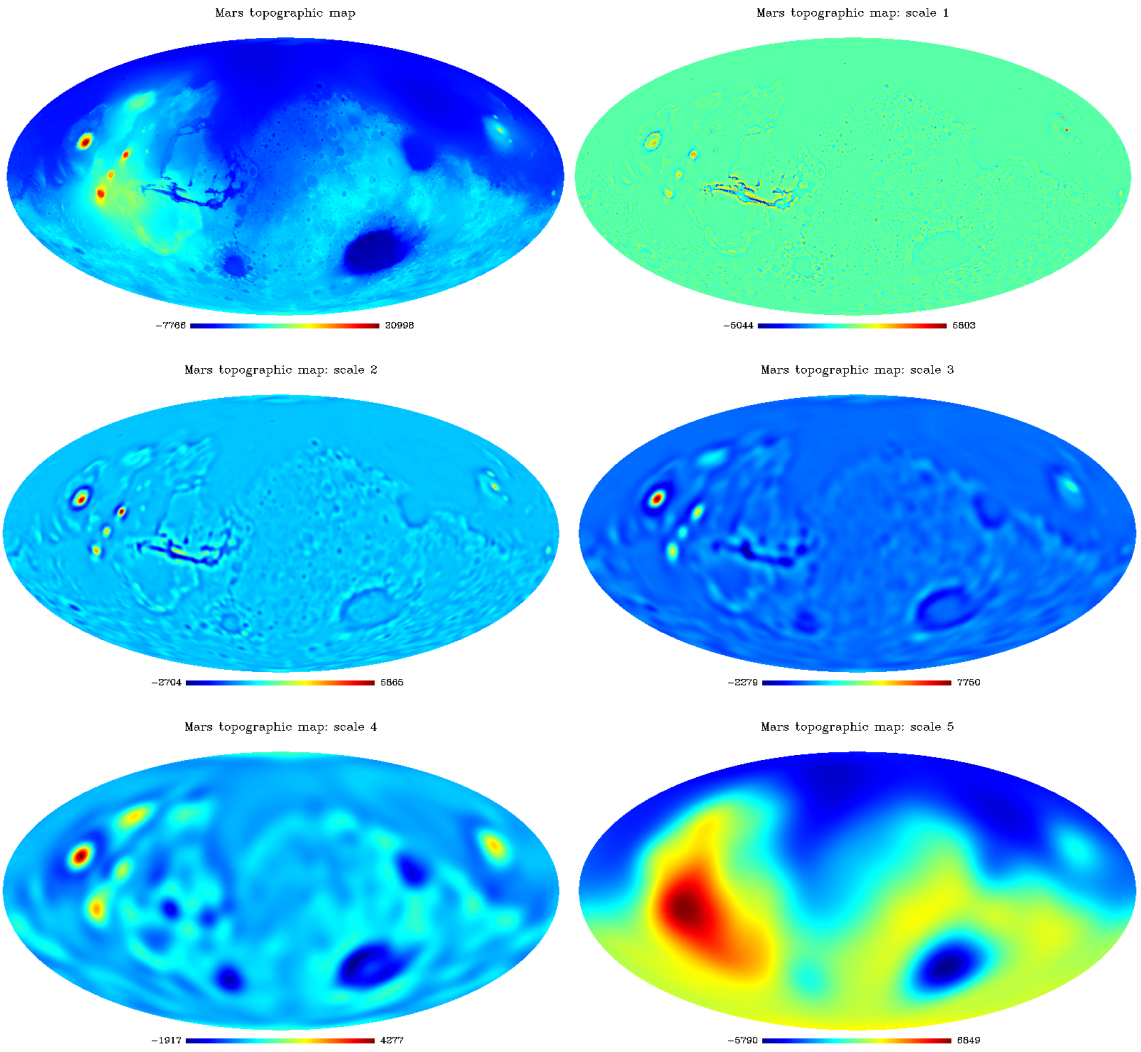


Figure 3.8: Mars topographic map and its UWTS (four wavelet detail scales and the scaling (smooth) band).

If the wavelet is the difference between two resolutions, a straightforward reconstruction of an image from its wavelet coefficients $\mathcal{W} = \{w_1, \ldots, w_J, c_J\}$ is:

$$c_0(\theta, \vartheta) = c_J(\theta, \vartheta) + \sum_{j=1}^{J} w_j(\theta, \vartheta). \tag{3.23}$$

This reconstruction formula is the same as with the 2D undecimated isotropic wavelet algorithm (i.e. with à trous agorithm).

But since the transform is redundant there is actually no unique way to reconstruct an image from its coefficients see (Starck et al. 2007a). Indeed, using the relations:

$$\begin{aligned} \hat{c}_{j+1}[l,m] &= \widehat{H}_j[l,m]\hat{c}_j[l,m] \\ \hat{w}_{j+1}[l,m] &= \widehat{G}_j[l,m]\hat{c}_j[l,m] \end{aligned} \tag{3.24}$$

Algorithm 1 The Undecimated Wavelet Transform on the Sphere.

Task: Compute the UWTS of a discrete X.
Parameters: Data samples X and number of of wavelet scales J.
Initialization:

- $c_0 = X$.
- Compute the B_3-spline scaling function and derive $\hat{\psi}$, $\widehat{H}$ and $\widehat{G}$ numerically.
- Compute the corresponding spherical harmonics transform of c_0.

for $j = 0$ to $J - 1$ **do**

1. Compute the spherical harmonics transform of the scaling coefficients: $\hat{c}_{j+1} = \hat{c}_j \widehat{H}_j$.
2. Compute the inverse spherical harmonics transform of $\hat{c}_{j+1}$ to get c_{j+1}.
3. Compute the spherical harmonics transform of the wavelet coefficients: $\hat{w}_{j+1} = \hat{c}_j \widehat{G}_j$.
4. Compute the inverse spherical harmonics transform of $\hat{w}_{j+1}$ to get w_{j+1}.

Output: $\mathcal{W} = \{w_1, w_2, \ldots, w_J, c_J\}$ the UWTS of X.

a least-squares estimate of c_j from c_{j+1} and w_{j+1} gives:

$$\hat{c}_j = \hat{c}_{j+1} \widehat{\tilde{H}}_j + \hat{w}_{j+1} \widehat{\tilde{G}}_j \ , \tag{3.25}$$

where the dual filters $\tilde{h}$ and $\tilde{g}$ satisfy:

$$\begin{aligned} \widehat{\tilde{H}}_j &= \sqrt{\frac{4\pi}{2l+1}} \hat{\tilde{h}}_j = \widehat{H}_j^* / \left(\left| \widehat{H}_j \right|^2 + \left| \widehat{G}_j \right|^2 \right) \\ \widehat{\tilde{G}}_j &= \sqrt{\frac{4\pi}{2l+1}} \hat{\tilde{g}}_j = \widehat{G}_j^* / \left(\left| \widehat{H}_j \right|^2 + \left| \widehat{G}_j \right|^2 \right) . \end{aligned} \tag{3.26}$$

For the scaling function which is a B_3-spline function and a wavelet taken as the difference between two resolutions, the corresponding conjugate low pass and high pass filters $\widehat{\tilde{H}}$

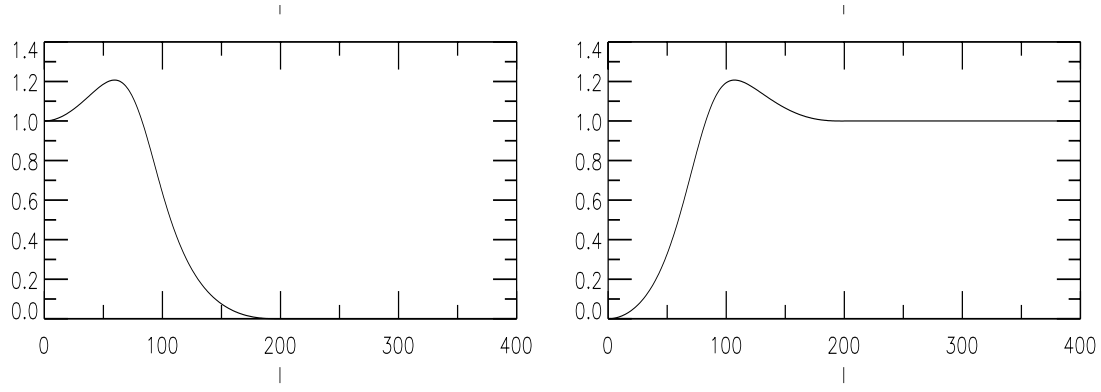


Figure 3.9: On the left, the filter $\hat{\tilde{h}}$, and on the right the filter $\hat{\tilde{g}}$.

and $\widehat{\tilde{G}}$ are plotted in Fig. 3.9. The reconstruction algorithm is given in Algorithm 2.

Algorithm 2 Inverse UWT on the sphere.
Task: Reconstruct an image from its UWTS coefficients.
Parameters: UWTS coefficients $\mathcal{W} = \{w_1, w_2, \ldots, w_J, c_J\}$.
Initialization:

- Compute the B_3-spline scaling function and derive $\hat{\psi}$, $\widehat{H}$, $\widehat{G}$, $\widehat{\tilde{H}}$ and $\widehat{\tilde{G}}$ numerically.
- Compute the spherical harmonics transform of c_J to get $\hat{c}_J$.

for $j = J - 1$ to 0, with step $= -1$ **do**

1. Compute the spherical harmonics transform of the wavelet coefficients w_{j+1} to get $\hat{w}_{j+1}$.
2. Multiply $\hat{c}_{j+1}$ by $\widehat{\tilde{H}}_j$.
3. Multiply $\hat{w}_{j+1}$ by $\widehat{\tilde{G}}_j$.
4. Get the spherical harmonics of $\hat{c}_j = \hat{c}_{j+1} + \hat{w}_{j+1}$.

Compute The inverse Spherical Harmonics transform of $\hat{c}_0$.
Output: c_0 is the inverse UWT on the sphere.

Fig. 3.10 shows the reconstruction by setting all wavelet coefficients but one at different scales and positions. Depending on the position and scale of the non-zero coefficient, the reconstructed map shows an isotropic feature at different scales and positions.

3.3.2 Isotropic Pyramidal Wavelet Transform on the Sphere

Forward Transform

In the previous algorithm, no down-sampling is performed and each scale of the wavelet decomposition has the same number of pixels as the original data set. Therefore the number of pixels in the decomposition is equal to the number of pixels in the data multiplied by the number of scales. For some applications, we may prefer to introduce a decimation in the decomposition so as to reduce the required memory size and the computation time. This can be done easily by using a specific property of the chosen scaling function. Indeed, since we are considering here a scaling function with an initial cut-off l_c in spherical harmonic multipole number l, and since the actual cut-off is reduced by a factor of two at each step, the number of significant spherical harmonics coefficients is then reduced by a factor of four after each convolution with the low pass filter h. Therefore, we need less pixels in the direct space when we compute the inverse spherical harmonics transform. Using the HEALPix pixelization scheme (Górski et al. 2005), this can be done easily by dividing by 2 the $N_{\rm side}$ parameter when calling the inverse spherical harmonics transform routine. The pyramidal wavelet transform on the sphere algorithm is given in Algorithm 3.

Fig. 3.11 shows an Earth image and its pyramidal wavelet transform (PWTS) using five scales. As the scale number increases (i.e. the resolution decreases), the pixel size becomes larger. The data are land and sea-floor elevations obtained from the ETOPO5

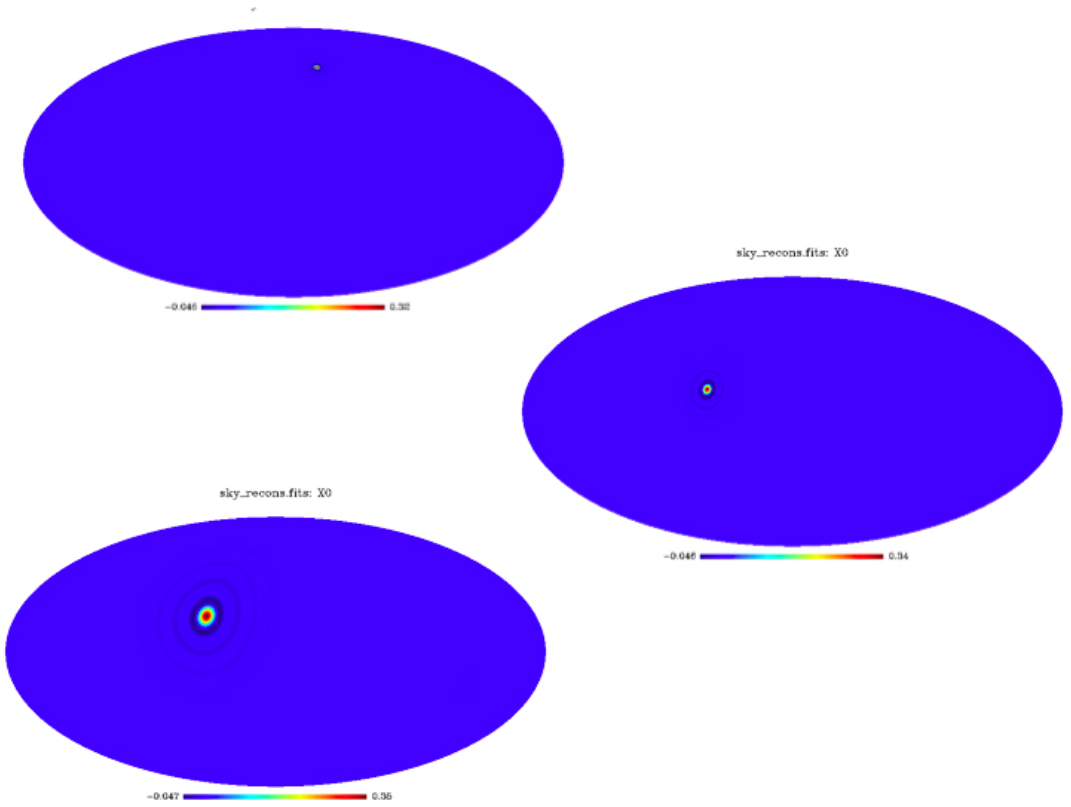


Figure 3.10: Reconstruction from a single wavelet coefficient at different scales. Each map is obtained by setting all wavelet coefficients to zero but one, and by applying an inverse UWTS. Depending on the position and scale of the non-zero coefficient, the reconstructed map shows an isotropic feature at different scales and positions.

5-minute gridded elevation data set. A thorough explanation of the data set is provided at `www.ngdc.noaa.gov` [2].

Inverse Transform

This reconstruction is not as straightforward as in the undecimated case, since the different scales do not have the same resolution. For each resolution level, we have to up-sample the scaling band before co-adding it to the wavelet coefficients. Algorithm 4 describes this.

The wavelet transform on the sphere and its pyramidal version have both a reconstruction operator, so they are very well designed for any restoration application when the data contains isotropic features. In the following, we present other transforms on the sphere more adapted to the analysis of anisotropic features.

[2]The ETOPO5 data are credited to "Data Announcement 88-MGG-02, Digital relief of the Surface of the Earth. NOAA, National Geophysical Data Center, Boulder, Colorado, 1988". The HEALPix image is available at `http://astro.ic.ac.uk/~pdineen/earth/index.html#earthmap`.

Algorithm 3 Pyramidal wavelet transform on the sphere.
Task: Compute the pyramidal WT on the sphere of a discrete image X.
Parameters: Data X and number of of wavelet scales J.
Initialization:

- $c_0 = X$.
- Compute the B_3-spline scaling function and derive $\hat{\psi}$, $\widehat{H}$ and $\widehat{G}$ numerically.
- Compute the corresponding spherical harmonics transform of c_0.

for $j = 0$ to $J - 1$ **do**

1. Compute the spherical harmonics transform of the scaling coefficients: $\hat{c}_{j+1} = \hat{c}_j \widehat{H}_j$.
2. Compute the inverse spherical harmonics transform of $\hat{c}_{j+1}$ to get c_{j+1}.
3. Down-sample c_{j+1}, since its support in the spherical harmonic domain has been divided by two.
4. Compute the spherical harmonics transform of the wavelet coefficients: $\hat{w}_{j+1} = \hat{c}_j \widehat{G}_j$.
5. Compute the inverse spherical harmonics transform of $\hat{w}_{j+1}$ to get w_{j+1}.

Output: $\mathcal{W} = \{w_1, w_2, \ldots, w_J, c_J\}$ the Pyramidal WT on sphere of X.

3.4 Ridgelet and Curvelet Transform on the Sphere (CTS)

3.4.1 Introduction.

The 2D curvelet transform, proposed in (Donoho and Duncan 2000; Starck et al. 2002a, 2003a), enables the directional analysis of an image in different scales. The fundamental property of the curvelet transform is to analyze the data with functions of length about $2^{-j/2}$ for the j^{th} sub-band $[2^j, 2^{j+1}]$ of the two dimensional wavelet transform. Following the implementation described in (Starck et al. 2002a, 2003a), the data first undergoes an Isotropic Undecimated Wavelet Transform (i.e. "à trous" algorithm). Each scale j is then decomposed into smoothly overlapping blocks of side-length B_j pixels in such a way that the overlap between two vertically adjacent blocks is a rectangular array of size $B_j \times B_j/2$. And finally, the ridgelet transform (Candès and Donoho 1999b) is applied on each individual block. Recall that the ridgelet transform precisely amounts to applying a 1-dimensional wavelet transform to the slices of the Radon transform. More details on the implementation of the digital curvelet transform can be found in (Starck et al. 2002a, 2003a). It has been shown that the curvelet transform could be very useful for the detection and the discrimination of non-Gaussianity in CMB (Starck et al. 2003b). The curvelet transform is also redundant, with a redundancy factor of $16J + 1$ whenever J scales are employed. Its complexity scales like that of the ridgelet transform that is as $O(n^2 \log_2 n)$. The curvelet transform was shown to sparsely represent anisotropic structures and smooth curves and edges of different lengths.

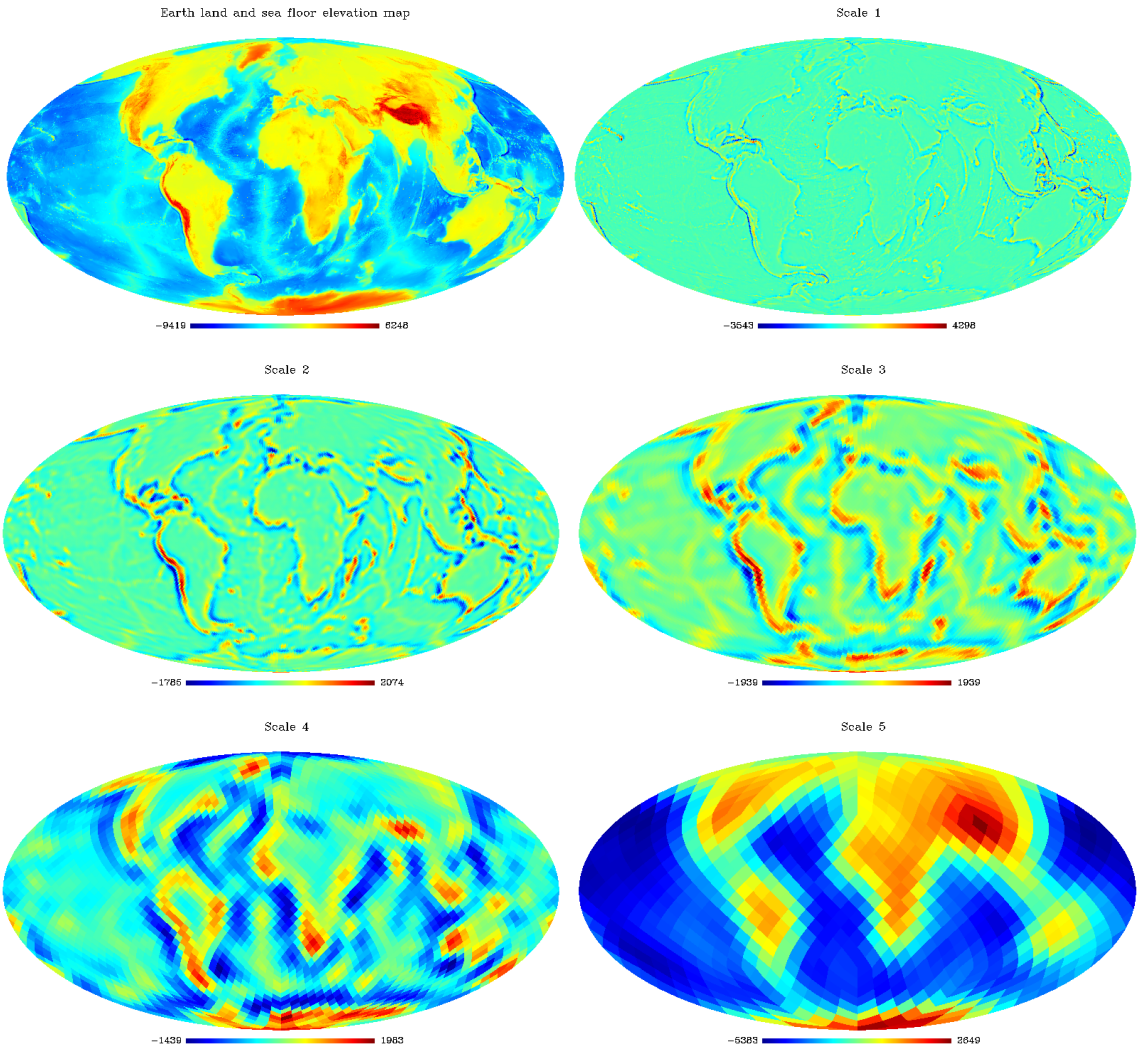


Figure 3.11: Pyramidal wavelet transform on the sphere.

.

3.4.2 Ridgelets and Curvelets on the Sphere.

The Curvelet transform on the sphere (CTS) can be similar to the 2D digital curvelet transform, but replacing the "à trous" algorithm by the Isotropic Wavelet Transform on the Sphere previously described. The CTS algorithm consists in the following three steps which we describe in more details next.

- *Isotropic Wavelet Transform on the Sphere.*
- *Partitioning.* Each scale is decomposed into blocks of an appropriate scale (of side-length $\sim 2^{-s}$),using the HEALPix pixelization.
- *Ridgelet Analysis.* Each square is analyzed via the discrete ridgelet transform.

We now describe these three steps.

Algorithm 4 Inverse Pyramidal Wavelet Transform on the sphere.
Task: Reconstruct an image from its pyramidal WT on the sphere.
Parameters: Pyramidal WT coefficients $\mathcal{W} = \{w_1, w_2, \ldots, w_J, c_J\}$.
Initialization:

- Compute the B_3-spline scaling function and derive $\hat{\psi}$, $\widehat{H}$, $\widehat{G}$, $\widehat{\tilde{H}}$ and $\widehat{\tilde{G}}$ numerically.
- Compute the spherical harmonics transform of c_J to get $\hat{c}_J$.

for $j = J - 1$ to 0 **do**

1. Upsample c_{j+1} to the resolution of c_j.
2. Compute the spherical harmonics transform of the wavelet coefficients w_{j+1} to get $\hat{w}_{j+1}$.
3. Multiply $\hat{c}_{j+1}$ by $\widehat{\tilde{H}}_j$.
4. Multiply $\hat{w}_{j+1}$ by $\widehat{\tilde{G}}_j$.
5. Get the spherical harmonics of $\hat{c}_j = \hat{c}_{j+1} + \hat{w}_{j+1}$.

Compute The inverse spherical harmonics transform of $\hat{c}_0$.
Output: c_0 is the inverse pyramidal WT on the sphere.

Partitioning Using the HEALPix Representation

The HEALPix representation is a curvilinear hierarchical partition of the sphere into quadrilateral pixels of exactly equal area but with varying shape. The base resolution divides the sphere into 12 quadrilateral faces of equal area placed on three rings around the poles and equator. Each face is subsequently divided into N_{side}^2 pixels following a quadrilateral multiscale tree structure (see Fig. 2.1). The pixel centers are located on iso-latitude rings, and pixels from the same ring are equispaced in azimuth. This is critical for computational speed of all operations involving the evaluation of spherical harmonics transforms, including standard numerical analysis operations such as convolution, and power spectrum estimation.

An important geometrical feature of the HEALPix sampling grid is the hierarchical quadrilateral tree structure. This defines a natural one-to-one mapping of the sphere sampled according to the HEALPix grid, into twelve flat images, on all scales. It is then easy to partition a spherical map using HEALPix into quadrilateral blocks of a specified size. One first extracts the twelve base-resolution faces, and each face is then decomposed into overlapping blocks of the specified size. This decomposition into blocks is an essential step of the traditional flat 2D curvelet transform. Based on the reversible warping of the sphere into a set of flat images made possible by the HEALPix sampling grid, the ridgelet and curvelet transforms can be extended to the sphere.

With the decomposition into blocks described above, there is no overlap between neighboring blocks belonging to different base-resolution faces. This may result for instance in blocking effects in denoising experiments via nonlinear filtering. It is possible to overcome this difficulty in some sense by working simultaneously with various rotations of the data with respect to the sampling grid. This will average out undesirable effects at edges

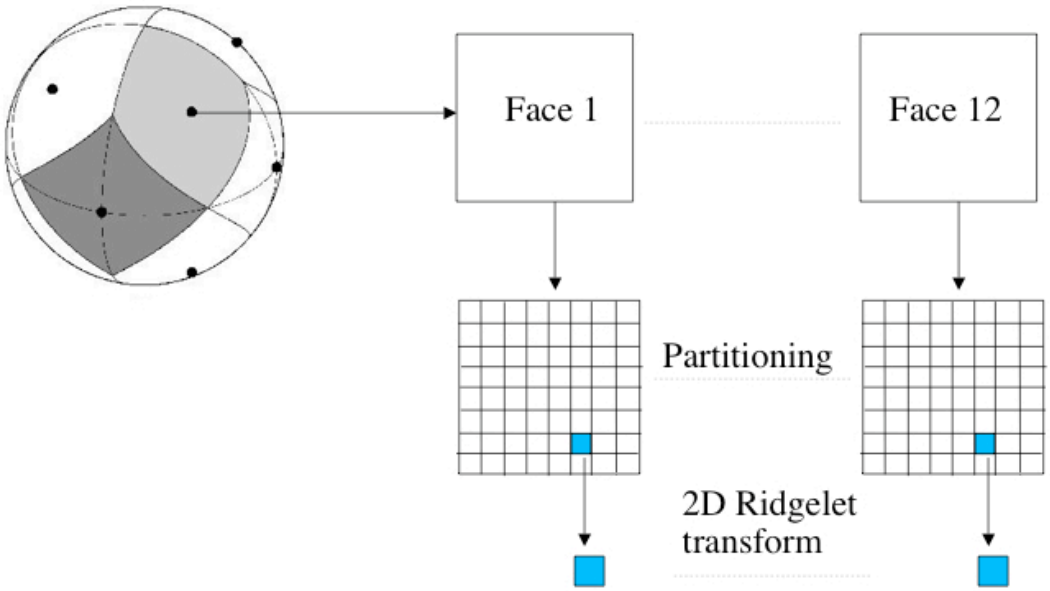


Figure 3.12: Flowgraph of the ridgelet transform on the sphere.

between base resolution faces.

Ridgelet transform

Once the partitioning is performed, the standard 2D ridgelet transform described in (Starck et al. 2003a) is applied in each individual block :

1. Compute the 2D Fourier transform.
2. Extract lines going through the origin in the frequency plane.
3. Compute the 1D inverse Fourier transform of each line. We get the Radon transform.
4. Compute the 1D wavelet transform of the lines of the Radon transform.

The first three steps correspond to a Radon transform method called the *linogram*. Other implementations of the Radon transform, such as the *Slant Stack Radon Transform* (Donoho and Flesia 2002), can be used as well, as long as they offer an exact reconstruction.

Fig. 3.12 shows the flowgraph of the ridgelet transform on the sphere and Fig. 3.13 shows the reconstruction from a single ridgelet coefficient at different scales and orientations.

3.4.3 Curvelet Transform Algorithm

The curvelet transform algorithm on the sphere is described in Algorithm 5.

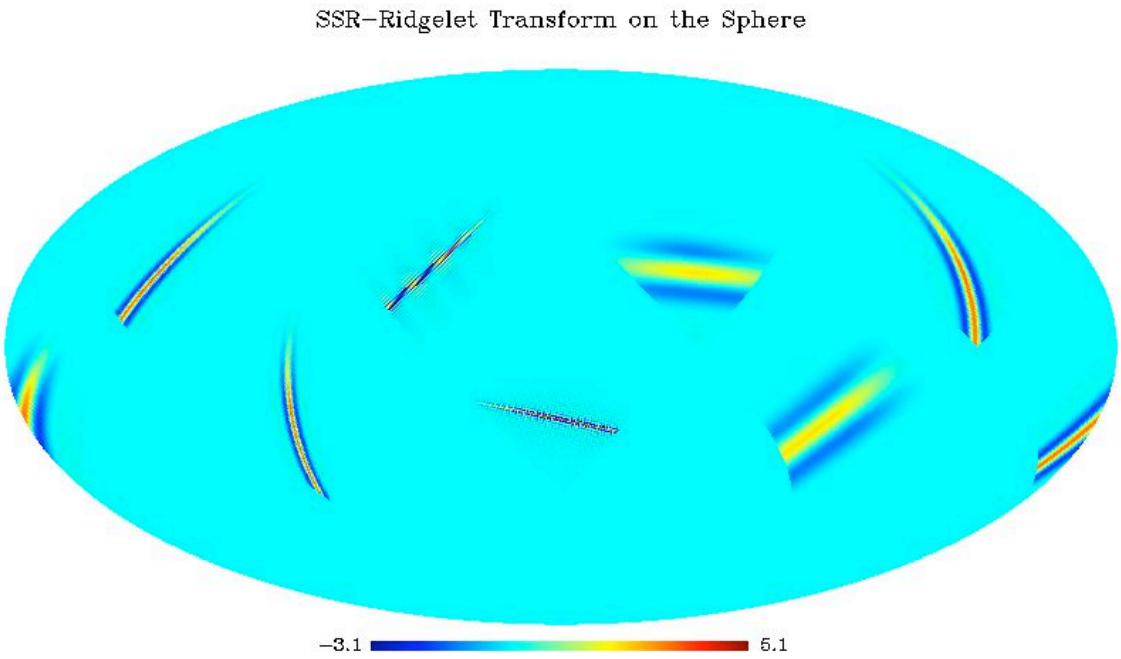


Figure 3.13: Ridgelet atoms on the sphere obtained by reconstruction from a few ridgelet coefficient at different scales and orientations.

The sidelength of the localizing windows is doubled *at every other* dyadic subband, hence maintaining the fundamental property of the curvelet transform which says that elements of length about $2^{-j/2}$ serve for the analysis and synthesis of the j^{th} subband $[2^j, 2^{j+1}]$. We used the default value $B_{\min} = 16$ pixels in our implementation. Fig. 3.14 gives an overview of the organization of the algorithm.

Fig. 3.15 shows the backprojection of curvelet coefficients at different scales and orientations.

3.4.4 Pyramidal Curvelet Transform on the Sphere (PCTS)

The CTS is very redundant, which may be a problem for handling huge data sets such as Planck data (see Section 5.3.2 below). The redundancy can be reduced by substituting, in the curvelet transform algorithm, the pyramidal wavelet transform with the undecimated wavelet transform. The second step which consists of applying the ridgelet transform on the wavelet scale is unchanged. The pyramidal curvelet transform (PCTS) algorithm is summarized in Algorithm 6.

Algorithm 5 Curvelet Transform on the sphere.

Task: Compute the curvelet transform on the sphere of a discrete image X.
Parameters: Image X and number of scales J.
Initialization:

- $B_1 = B_{\min}$.
- Compute the isotropic UWTS of X with J scales, get $\{w_1, \ldots, w_J, c_J\}$.

for $j = 0$ to $J - 2$ **do**

1. Partition the wavelet subband w_j with a block size B_j.
2. Apply the digital ridgelet transform to each block; get the curvelet coefficients at scale j.

if j modulo $2 = 1$ **then** $B_{j+1} = 2B_j$, else $B_{j+1} = B_j$.

Output: The curvelet transform on the sphere of X.

Algorithm 6 Pyramidal Curvelet Transform on the sphere.

Task: Compute the pyramidal curvelet transform on the sphere of a discrete image X.
Parameters: Image X and number of scales J.
Initialization:

- $B_1 = B_{\min}$.
- Compute the pyramidal wavelet transform of X with J scales, get $\{w_1, \ldots, w_J, c_J\}$.

for $j = 0$ to $J - 2$ **do**

1. Partition the wavelet subband w_j with a block size B_j.
2. Apply the digital ridgelet transform to each block; get the curvelet coefficients at scale j.

if j modulo $2 = 1$ **then** $B_{j+1} = 2B_j$, else $B_{j+1} = B_j$.

Output: The pyramidal curvelet transform on the sphere of X.

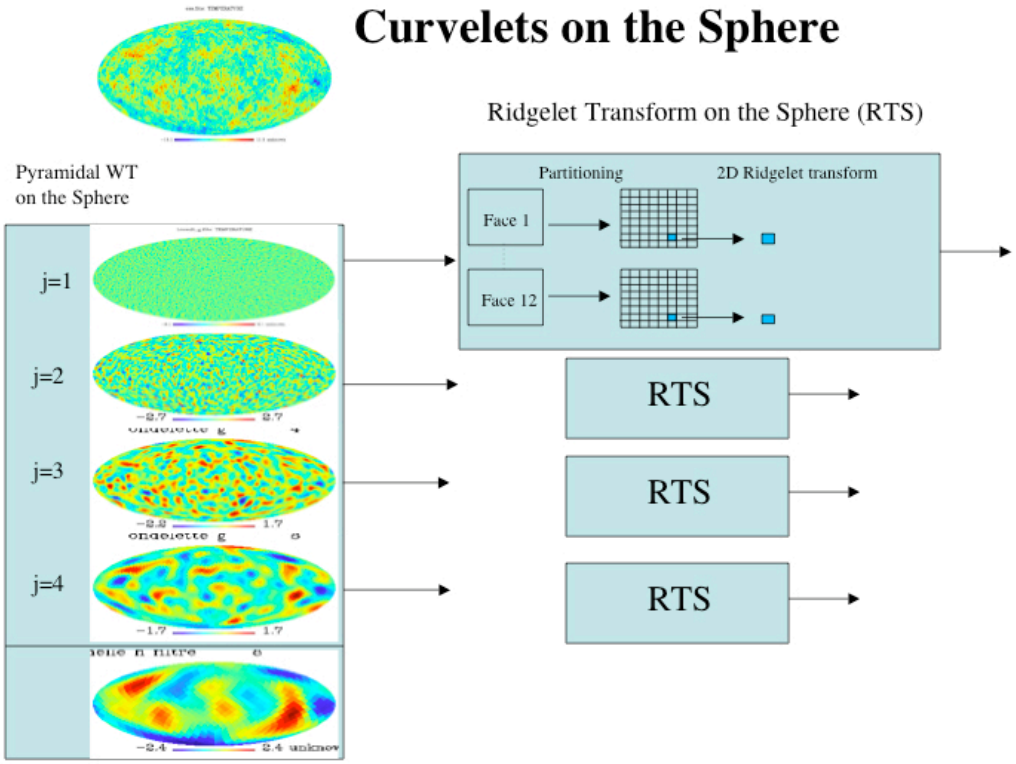


Figure 3.14: Flowgraph of the curvelet transform on the sphere.

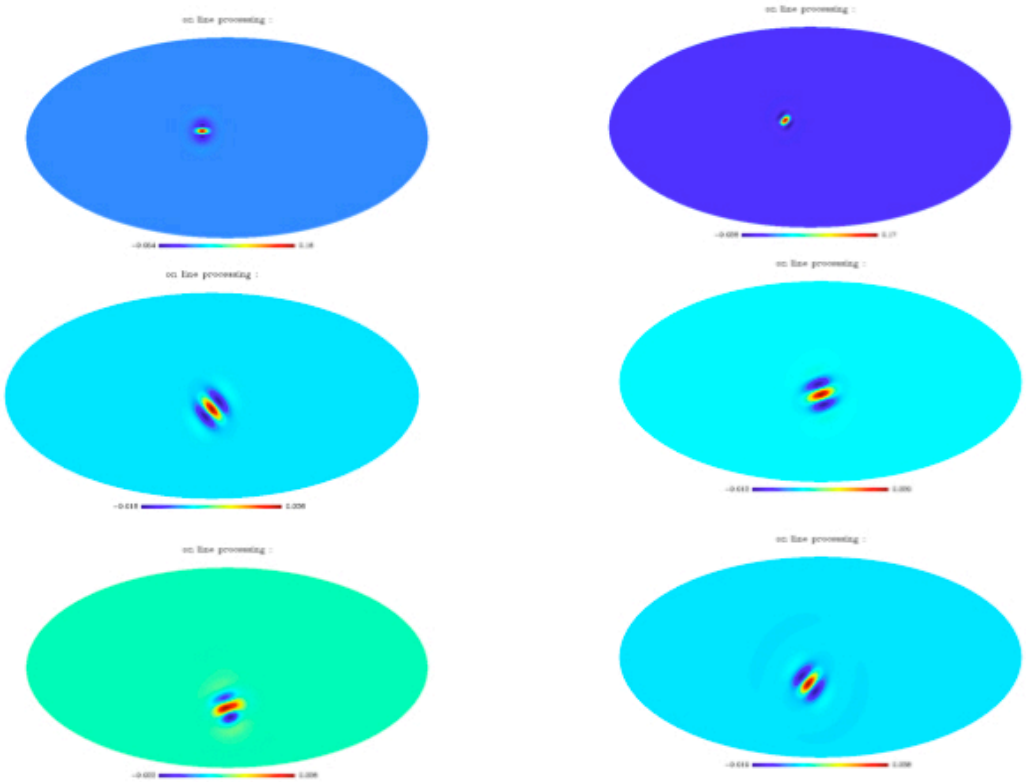


Figure 3.15: Reconstruction from a single curvelet coefficient at different scales and orientations.

Chapter 4

Data Restoration on the Sphere

4.1 Introduction

Wavelets and Curvelets have been used successfully for image denoising via non-linear filtering or thresholding methods (Starck and Murtagh 2006; Starck et al. 2010). Hard thresholding, for instance, consists in setting all insignificant coefficients (i.e. coefficients with an absolute value below a given threshold) to zero. In practice, we need to estimate the noise standard deviation σ_j in each band j and a wavelet (or curvelet) coefficient w_j is significant if $\mid w_j \mid > k\sigma_j$, where k is a user-defined parameter, typically chosen between 3 and 5. The σ_j estimation in band j can be derived from simulations (Starck and Murtagh 2006). Denoting D the noisy data and δ the thresholding operator, the filtered data $\tilde{D}$ are obtained by :

$$\tilde{D} = \mathcal{R}\delta(\mathcal{T}D) \tag{4.1}$$

where $\mathcal{T}$ is the wavelet (resp. curvelet) transform operator and $\mathcal{R}$ is the wavelet (resp. curvelet) reconstruction operator.

4.2 Significant Wavelet Coefficients

4.2.1 Definition

In most applications, it is necessary to know if a wavelet coefficient is due to signal (i.e. it is significant) or to noise.

The wavelet (resp. curvelet) transform yields a set of resolution-related views of the input image. A wavelet (resp. curvelet) band at level j has coefficients given by $w_{j,k}$. If we obtain the distribution of the coefficient $w_{j,k}$ for each band of the decomposition, based on the noise, we can introduce a statistical significance test for this coefficient. This procedure is the classical significance-testing one. Let $\mathcal{H}_0$ be the hypothesis that the image is locally constant at scale j. Rejection of hypothesis $\mathcal{H}_0$ depends (for positive coefficient values) on:

$$P = Prob(\mid w_{j,k} \mid < \tau \mid \mathcal{H}_0) \tag{4.2}$$

The detection threshold, τ, is defined for each scale. Given an estimation threshold, ϵ, if $P = P(\tau) > \epsilon$ the null hypothesis is not excluded. Although non-null, the value of the

coefficient could be due to noise. On the other hand, if $P < \epsilon$, the coefficient value cannot be due to the noise alone, and so the null hypothesis is rejected. In this case, a significant coefficient has been detected.

4.2.2 Noise Modeling

If the distribution of $w_{j,l}$ is Gaussian, with zero mean and standard deviation σ_j, we have the probability density

$$p(w_{j,l}) = \frac{1}{\sqrt{2\pi}\sigma_j} e^{-w_{j,l}^2/2\sigma_j^2} \tag{4.3}$$

Rejection of hypothesis $\mathcal{H}_0$ depends (for a positive coefficient value) on:

$$P = Prob(w_{j,l} > W) = \frac{1}{\sqrt{2\pi}\sigma_j} \int_{w_{j,l}}^{+\infty} e^{-W^2/2\sigma_j^2} dW \tag{4.4}$$

and if the coefficient value is negative, it depends on

$$P = Prob(w_{j,l} < W) = \frac{1}{\sqrt{2\pi}\sigma_j} \int_{-\infty}^{w_{j,l}} e^{-W^2/2\sigma_j^2} dW \tag{4.5}$$

Given stationary Gaussian noise, it suffices to compare $w_{j,l}$ to $k\sigma_j$. Often k is chosen as 3, which corresponds approximately to $\epsilon = 0.002$. If $w_{j,l}$ is small, it is not significant and could be due to noise. If $w_{j,l}$ is large, it is significant:

$$\begin{array}{ll} \text{if } \mid w_{j,l} \mid \geq k\sigma_j & \text{then } w_{j,l} \text{ is significant} \\ \text{if } \mid w_{j,l} \mid < k\sigma_j & \text{then } w_{j,l} \text{ is not significant} \end{array} \tag{4.6}$$

So we need to estimate, in the case of Gaussian noise models, the noise standard deviation at each scale. These standard deviations can be determined analytically, but the calculations can become complicated.

The appropriate value of σ_j in the succession of wavelet planes is assessed from the standard deviation of the noise σ_N in the original data D, and from study of the noise in the wavelet space. This study consists of simulating a data set containing Gaussian noise with a standard deviation equal to 1, and taking the wavelet transform of this data set. Then we compute the standard deviation σ_j^e at each scale. We get a curve σ_j^e as a function of j, giving the behavior of the noise in the wavelet space (Note that if we had used an orthogonal wavelet transform, this curve would be linear). Due to the properties of the wavelet (resp. curvelet) transform, we have $\sigma_j = \sigma_N \sigma_j^e$. The noise standard deviation at scale j of the data is equal to the noise standard deviation σ_N multiplied by the noise standard deviation at scale j of the simulated data.

4.2.3 False Discovery Rate

The individual binary hypothesis testing discussed above can face serious problems because of the large numbers of coefficients (hypotheses) being tested simultaneously. In other words, if the type 1 error is controlled at an individual level, the chance of keeping erroneously a coefficient is extremely high as the number of false detections increases with the number of hypotheses being tested simultaneously.

Therefore, if one desires to have control over global statistical error rates, multiple hypothesis testing should be corrected for. For example, the Bonferroni correction consists of comparing the p-value of each coefficient to α/(total number of tested coefficients at subband (j, ℓ)). The Bonferroni correction controls the probability of erroneously rejecting even one of the true null hypotheses, i.e., the Family-Wise Error Rate (FWER). It is however too over-conservative entailing a dissipation of detection power. To mitigate this limitation, it is better to use the (Benjamini and Hochberg 1995) procedure to control the False Discovery Rate (FDR), i.e., the average fraction of false detections over the total number of detections. The control of FDR has the following advantages over that of FWER: (i) it usually has greater detection power; and (ii) it can easily handle correlated data (Benjamini and Yekutieli 2001). The latter point allows FDR control when the noise is not independent (e.g. when ε is additive white Gaussian and the transform is redundant). Minimaxity of FDR has also been studied in various settings: see (Abramovich et al. 2006) and (Donoho and Jin 2006).

4.2.4 Automatic Estimation of Gaussian Noise

k-sigma clipping

The Gaussian noise σ_N can be estimated automatically in a data set D. This estimation is particularly important, because all the noise standard deviations σ_j in the scales j are derived from σ_N. Thus an error associated with σ_N will introduce an error on all σ_j. Noise is therefore more usefully estimated in the high frequencies, where it dominates the signal. The resulting method consists first of filtering the data D with an average filter or the median filter and subtracting from D the filtered signal F: $S = D - F$. In our case, we replace S by the first scale of the wavelet transform ($S = w_1$), which is more convenient from the computation time point of view. The histogram of S shows a Gaussian peak around 0. A k-sigma clipping is then used to reject pixels where the signal is significantly large. We denote $S^{(1)}$ the subset of S which contains only the pixels such that $\mid S_l \mid < k\sigma_S$, where σ_S is the standard deviation of S, and k is a constant generally chosen equal to 3. By iterating, we obtain the subset $S^{(n+1)}$ verifying $\mid S_l^{(n)} \mid < k\sigma_{S^{(n)}}$, where $\sigma_{S^{(n)}}$ is the noise standard deviation of $S^{(n)}$. Robust estimation of the noise σ_1 in w_1 (as $S = w_1$) is now obtained by calculation of the standard deviation of $S^{(n)}$ ($\sigma_1 = \sigma_{S^{(n)}}$). In practice, three iterations are enough, and accuracy is generally better than 5%. σ_N is finally calculated by:

$$\sigma_N = \frac{\sigma_1}{\sigma_1^e} = \frac{\sigma_{S^{(n)}}}{\sigma_1^e} \tag{4.7}$$

4.2.5 Correlated Noise

In this case, the data can be treated as for the Gaussian case, but the noise standard deviation σ_j at scale j is calculated independently at each scale. Two methods can be used:

1. σ_j can be derived from a k-sigma clipping method applied at scale j.
2. The median absolute deviation, MAD, can be used as an estimator of the noise standard deviation:

$$\sigma_j = \text{median}(\mid w_j \mid)/0.6745 \tag{4.8}$$

4.3 Denoising

Many filtering methods have been proposed in the last ten years. *Hard thresholding* consists of setting to 0 all wavelet coefficients which have an absolute value lower than a threshold T_j (non-significant wavelet coefficient):

$$\tilde{w}_{j,k} = \begin{cases} w_{j,k} & \text{if } \mid w_{j,k} \mid \geq T_j \\ 0 & \text{otherwise} \end{cases}$$

where $w_{j,k}$ is a wavelet coefficient at scale j and at spatial position k.

Soft thresholding consists of replacing each wavelet coefficient by the value $\tilde{w}$ where

$$\tilde{w}_{j,k} = \begin{cases} sgn(w_{j,k})(\mid w_{j,k} \mid - T_j) & \text{if } \mid w_{j,k} \mid \geq T_j \\ 0 & \text{otherwise} \end{cases}$$

This operation is generally written as:

$$\tilde{w}_{j,k} = \text{soft}(w_{j,k}) = sgn(w_{j,k})(\mid w_{j,k} \mid - T_j)_+ \tag{4.9}$$

where $(x)_+ = MAX(0, x)$.

When the discrete orthogonal wavelet transform is used instead of the "à trous" algorithm, it is interesting to note that the hard and soft thresholded estimators are solutions of the following minimization problems:

$$\tilde{w} = \arg_w \min \frac{1}{2} \parallel D - \mathcal{W}^{-1} w \parallel_{l^2}^2 + \lambda \parallel w \parallel_{l^0}^2 \qquad \textbf{hard threshold}$$

$$\tilde{w} = \arg_w \min \frac{1}{2} \parallel D - \mathcal{W}^{-1} w \parallel_{l^2}^2 + \lambda \parallel w \parallel_{l^1}^2 \qquad \textbf{soft threshold}$$

where D is the input data, $\mathcal{W}$ the wavelet transform operator, and l^0 indicates the limit of l^δ when $\delta \to 0$. This counts in fact the number of non-zero elements in the sequence.

As described before, in the case of Gaussian noise, $T_j = K\sigma_j$, where j is the scale of the wavelet coefficient, σ_j is the noise standard deviation at the scale j, and K is a constant generally chosen equal to 3.

Other threshold methods have been proposed, like the *universal threshold* (Donoho and Johnstone 1994; Donoho 1993), or the SURE (Stein Unbiased Risk Estimate) method (Coifman and Donoho 1995), but they generally do not yield as good results as the hard thresholding method based on the significant coefficients. For astronomical data, soft thresholding should never be used because it leads to a photometry loss associated with all objects, which can easily be verified by looking at the residual map (i.e. data − filtered data). Concerning the threshold level, the universal threshold corresponds to a minimum risk. The larger the number of pixels, the larger is the risk, and it is normal that the threshold T depends on the number of pixels ($T = \sigma_j\sqrt{2\log n}$, n being the number of pixels). The $K\sigma$ threshold corresponds to a false detection probability, the probability to detect a coefficient as significant when it is due to the noise. The 3σ value corresponds to 0.27 % false detection.

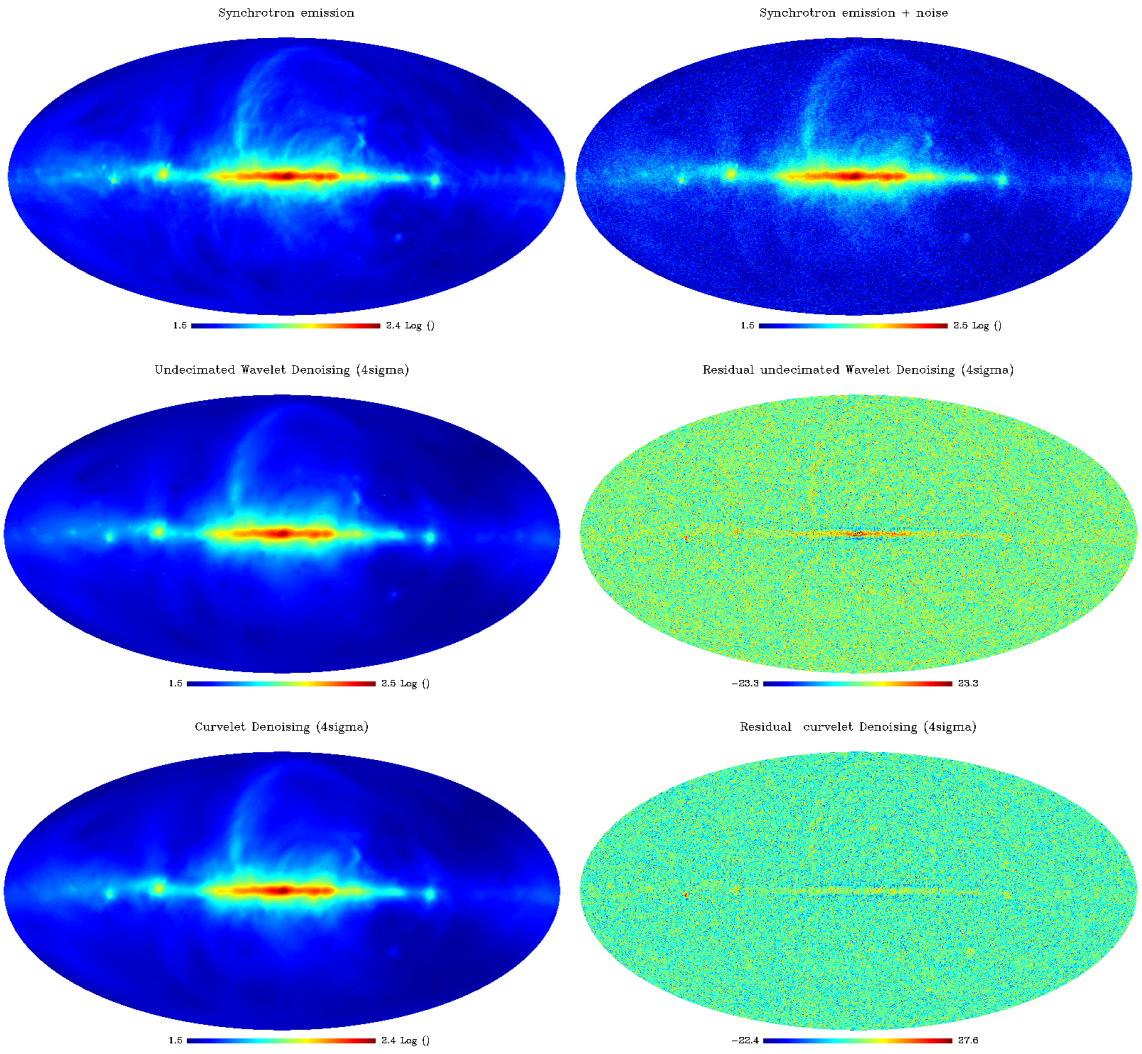


Figure 4.1: Denoising. Upper left and right: simulated synchrotron image and same image with additive Gaussian noise (i.e. simulated data). Middle: undecimated wavelet filtering and residual. Bottom: pyramidal curvelet filtering and residual.

Fig. 4.1 describes the setting and the results of a simulated denoising experiment: upper left, the original simulated map of the astrophysical synchrotron emission; upper right, the same image plus additive Gaussian noise ($\sigma = 5$). Since the synchrotron image has a standard deviation (after renormalization) equal to 16.26, the SNR is around 3.25. The middle panels in this figure show the UWTS denoised image and the residuals. The bottom panels show the pyramidal curvelet transform filtered image and the residuals. On such data, exhibiting very anisotropic features, the curvelets produce better results than wavelets.

Although the results obtained by simply thresholding the curvelet expansion are encouraging, there is of course ample room for further improvement. A quick inspection of the residual images for both the wavelet and curvelet transforms shown in Figure 4.1 reveals the existence of very different features. For instance, wavelets do not restore long features with high fidelity while curvelets are seriously challenged by isotropic or small features. Each transform has its own area of expertise and this complementarity is of great potential. The Combined Filtering Method (CFM) (Starck et al. 2001) allows us to benefit from the advantages of both transforms. This iterative method detects the significant coefficients in both the wavelet domain and the curvelet domain and guarantees that the reconstructed map will take into account any pattern which is detected as significant by either of the transforms.

4.4 The Combined Filtering Method on the Sphere

In general, suppose that we are given K linear transforms $T_1, \ldots, T_K$ and let α_k be the coefficient sequence of an object x after applying the transform T_k, i.e. $\alpha_k = T_k x$. We will assume that for each transform T_k we have available a reconstruction rule that we will denote by T_k^{-1} although this is clearly an abuse of notation. Finally, T will denote the block diagonal matrix with the T_k's as building blocks and α the amalgamation of the α_k's.

A hard thresholding rule associated with the transform T_k synthesizes an estimate $\tilde{s}_k$ via the formula

$$\tilde{s}_k = T_k^{-1} \delta(\alpha_k) \tag{4.10}$$

where δ is a rule that sets to zero all the coordinates of α_k whose absolute value falls below a given sequence of thresholds (such coordinates are said to be non-significant).

Given data y of the form $y = s + \sigma z$, where s is the image we wish to recover and z is standard white noise, we propose solving the following optimization problem (Starck et al. 2001):

$$\min \|T\tilde{s}\|_{\ell_1}, \quad \text{subject to} \quad s \in C, \tag{4.11}$$

where C is the set of vectors $\tilde{s}$ which obey the linear constraints

$$\begin{cases} \tilde{s} \geq 0, \\ |T\tilde{s} - Ty| \leq e; \end{cases} \tag{4.12}$$

here, the second inequality constraint only concerns the set of significant coefficients, i.e. those indices μ such that $\alpha_\mu = (Ty)_\mu$ exceeds (in absolute value) a threshold t_μ. Given a

vector of tolerance (e_μ), we seek a solution whose coefficients $(T\tilde{s})_\mu$ are within e_μ of the noisy empirical α_μ's. Think of α_μ as being given by

$$y = \langle y, \varphi_\mu \rangle,$$

so that α_μ is normally distributed with mean $\langle f, \varphi_\mu \rangle$ and variance $\sigma_\mu^2 = \sigma^2 \|\varphi_\mu\|_2^2$. In practice, the threshold values range typically between three and four times the noise level σ_μ and in our experiments we will put $e_\mu = \sigma_\mu/2$. In short, our constraints guarantee that the reconstruction will take into account any pattern which is detected as significant by any of the K transforms.

4.4.1 The Minimization Method

We propose solving (4.11) using the method of hybrid steepest descent (HSD) (Yamada 2001). HSD consists of building the sequence

$$s^{n+1} = P(s^n) - \lambda_{n+1} \nabla_J (P(s^n)); \tag{4.13}$$

Here, P is the ℓ_2 projection operator onto the feasible set C, ∇_J is the gradient of equation (4.11), and $(\lambda_n)_{n\geq 1}$ is a sequence obeying $(\lambda_n)_{n\geq 1} \in [0, 1]$ and $\lim_{n\to+\infty} \lambda_n = 0$.

Algorothm 7 gives the resulting combined filtering method.

Method	Error standard deviation
Noisy map	5 8
Wavelet	1.25
Curvelet	1.07
CFA	0.86

Table 4.1: Error standard deviations after denoising the synchrotron noisy map (additive white Gaussian noise, $\sigma = 5$) by the wavelet, the curvelet and the combined denoising algorithm.

4.4.2 Experiments

Figure 4.2 shows the CFM denoised image and its residual. Figure 4.3 shows one face (face 6) of the following Healpix images: upper left, original image; upper right, noisy image; middle left, restored image after denoising by the combined transform; middle right, the residual; bottom left and right, the residual using respectively the curvelet and the wavelet denoising method. The results are reported in Table 4.1. The residual is much better when the combined filtering is applied, and no feature can be detected any more by eye in the residual. This was not the case for either the wavelet and the curvelet filtering.

Algorithm 7 The combined filtering on the Sphere.

Task: Compute the combined filtering on the sphere of a discrete s.
Parameters: Data samples s, number of iterations N_i and number of selected transforms K.
Initialization:

- $L_{\max} = 1$
- $\delta_\lambda = \frac{L_{\max}}{N_i}$
- Estimate the noise standard deviation σ, and set $e_k = \frac{\sigma}{2}$.
- **for** $k = 1, \ldots, K$ **do** calculate the transform: $\alpha_k^{(s)} = T_k s$.
- Set $\lambda = L_{\max}$, $n = 0$, and $\tilde{s}^n$ to 0.

while $\lambda >= 0$ **do**

1. $u = \tilde{s}^n$.
2. **for** $k = 1, \ldots, K$ **do**
   - Calculate the transform $\alpha_k = T_k u$.
   - **foreach** $\alpha_{k,l}$ **do**
     - Calculate the residual $r_{k,l} = \alpha_{k,l}^{(s)} - \alpha_{k,l}$
     - **if** $\alpha_{k,l}^{(s)}$ is significant AND $\mid r_{k,l} \mid > e_{k,l}$ **then** $\alpha_{k,l} = \alpha_{k,l}^{(s)}$
     - $\alpha_{k,l} = sgn(\alpha_{k,l})(\mid \alpha_{k,l} \mid -\lambda)_+$.
   - $u = T_k^{-1} \alpha_k$
3. Threshold negative values in u and $\tilde{s}^{n+1} = u$.
4. $n = n + 1$, $\lambda = \lambda - \delta_\lambda$

Output: $\tilde{s}^m$ with $m = N_i$, filtered map.

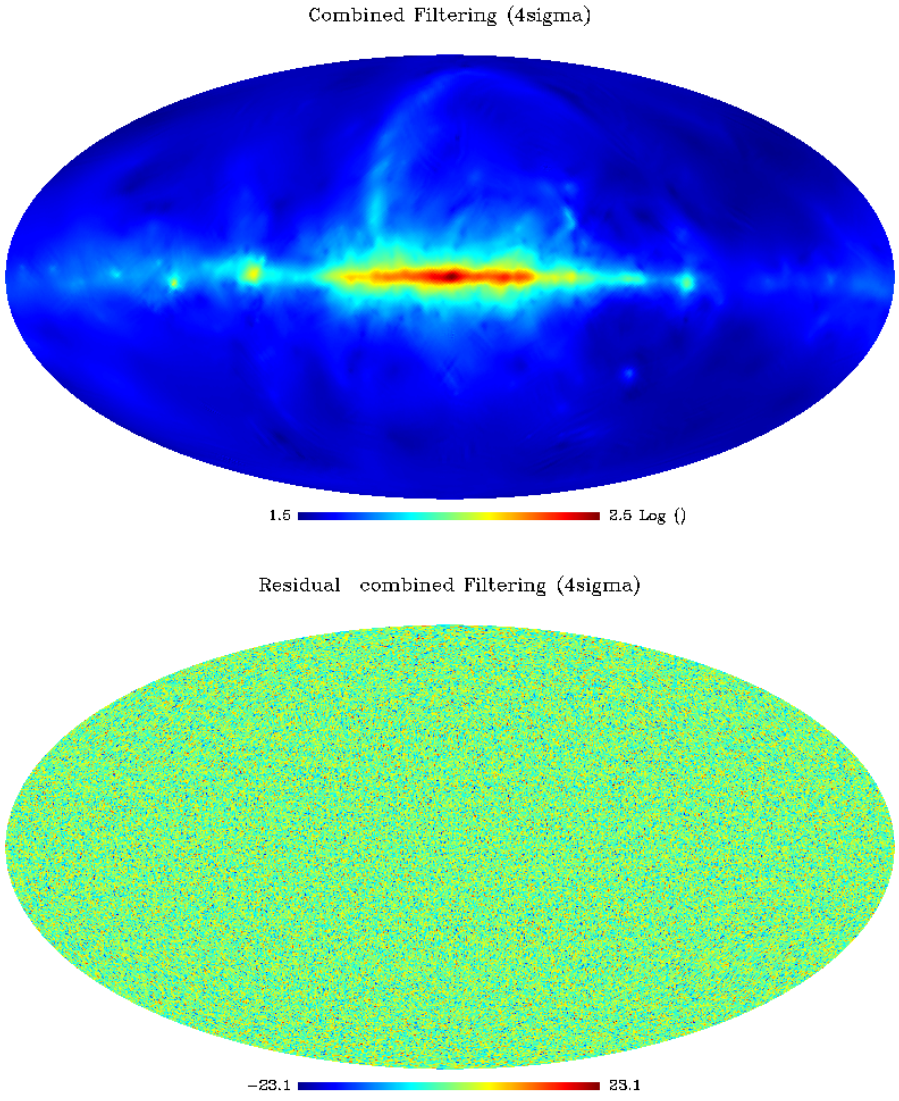


Figure 4.2: Combined denoising (using both wavelets and curvelets) and residuals.

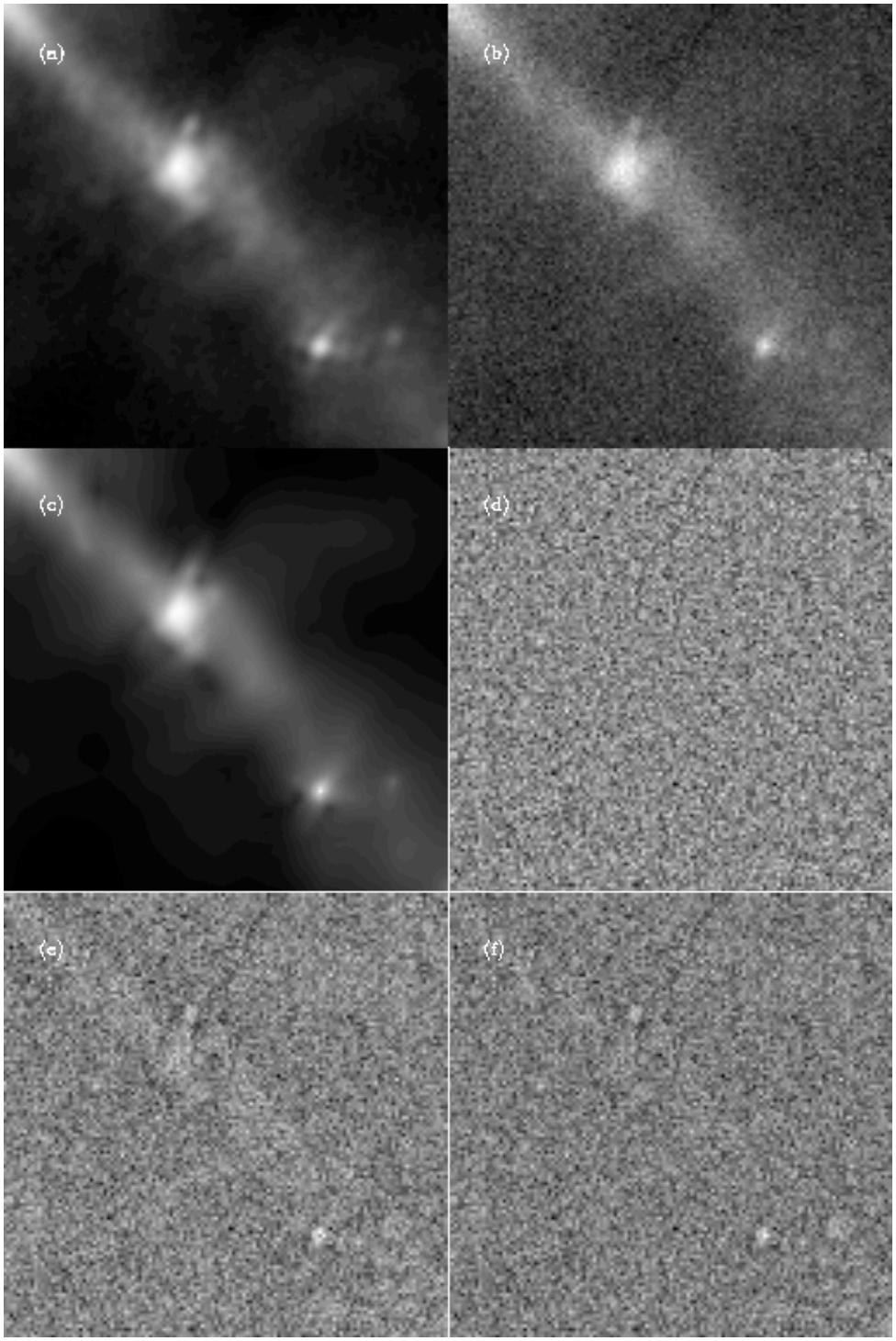


Figure 4.3: Combined Filtering Method, face 6 in the Healpix representation of the image shown in figure 4.2. From top to bottom and left to right, respectively the a) original image face, b) the noisy image, c) the combined filtered image, d) the combined filtering residual, e) the wavelet filtering residual and f) the curvelet filtering residual.

Chapter 5

Sparse Component Analysis, data restoration and inpainting on the sphere

5.1 Morphological Component Analysis on the sphere

A usual task in processing signals, images as well as spherical data maps, is to decompose the data into its elementary building blocks. This can be formulated as an inverse problem where the data is assumed to have been generated according to the following model :

$$y = \sum_i \alpha_i \phi_i + \eta \tag{5.1}$$

that is a linear combination of relevant waveforms $\phi_i \in \mathbb{R}^n$ with weights α_i. Here η represents possible contamination by additive, typically Gaussian white noise. Given data $y \in \mathbb{R}^n$, one then wants to recover the underlying structures that is to say estimate a set of waveforms ϕ_i that build the data and their corresponding weights $\tilde{\alpha}_i$. The solution to this estimation problem will depend heavily on the available prior information. Of interest here is the case where one is given a priori a set a waveforms from which to select a good subset. This set may be a basis, a frame or several bases or frames grouped into a large redundant dictionary.

Possible dictionaries in 1D and 2D include Fourier and related bases, wavelet bases, as well as other more recent multiscale systems such as the ridgelet (Candès and Donoho 1999b) and curvelet frames (Donoho and Duncan 2000; Starck et al. 2002a), etc. Depending on the morphology of the data, each of these dictionaries will have different performance characteristics in a non-linear approximation scheme. For instance, sparse approximations of piecewise smooth signals or images with point singularities are easily obtained using wavelets. However these are no longer optimal in the case of piecewise smooth images with singularities along smooth curves or edges. Such images are more efficiently approximated using curvelets which are highly anisotropic and thus exhibit high directional selectivity. Digital implementations of both ridgelet and curvelet transforms and their application to image denoising are described in (Starck et al. 2002a).

Available transforms in the spherical topology include the spherical harmonics and several wavelet transforms. Software packages such as Healpix[1] (Górski et al. 2005) or Glesp (Doroshkevich et al. 2005) provide approximate digital spherical harmonic transform routines based on their specific pixelization schemes. Schröder and Sweldens (Schröder and Sweldens 1995) have developed an orthogonal wavelet transform on the sphere based on the Haar wavelet function which then suffers from the poor frequency domain localization properties of the primitive Haar function and from the problems inherent in orthogonal decomposition (e.g. lack of translation invariance). A few papers describe continuous wavelet transforms on the sphere (Antoine and Vandergheynst 1999; Cayón et al. 2001a; Holschneider 1996; Wiaux et al. 2005; Bogdanova et al. 2005) which have been extended to directional wavelet transforms (Wiaux et al. 2006; McEwen et al. 2005). Although useful for data analysis, these continuous transforms lack an inverse transform and hence are clearly not suitable for restoration or synthesis purposes.

In their pioneering work, Freeden and Maier (Freeden and Maier 2002; Freeden et al. 2003) gave a wavelet transform and reconstruction scheme on the sphere which is based on the spherical harmonic transform. Following this idea, Starck et al. (Starck et al. 2006) have proposed a new invertible isotropic undecimated wavelet transform (UWT) on the sphere which preserves the same desirable properties as the standard isotropic UWT for flat 2D maps (Starck et al. 1998): the reconstruction is simple and immediate since it is just the addition of all the wavelet bands with the coarsest scale. Based on this new decomposition, other multiscale transforms such as the pyramidal wavelet transform, the ridgelet transform and the curvelet transform have been successfully constructed on the sphere (Starck et al. 2006). Each of these decompositions on the sphere will sparsely represent parts of the image based on their morphological properties. Wavelets will easily detect more or less isotropic localized structures, while curvelets are better suited for efficiently detecting highly anisotropic objects.

A data set y has an exact representation over any complete basis of the data space, or several such exact representations in the case of redundant overcomplete dictionaries. However, these representations are not equally interesting in terms of data modeling or feature detection. In fact, a strong a priori is to favor representations of y that use only a small number of waveforms leading to a more concise and possibly more interpretable representation of the data. In fact, building sparse representations or approximations is the (he)art of structured data processing: the design of good detection, denoising, restoration and compression algorithms relies on the availability of good dictionaries and good selection algorithms. Indeed, selecting the smallest subset of waveforms from a large dictionary, that will linearly combine to reproduce the salient features of a given signal or image, is a hard combinatorial problem. Several pursuit algorithms have been proposed that can help build very sparse decompositions such as the greedy Matching Pursuit (MP) (Mallat and Zhang 1993) algorithm which refines the signal approximation by picking at each iteration the one waveform which best correlates with the current approximation error. Basis Pursuit (BP) (Chen et al. 1999) is a global procedure which seeks an approximation $\tilde{y}$ to y by solving the linear programming problem:

$$\min_{\alpha} \ \|\alpha\|_{\ell_1} \text{ subject to } y = \Phi\alpha. \tag{5.2}$$

where the ℓ_1 norm measures sparsity in place of the ℓ_0 counting norm.

[1] http://www.eso.org/science/healpix

In the presence of noise, a noise-aware variant of BP, known as BPDN (for BP denoising), can be stated as a convex quadratic programming problem and solved using the Interior Point method (Chen et al. 1999). The BPDN problem can also be written in the augmented Lagrangian form:

$$\min_{\alpha} \ \|y - \Phi\alpha\|_{\ell_2}^2 + \lambda \cdot \|\alpha\|_{\ell_1} \tag{5.3}$$

Among all possible solutions, the chosen one has the minimum ℓ_1 norm. This choice of ℓ_1 norm is very important. An ℓ_2 norm, as used in the method of frames (Daubechies 1988), does not favor sparsity (Chen et al. 1999). A number of recent results prove that these algorithms will recover the unique maximally sparse decomposition provided this solution is sparse enough and the dictionnary is sufficiently incoherent (Donoho and Elad 2003; Elad and Bruckstein 2002; Gribonval and Nielsen 2003; Donoho et al. 2006a; Fuchs 2005). Nevertheless, in problems involving large data sets (e.g. images, spherical maps), BP or MP synthesis algorithms are computationally prohibitive. Morphological Component Analysis (MCA) is a recent faster alternative described in (Starck et al. 2004b) that constructs a sparse representation of a signal or an image assuming that it is a combination of morphologically distinct features which are sparsely represented in different dictionaries associated with fast transform algorithms. For instance, images commonly combine contours and textures: the former are well accounted for using curvelets, while the latter may be well represented using local cosine functions. In searching for a sparse decomposition of a signal or image y, it is assumed that y is a sum of K components $(s_k)_{1,\ldots,K}$, where each can be described as $s_k = \mathbf{\Phi}_k\alpha_k$ with a possibly over-complete dictionary $\mathbf{\Phi}_k$ and a sparse vector of coefficients α_k. It is further assumed that for any given component the sparsest decomposition over the proper dictionary yields a highly sparse description, while its decomposition over the other dictionaries, $\mathbf{\Phi}_{k'\neq k}$, is non sparse. Thus, the different $\mathbf{\Phi}_k$ can be seen as discriminating between the different components of the initial signal. MCA achieves its sparse decomposition relying on an iterative thresholding algorithm with a successively decreasing threshold (Bobin et al. 2007b) thus refining the current approximation by including finer structures alternatingly in the different morphological components. Based on MCA, it has also been shown that we can derive a very efficient inpainting method (Elad et al. 2005).

5.2 MCA on the Sphere

5.2.1 Principle and algorithm

For a given spherical map y modeled as a linear combination of K spherical maps s_k, $y = \sum_{k=1}^{K} s_k$, having different morphologies, MCA assumes that a dictionary of bases $\{\mathbf{\Phi}_1, \cdots, \mathbf{\Phi}_K\}$ exists such that, for each k, s_k is sparse in $\mathbf{\Phi}_k$ while its representation in the other $\mathbf{\Phi}_{k'}$ ($k' \neq k$) is not sparse : $\forall k' \neq k, ||\mathbf{\Phi}_k^T s_k||_0 < ||\mathbf{\Phi}_{k'}^T s_k||_0$, where $||x||_0$ denotes the ℓ_0 pseudo-norm of the vector x The problem is to separate the mixture y into its constitutive morphological components $(s_k)_{k=1,\cdots,K}$ relying on the discriminating power of the different dictionaries $\mathbf{\Phi}_k$. Ideally, the α_k are the solutions to :

$$\min_{\alpha_1,\ldots,\ \alpha_K} \ \sum_{k=1}^{K} \|\alpha_k\|_0 \quad \text{subject to} \quad y = \sum_{k=1}^{K} \mathbf{\Phi}_k\alpha_k \tag{5.4}$$

While sensible from the point of view of the desired solution, the problem formulated in Equation (5.4) is non-convex and combinatorial by nature. Its complexity grows exponentially with the number of columns in the overall dictionary (NP-hard problem). Motivated by recent equivalence results e.g. in (Donoho and Elad 2003), the MCA algorithm seeks a solution to the following minimization problem:

$$\min_{s_1,\ldots,s_K} \lambda \sum_{k=1}^{K} \|\alpha_k\|_1 + \left\| y - \sum_{k=1}^{K} s_k \right\|_2^2 \quad \text{with} \quad s_k = \mathbf{\Phi}_k \alpha_k \tag{5.5}$$

where an ℓ_1 sparsity measure is substituted to the ℓ_0 counting norm following a prescription of the Basis Pursuit algorithm (Chen et al. 1999). In the above, the equality constraint was relaxed and again $s_k = \mathbf{\Phi}_k \alpha_k$. In the case where each $\mathbf{\Phi}_k$ is an orthonormal basis, a block-coordinate solution to the above problem is given by the following set of coupled equations:

$$\forall k, s_k = r_k - \frac{\lambda_k}{2} \mathbf{\Phi}_k \text{sign}(\mathbf{\Phi}_k^T s_k) \text{ with } r_k = s - \sum_{k' \neq k} s_{k'} \tag{5.6}$$

This can be solved efficiently using the iterative *Block-Coordinate Relaxation Method* (Bruce et al. 1998) in conjunction with, at a given k, a soft-thresholding of the decomposition of r_k over $\mathbf{\Phi_k}$. However, when non-unitary or redundant transforms are used, the above is no longer strictly valid. Nevertheless, simple shrinkage still gives satisfactory results as explained in (Elad 2006). Finally, denoting by $\mathbf{T}_k$ and $\mathbf{R}_k$ the forward and inverse transforms associated with the redundant dictionary $\mathbf{\Phi_k}$, MCA seeks a solution to problem (5.5) with the algorithm 8:

5.2.2 Thresholding strategy

The operator δ in the above algorithm is a soft thresholding operator as a result of the use of an ℓ_1 sparsity measure in approximation to the ideal ℓ_0 norm. In practice, hard thresholding leads generally to better results (Starck et al. 2004b). The final threshold should vanish in the noiseless case or it may be set to a multiple of the noise standard deviation in the presence of noise as in common detection or denoising methods. The way the threshold is decreased along the iterations of the proposed iterative thresholding scheme is paramount in terms of performance of the MCA separation mechanism. The original algorithm (Starck et al. 2004b) used a linear strategy :

$$\lambda^{(t)} = \lambda^{(0)} - (t-1) \frac{\lambda^{(0)} - \lambda_{\min}}{I_{\max} - 1} \tag{5.7}$$

where $\lambda^{(0)}$ is the initial threshold, and $I_{\max}$ is the number of iterations. The first threshold can be set automatically to a large enough value such as the maximum of all coefficients $\lambda^{(0)} = \max_k \|\mathbf{T}_k y\|_\infty$. But there is no way to estimate the minimum number of iterations yielding a successful separation. Too small a number of iterations leads to bad separation while too large a number is computationally costly. Further, experiments have clearly shown that the optimal number of iterations depends on the data. We recently focused on devising some new data adaptive thresholding strategies to speed up the MCA decomposition preserving the quality of the component separation. Hereafter we describe two

Algorithm 8 The Morphological Component Analysis algorithm .
Task: Compute the MCA of data y.
Parameters: Data samples y, number of selected transforms K, transforms with $\mathbf{T}_k$ and $\mathbf{R}_k$ the forward and inverse transforms operators.
Initialization:

- Set the number of iterations $I_{\max}$
- Set the initial thresholds $\left(\lambda_k^{(0)}\right)_k$
- Set the final thresholds $\lambda_{\min}$, it can depend on the noise standard deviation

while $\lambda_k^{(t)} > \lambda_{\min}$ **do**

1. **for** $k = 1, \ldots, K$ **do**
   - Compute the residual term $r_k^{(t)}$ assuming the current estimates $\tilde{s}_{k' \neq k}^{(t-1)}$ of $s_{k' \neq k}$, are fixed: $r_k^{(t)} = y - \sum_{k' \neq k} \tilde{s}_{k'}^{(t-1)}$
   - Estimate the current coefficients of $\tilde{s}_k^{(t)}$ by thresholding with threshold $\lambda_k^{(t)}$: $\tilde{\alpha}_k^{(t)} = \delta_{\lambda_k^{(t)}}\left(\mathbf{T}_k r_k^{(t)}\right)$
   - Get the new estimate of s_k by reconstructing from the selected coeffcients $\tilde{\alpha}_k^{(t)}$: $\tilde{s}_k^{(t)} = \mathbf{R}_k \tilde{\alpha}_k^{(t)}$
2. Decrease the thresholds λ_k following a given strategy.

Output: $(\tilde{s}_k^{(m)})$ $k = 1, \ldots, K$ with $m = I_{\max}$: separated components.

promising strategies, namely MAD and MOM, in the case where $K = 2$; generalizing to $K \geq 2$ is straightforward.

MAD Consider a map y such that $y = s_1 + s_2 = \mathbf{\Phi}_1 \alpha_1 + \mathbf{\Phi}_2 \alpha_2$ where s_1 and s_2 have similar ℓ_2 norm and $\alpha_{k=1,2} = \mathbf{\Phi}_{k=1,2}^T s_{k=1,2}$ are sparse. When both $\mathbf{\Phi}_{k=1,2}$ are orthonormal bases, decomposing y in $\mathbf{\Phi}_1$ leads to $y\mathbf{\Phi}_1^T = \alpha_1 + \mathbf{\Phi}_1^T \mathbf{\Phi}_2 \alpha_2$. Provided the mutual coherence (Elad and Bruckstein 2002; Gribonval and Nielsen 2003; Donoho and Elad 2003) of $\mathbf{\Phi}_1$ and $\mathbf{\Phi}_2$ is low, y_2 has no particular structure in $\mathbf{\Phi}_1$ and hence it is tempting to model $\mathbf{\Phi}_1^T s_2$ as a Gaussian noise. Its standard deviation can be estimated using a robust estimator such as the Median Absolute Deviation (MAD) (Donoho and Johnstone 1994). It follows that estimating the significant entries $\tilde{\alpha}_1$ in α_1 is a denoising problem readily solved by thresholding $\mathbf{\Phi}_1^T y$ with a threshold $k\sigma$ (typically k is in the range 3 to 4). The next step is to project the residual $a\ y - \tilde{s_1} = y - \mathbf{\Phi}_1 \tilde{\alpha_1}$ on $\mathbf{\Phi}_2$ and so on. Clearly, the variance of the residual decreases along iterations and so this provides a simple strategy to adaptively control the threshold in the MCA algorithm. In practice, this strategy remains fruitful in the case of redundant dictionaries. Donoho and al. in (Donoho et al. 2006b) have recently focused on an iterative thresholding scheme applied

to solving under-determined linear sparse problems in which they use a similar rule to manage their decreasing threshold.

MOM Let $\tilde{s}_1^{(t)}$ and $\tilde{s}_2^{(t)}$ denote the current estimates of components s_1 and s_2 at the t^{th} iteration of the MCA decomposition of y. The current residual is $r^{(t)} = y - \tilde{s}_1^{(t)} - \tilde{s}_2^{(t)}$. In the strategy coined MOM as in "Mean of Max", the value of the threshold at iteration t is given by :

$$\lambda^{(t)} = \frac{1}{2}\left[||\mathbf{\Phi}_1^T\left(y - \tilde{s}_1^{(t-1)} - \tilde{s}_2^{(t-1)}\right)||_\infty + ||\mathbf{\Phi}_2^T\left(y - \tilde{s}_1^{(t-1)} - \tilde{s}_2^{(t-1)}\right)||_\infty\right] \tag{5.8}$$

which is easily computed at each step of the iterative process. When one considers more than two dictionaries, one should take the mean of the two largest decomposition coefficients of the full residual over two distinct dictionaries. The intuition underlying this strategy is that the next significant coefficients to be selected should be attached to the dictionary in which the projection of the full residual has coefficients of largest amplitudes. Assuming the coefficients selected at iteration t are in $\mathbf{\Phi}_1$, it can be shown, under some conditions on the sparsity of the components and the mutual coherence of the dictionary (Bobin et al. 2007b), that the proposed strategy fixes the threshold so that :

$$||\mathbf{\Phi}_1^T\mathbf{\Phi}_2\bar{\alpha}_2^{(t-1)}||_\infty < \lambda_1^{(t)} < ||\bar{\alpha}_1^{(t-1)}||_\infty, \bar{\alpha}_{k=1,2}^{(t-1)} = \alpha_{k=1,2} - \tilde{\alpha}_{k=1,2}^{(t-1)} \tag{5.9}$$

hence avoiding false detections (upper bound) and ensuring that at least one coefficient is selected (lower bound). This thresholding strategy can easily be made more or less conservative depending on the desired decomposition speed. With these new thresholding strategies, MCA is a fast and robust algorithm to achieve sparse decompositions in redundant dictionaries and a practical alternative to other well-known sparse decomposition algorithms (Bobin et al. 2007b).

Example

The spherical maps shown on figure 5.1 illustrate a simple numerical experiment. We applied the proposed Morphological Component Analysis on the Sphere to synthetic data resulting from the linear mixture of components respectively sparse in the spherical harmonics and the isotropic wavelet representations. The method was able to separate the data back into its original constituents. A more involved application is described in the next section.

5.2.3 Application in Physics

In Inertial Confinement Fusion (ICF) a spherical shell is irradiated by laser energy directly or after the laser energy has been converted to soft X-rays (Atzeni and ter Vehn 2004). Either way, the aim is to implode the capsule which contains a shell of nuclear fusion fuel (deuterium and tritium) ready to ignite if, after it has been imploded, its density is high enough and a hot spot in its center becomes hot enough to cause a propagating nuclear burn wave to travel through the rest of the fuel. This ultimate energy source will not work if during the implosion hydrodynamic instabilities develop which can break

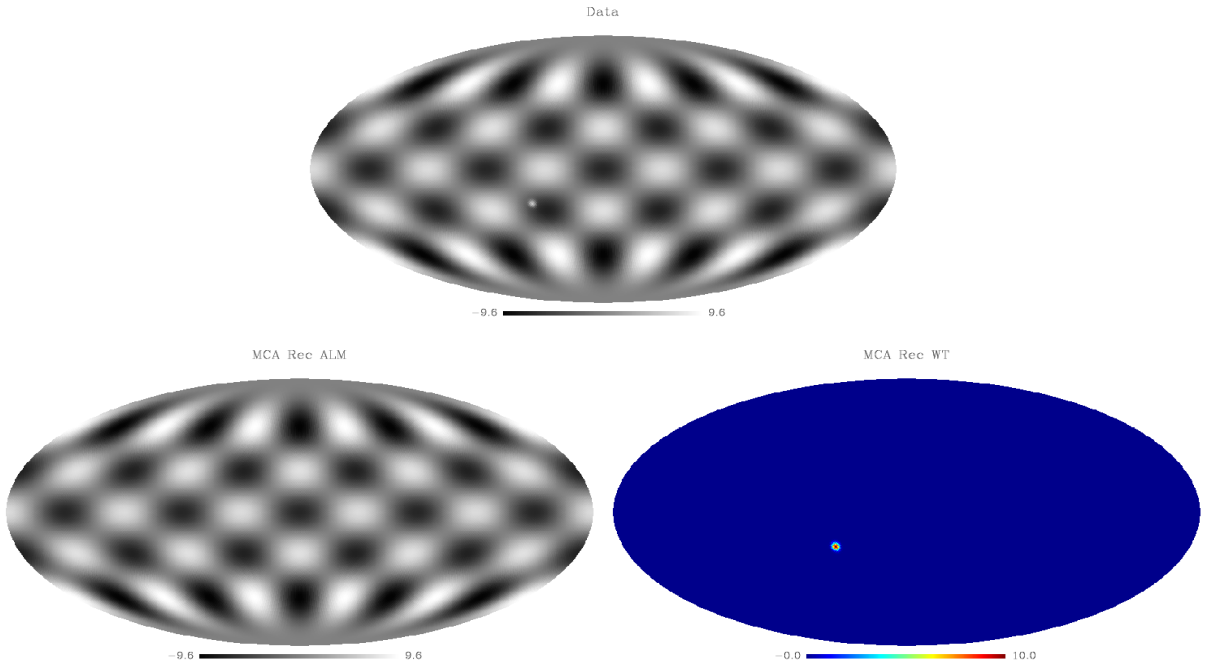


Figure 5.1: Simple toy experiment with MCA on the sphere. The top map shows a linear combination of a spherical harmonic function and a localized Gaussian-like function on the sphere. The bottom maps show the resulting separated components that were obtained using the proposed Morphological Component Analysis on the sphere.

apart the shell before it assembles at the center and a hot spot forms (Lindl 1997). Hydrodynamic instabilities such as Rayleigh-Taylor occur due to nonuniformities in the laser spatial profile or imperfections in the composition of multiple surfaces which make up the layers of thin material that surround the nuclear fuel. Very small amplitude imperfections initially can result in the ultimate failure of the target due to the large compression ratios involved in ICF. It is therefore extremely important to characterize the inner and outer surfaces of ICF shell targets so as to know whether they are worthy of consideration for ICF implosions. One day in a reactor setting tens of thousands of targets will have to be imploded daily so that checking each one is totally out of the question. Instead, very good target fabrication quality control processes have to be adopted so that confidence levels in proper performance will be high. A major step along this path to fusion energy then is to understand why imperfections occur and to correct the systematic elements and control the harm done by random sources. Fine structures on the surfaces of spherical shells can be measured on the nanometer scale, among others, by atomic force microscopy or phase shifting spherical diffractive optical interferometry. An example of such measurements is shown on figure 5.2. As can be seen from the figure, there appears to be a superposition of global scale variations, isolated bumps and scratches as well as artifacts which look like interference patterns on intermediate scales of localization. The latter must be isolated and eliminated from consideration when deciding the readiness of the target for implosion. We have achieved the morphological feature separation by first doing an isotropic wavelet transform on the spherical data and subtracting the coarsest scale information. MCA on the sphere was used on the rest of the image using the undecimated wavelet and

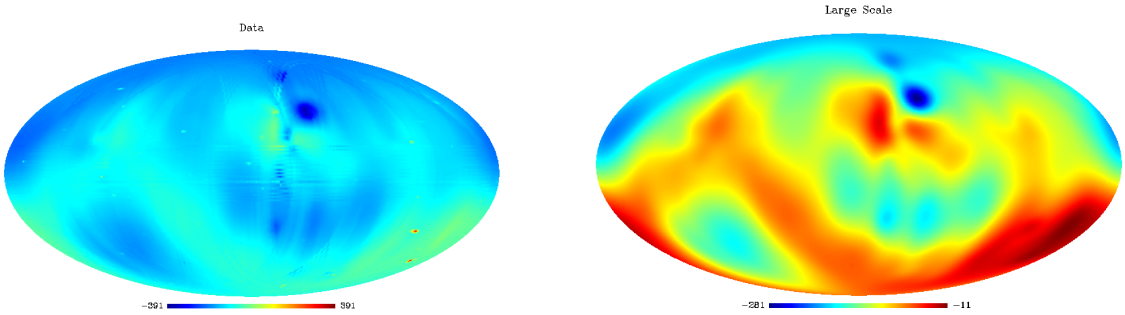


Figure 5.2: Left: Surface structures of ICF spherical shells measured on the nanometer scale are a superposition of global scale variations, isolated bumps and scratches as well as artifacts which look like interference patterns on intermediate scales. Right: Coarsest scale of the undecimated isotropic wavelet transform of the surface measurements of an ICF target.

the local cosine transforms on the sphere. The isolated bumps were thus identified and the measurement technique caused artifacts were removed easily. The resulting bumps added to the coarsest scale, is the clean data with the interference patterns and artifacts removed as shown in figure 5.3. The spherical harmonic decomposition of the cleaned image gives rise to coefficients of various ℓ modes which will be amplified by the implosion process which can now be assessed correctly using numerical hydrodynamics simulation generated growth factors. If the bumps are clustered and not randomly distributed, then systematic errors in the manufacturing process can be tracked down. A code called MODEM has been put together to study such target surface data and extract the localized bump statistics including their correlations in height, size and relative location. For more details see (Afeyan et al. 2006).

5.3 Inpainting on the Sphere

5.3.1 Algorithm

Named after the expert recovery process used for the restoration of deteriorated masterpieces, inpainting refers to a set of techniques used to alter images in a way that is undetectable to people who are unaware of the original images. There are numerous applications among which removing scratches or objects in digitized photographs, removing overlayed text or graphics, filling-in missing blocks in unreliably transmitted images, predicting values in images for better compression or image upsampling. Inpainting algorithms strive to interpolate through the gaps in the image relying on the available pixels, the continuation of edges, the periodicity of textures, etc. The preservation of edges and texture, in other words discontinuities, across gaps has attracted much interest, and many contributions have been proposed to solve this interpolation task. Non-texture image inpainting has received considerable interest and excitement since the pioneering paper by Masnou and Morel (Masnou and Morel 1998, 2002) who proposed variational principles for image disocclusion. A recent wave of interest in inpainting has started from the recent contributions of Sapiro and al. (Ballester et al. 2001; Bertalmio et al. 2001, 2000), followed by Chan and Shen (Chan and Shen 2001). In these works, authors point to the

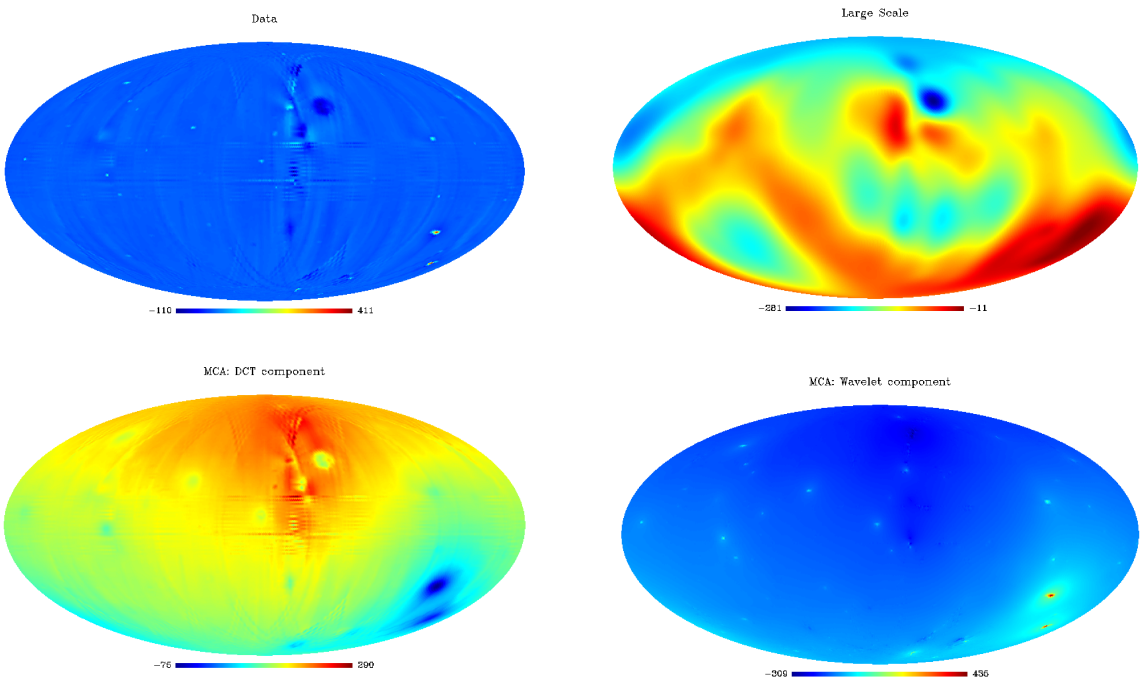


Figure 5.3: Top: Spherical map obtained by subtracting the coarse scale map on the right of Fig. 5.2 from the initial map on the left of Fig. 5.2. Bottom: Component maps separated by the MCA method on the sphere: interference patterns and measurement artifacts were caught by the local cosine functions on the sphere (left) while the isolated bumps were caught using the undecimated wavelet on the sphere (right). Adding back the coarse scale on the right of Fig. 5.2 to the latter map results in a clean map of the surface structures of an ICF spherical shell with the interference patterns and artifacts removed.

importance of geometry and design anisotropic diffusion PDEs to fill in gaps by smooth continuation of isophotes. PDE methods have been shown to perform well on piecewise smooth functions. A very different approach is the inpainting algorithm based on MCA described in (Elad et al. 2005) which has proved capable of filling in holes in either texture or cartoon content in 2D images. To make the link between building sparse representations and inpainting, consider the effect of a rectangular gap on the set of Fourier coefficients of a monochromatic sinewave : because of the non-locality of the Fourier basis functions it takes a large number of coefficients to account for the gap, which is known as the Gibbs effect. Seeking a sparse representation of the incomplete sine-wave outside the gap, that is without fitting the gap, enables the recovery of the complete monochromatic sinewave. Following (Elad et al. 2005), an inpainting algorithm on the sphere is readily built from the Morphological Component Analysis on the sphere described in the previous section. Consider a discrete spherical data map y and a binary map M such that ones in M indicate that the corresponding pixels in y are valid data while zeros indicate invalid data. The objective function of MCA equation (5.5) can be modified as follows :

$$\min_{s_1,\ldots,s_n} \lambda \sum_{k=1}^{K} \|\alpha_k\|_1 + \left\| M \odot \left(y - \sum_{k=1}^{K} s_k\right) \right\|_2^2 \quad \text{with} \quad s_k = \mathbf{\Phi}_k \alpha_k. \tag{5.10}$$

where $\odot$ stands for entry-wise multiplication. Thus we are preventing the sparse model under construction from attempting to fit the invalid data. Other constraints can be easily imposed on the interpolated sparse components. For instance, in (Elad et al. 2005), a total variation penalty is shown to enhance the recovery of piece-wise smooth components. Asking for the regularity across the gaps of some localized statistics (e.g. enforcing that the empirical variance of a given inpainted sparse component be nearly equal outside and inside the masked areas) are other possible constraints. In practice, because of the lack of accuracy of some digital transformations we used in the spherical topology, additional constraints, which may be relaxed close to convergence, were also found useful in some cases to stabilize the described iterative algorithms. It is proposed that a solution to the above minimization problem can be reached using the same iterative thresholding process as in the MCA algorithm detailed in the previous section, with the only required modification consisting in masking the full residual using M after each residual estimation, thus giving the MCA-inpainting algorithm 9.

Algorithm 9 The Morphological Component Analysis algorithm for inpainting.

Task: Compute the MCA-inpainting of masked data y.

Parameters: Data samples y, inpainting mask M, number of selected transforms K, transforms with $\mathbf{T}_k$ and $\mathbf{R}_k$ the forward and inverse transforms operators.

Initialization:

- Set the number of iterations $I_{\max}$
- Set the initial thresholds $\left(\lambda_k^{(0)}\right)_k$
- Set the final thresholds $\lambda_{\min}$, it can depend on the noise standard deviation

while $\lambda_k^{(t)} > \lambda_{\min}$ **do**

1. **for** $k = 1, \cdots, K$ **do**
   - Compute the residual term $r^{(t)}$: $r^{(t)} = y - \sum_k \tilde{s}_k^{(t-1)}$
   - Estimate the current coefficients of $\tilde{s}_k^{(t)}$ by thresholding with threshold $\lambda_k^{(t)}$: $\tilde{\alpha}_k^{(t)} = \delta_{\lambda_k^{(t)}}\left(\mathbf{T}_k\left(M \odot r^{(t)} + \tilde{s}_k^{(t-1)}\right)\right)$
   - Get the new estimate of s_k by reconstructing from the selected coeffcients $\tilde{\alpha}_k^{(t)}$: $\tilde{s}_k^{(t)} = \mathbf{R}_k \tilde{\alpha}_k^{(t)}$
2. Decrease the thresholds λ_k following a given strategy.

Output: $(\tilde{s}_k^{(m)})$ $k = 1, \ldots, K$ with $m = I_{\max}$: components separated and inpainted.

The different thresholding strategies described in the previous section can be used in the proposed MCA inpainting iterative thresholding algorithm.

Example

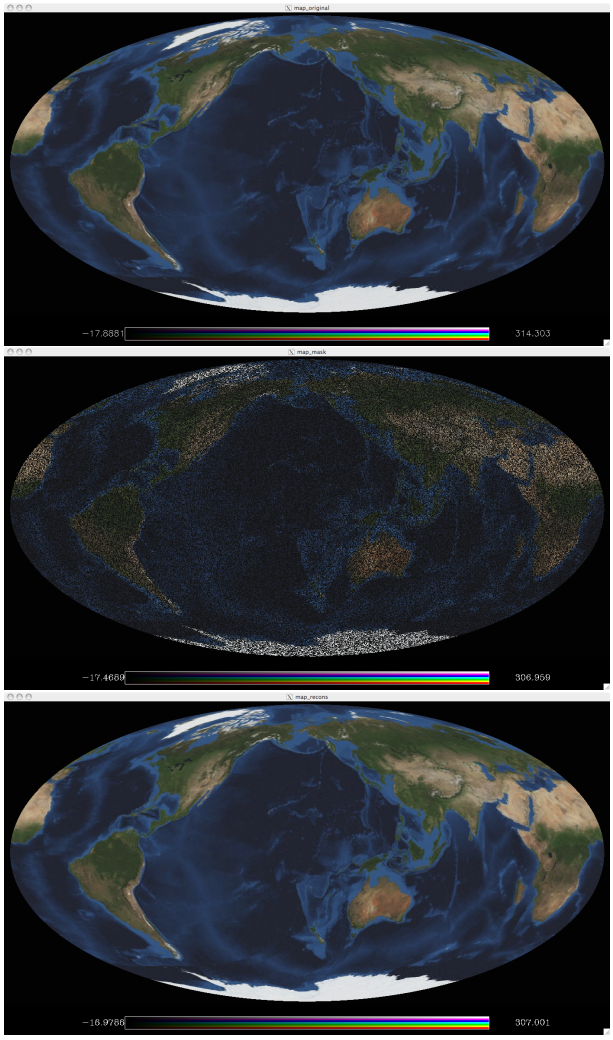


Figure 5.4: Application of the proposed MCA-inpainting algorithm on the sphere. Top: original satellite view of the Earth. Middle: incomplete map retaining 40 percent of the original pixels. Bottom: inpainted map.

A simple numerical experiment is shown in Fig. 5.4. Starting with a full satellite view of the Earth[2], an incomplete spherical map was obtained by randomly masking some of the pixels. In fact, as much as sixty percent of the pixels were masked. Using both the spherical harmonics transform and the curvelet transform on the sphere within the proposed MCA inpainting algorithm, it is possible to fill in the missing pixels in a visually undetectable way.

[2]Available from: http://www.nasa.gov/vision/earth/features/bmng_gallery_4.html

5.3.2 Application in Cosmology

A major issue in modern cosmology is the measurement and the statistical characterization (spatial power spectrum, Gaussianity) of the slight fluctuations in the Cosmic Microwave Background radiation field. These are strongly related to the cosmological scenarios describing the properties and evolution of our Universe. Some 370,000 years after the Big Bang, when the temperature of the Universe was around 3000 K, thermal energy was no longer sufficient to keep electrons and positively charged particles apart so they combined. Photons were then set free in a nearly transparent Universe. Since the Universe further expanded, these photons are now in the microwave range but they should still be distributed according to a black body emission law. Indeed, before recombination, the Universe was a highly homogeneous opaque plasma in near thermal equilibrium in which photons and charged particles were highly interacting. Hence the slight fluctuations in matter density, from which such large scale structures as galaxies or clusters of galaxies have evolved, are also imprinted on the distribution of photons.

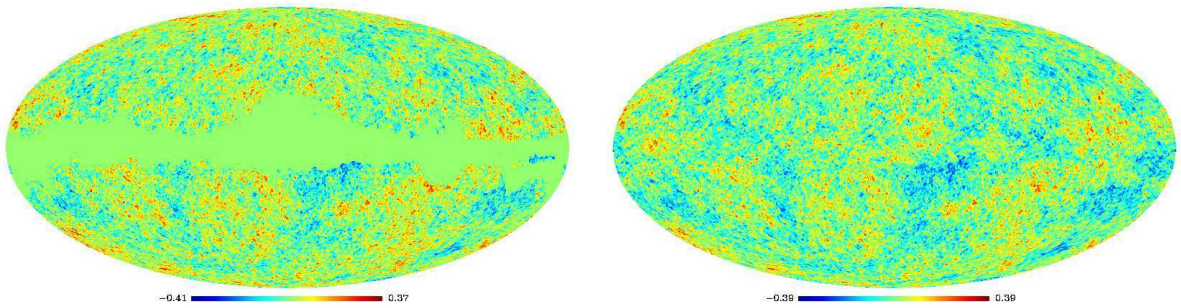


Figure 5.5: Left: CMB data map provided by the WMAP team. Areas of significant foreground contamination in the galactic region and at the locations of strong radio point sources have been masked out. Right: Map obtained by applying the MCA-inpainting algorithm on the sphere to the former incomplete WMAP CMB data map.

The Cosmic Microwave Background (CMB) was first observed in 1965 by Penzias and Wilson confirming a prediction made by Gamow in the late 1940s. But it was not until the early 1990s that evidence for small fluctuations in the CMB sky could finally be found thanks to the observations made by COBE (Smoot et al. 1992). This was confirmed by several subsequent observations and recently by NASA's Wilkinson Microwave Anisotropie Probe (WMAP) [3]. Full-sky multi-spectral observations with unprecedented sensitivity and angular resolution are expected from ESA's Planck[4] mission, which was launched in 2009. The statistical analysis of this data set will help set tighter bounds on major cosmological parameters.

A simple numerical experiment is shown in Fig. 5.5, starting with the full-sky CMB map provided by the WMAP team. This CMB map was partially masked to discard pixels where the level of contamination by residual foregrounds is expected to be the highest. Applying the described inpainting algorithm, making use of the sparsity of the

[3]The WMAP data and mask we used here are available online at http://map.gsfc.nasa.gov/
[4]http://astro.estec.esa.nl/Planck

representation of CMB in the spherical harmonics domain, leads to the map shown on the right of Fig. 5.5: the stationarity of the CMB field appears to have been restored and the masked region is completely undetectable to the eye. Fig. 5.6 shows the wavelet decomposition of the inpainted map allowing for further visual positive assessment of the quality of the proposed method as again the masked regions are undetectable at all scales. It was shown in (Abrial et al. 2007; Abrial et al. 2008) that inpainting the CMB map is an interesting approach for analyzing it, especially for non-Gaussianity studies and power spectrum estimation.

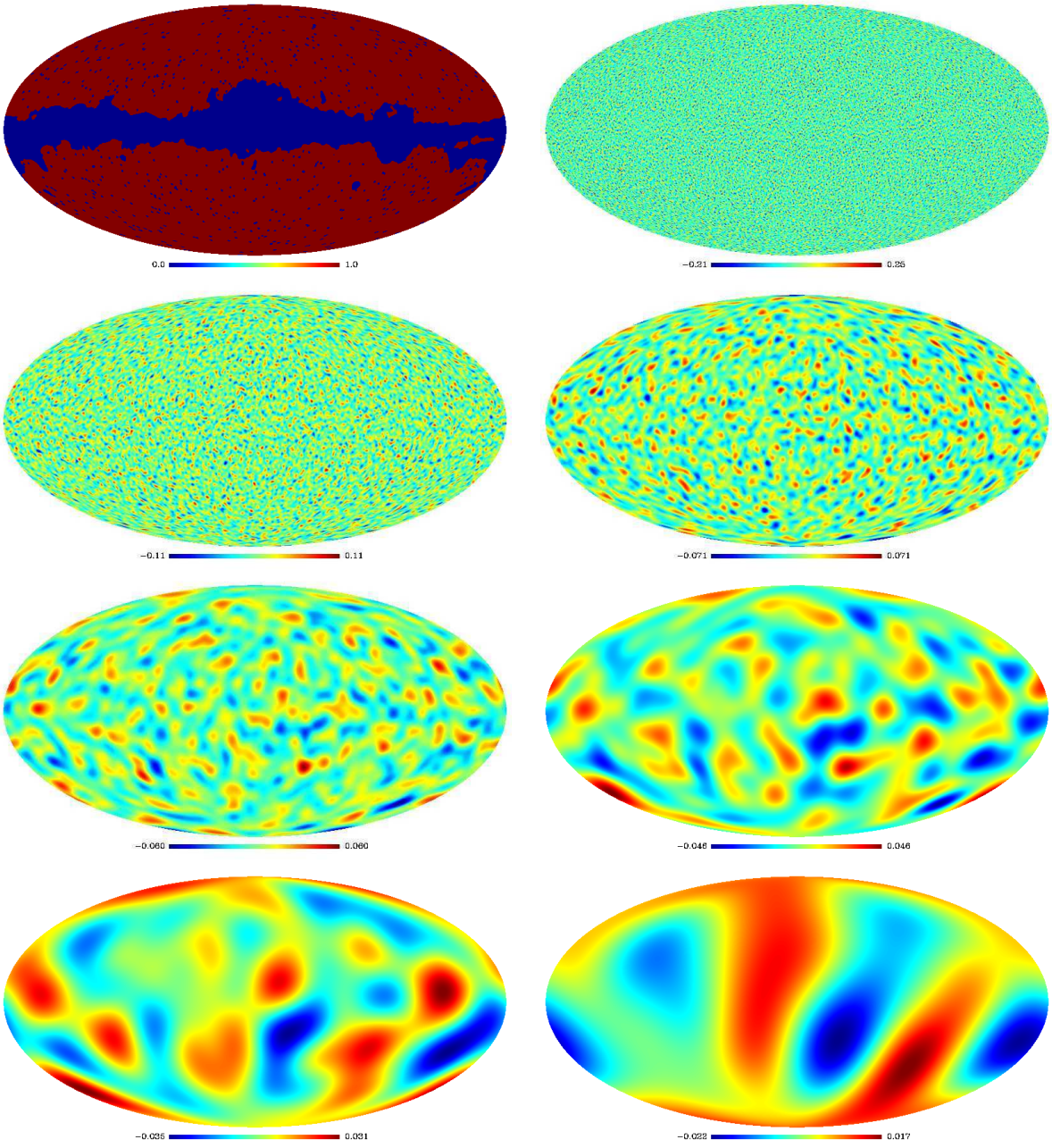


Figure 5.6: Top left: Masked area. From top to bottom and left to right: The seven wavelet scales of the inpainted map. From the visual point of view, the initially masked area cannot be distinguished any more in the wavelet scales of the inpainted map.

Chapter 6

Blind Source Separation on the Sphere

6.1 Introduction

Blind Source Separation (BSS) is a problem that occurs in multi-dimensional data processing. The overall goal is to recover unobserved signals, images or sources S from mixtures of these sources X observed typically at the output of an array of sensors. The simplest mixture model takes the form:

$$X = AS \tag{6.1}$$

where X and S are random vectors of respective sizes $m \times 1$, $n \times 1$ and A is an $m \times n$ matrix. The entries of S are assumed to be independent random variables. Multiplying S by A linearly mixes the n sources into m observed processes.

Independent Component Analysis methods were developed to solve the BSS problem, i.e. given a batch of T observed samples of X, estimate the mixing matrix A and reconstruct the corresponding T samples of the source vector S, relying mostly on the statistical independence of the source processes. Note that with the above model, the independent sources can only be recovered up to a multiplication by a non-mixing matrix i.e. up to a permutation and a scaling of the entries of S. Although independence is a strong assumption, it is in many cases physically plausible. The point is that it goes beyond the simple second order decorrelation obtained for instance using Principal Component Analysis (PCA) : decorrelation is not enough to recover the source processes since any rotation of a white random vector remains a white random vector.

Algorithms for blind component separation and mixing matrix estimation depend on the model used for the probability distribution of the sources. In a first set of techniques, source separation is achieved in a noise-less setting, based on the non-Gaussianity of all but possibly one of the components. Most mainstream Independent Component Analysis (ICA) techniques belong to this category : JADE (Cardoso 1999), FastICA, Infomax (Hyvärinen et al. 2001). In a second set of blind techniques, the components are modeled as Gaussian processes, either stationary or non stationary and, in a given representation, separation requires that the sources have diverse, i.e. non proportional,

variance profiles. The Spectral Matching ICA method (SMICA) (Delabrouille et al. 2003), considers in this sense the case of mixed stationary Gaussian components and goes further than the above model Eq. (6.1) by taking into account additive instrumental noise N:

$$X = AS + N \tag{6.2}$$

Moving to a Fourier representation, the idea is that colored components can be separated based on the diversity of their power spectra.

The next section give a short overview of two significant ICA methods mentioned above and implemented in the MRS package: JADE and FastICA (Hyvärinen et al. 2001). This is followed by a description of ways to combine wavelets and ICA techniques. Some useful properties of wavelet transforms can indeed come enhance the performance of ICA methods in several situations. Finally, we present the method GMCA, which performs a blind source separation using the the sparsity concept.

6.2 JADE

The Joint Approximate Diagonalization of Eigenmatrices method (JADE) assumes the observed data X follows the noiseless mixture model (6.1) where the independent sources S are non-Gaussian i.i.d.[1] random processes. The mixing matrix is assumed to be square and invertible so that (de)mixing is actually just a change of basis.

As mentioned above, second order statistics do not retain enough information for source separation in this context: finding a change of basis in which the data covariance matrix is diagonal will not in general enable to identify the independent sources properly. Nevertheless, decorrelation is half the job (Cardoso 1998) and one may seek the basis in which the data is represented by maximally independent processes among those bases in which the data is decorrelated. This leads to so-called orthogonal algorithms: after a proper whitening of the data by multiplication with the inverse of a square root of the covariance matrix of the data W, one is then seeking a rotation R (which leaves things white) so that $\hat{S}$ defined by

$$\hat{S} = W^{-1}\,Y = W^{-1}\,R\,X_{\text{white}} = W^{-1}\,R\,W\,X \tag{6.3}$$

and $\hat{B} = \widehat{A^{-1}} = W^{-1}\,R\,W$ are estimations of the sources and of the inverse of the mixing matrix.

JADE is such an orthogonal ICA method and, like most mainstream ICA techniques, it exploits higher order statistics so as to achieve some sort of non linear decorrelation. Precisely, in the case of JADE, statistical independence is assessed using fourth order cross cumulants :

$$\begin{aligned} F_{ijkl} &= \text{cum}(y_i, y_j, y_k, y_l) \\ &= \mathcal{E}(y_i y_j y_k y_l) - \mathcal{E}(y_i y_j)\mathcal{E}(y_k y_l) - \mathcal{E}(y_i y_l)\mathcal{E}(y_j y_k) - \mathcal{E}(y_i y_k)\mathcal{E}(y_j y_k) \end{aligned} \tag{6.4}$$

[1]The letters i.i.d. stand for independently and identically distributed meaning that each entries of X at a given time t are independent of X at any other time t' and that the distribution of X does not depend on time.

where $\mathcal{E}$ stands for statistical expectation and the y_i's are the entries of vector Y modeled as random variables, and the correct change of basis (i.e. rotation) is found by somehow diagonalizing the fourth order cumulant tensor. Indeed, if the y_i's were independent, all the cumulants with at least two different indices would be zero. As a consequence of the independence assumption of the source processes S and of the whiteness of Y for all rotations R, the fourth order tensor F is well structured: JADE was precisely devised to take advantage of the algebraic properties of F. JADE's objective function is given by

$$\mathcal{J}_{\text{jade}}(R) \;=\; \sum_{ij} \sum_{k \neq l} \text{cum}(y_i, y_j, y_k, y_l)^2$$

which can be interpreted as a joint diagonalization criterion. Fast and robust algorithms are available for the minimization of $\mathcal{J}_{\text{jade}}(R)$ with respect to R based on Jacobi's method for matrix diagonalization (Pham 2001). More details on JADE can be found in (Cardoso 1999, 1998; Hyvärinen et al. 2001).

JADE for spherical maps

Applying JADE on multichannel data mapped to the sphere does not require any particular modification of the algorithm. Indeed, JADE estimates the fourth order cumulant tensor from the available data samples assuming an i.i.d. random field. Hence, given a pixelization scheme on the sphere such as provided by the Healpix package, JADE can be directly applied to the multichannel spherical data pixels.

6.3 FastICA

FastICA is by now a standard technique in ICA. Like JADE, it is meant for the analysis of mixtures of independent non-Gaussian sources in a noise-less setting. A complete description of this method can be found in (Hyvärinen et al. 2001) and references therein[2]. We give here a brief and simplified account of the algorithm. FastICA, again like JADE, is a so-called orthogonal ICA method: the independent components are sought by maximizing a measure of non-Gaussianity under the constraint that they are decorrelated. Intuitively, one should understand that mixtures of independent non-Gaussian random variables tend to look more Gaussian. An enlightening view on the relation between mutual information, which is a natural measure of independence, decorrelation and non-Gaussianity can be found in (Cardoso 2001, 2003). Non-Gaussianity is assessed in FastICA using a contrast function G based on a non-linear approximation to negentropy (Hyvärinen et al. 2001). In practice, depending on the application, different approximations or non-linear (non-quadratic) functions should be experimented with. In a simple deflation scheme, for sphered data, the directions are found sequentially : a direction r of maximal non-Gaussianity is sought by maximizing

$$J_G(r) = \Big(\mathcal{E}\{G(r^T x_{\text{white}})\} - \mathcal{E}\{G(\nu)\} \Big)^2 \tag{6.5}$$

where ν stands for centered unit variance Gaussian variable, under the constraint that r has unit norm and that r is orthogonal to the directions found previously.

[2]Many papers on this algorithm are available at http://www.cs.helsinki.fi/u/ahyvarin/papers/fastica.shtml

The contrast function G can for instance be chosen among the following (Hyvärinen et al. 2001):

$$\begin{aligned} G_0(u) &= \frac{1}{a}\log\cosh(au) \\ G_1(u) &= -\frac{1}{a}\exp(-au^2/2) \\ G_2(u) &= \frac{1}{4}u^4 \end{aligned} \tag{6.6}$$

where a is a constant to be determined depending on the application. It can be shown that the maxima of J_G occur at certain maxima of $\mathcal{E}\{G(r^T x_{\text{white}})\}$. These are obtained for r solution to :

$$\mathcal{E}\{x_{\text{white}} g(r^T x_{\text{white}})\} - \lambda r = 0 \tag{6.7}$$

where λ is a constant easily expressed in terms of the optimal direction r_0, and g is the derivative of G. Solving this equation using Newton's method, and a few approximations, a fixed-point algorithm is derived which consists in repeating the following two steps until convergence :

$$\begin{aligned} r &\leftarrow \mathcal{E}\{x_{\text{white}} g(r^T x_{\text{white}})\} - \mathcal{E}\{g'(r^T x_{\text{white}})\} r \\ r &\leftarrow \frac{r}{\|r\|} \end{aligned} \tag{6.8}$$

A simple implementation of this algorithm is included in the present package. It is largely based on the **Matlab**^{TM} code available at www.cis.hut.fi/projects/ica/fastica/.

6.4 Wavelet and BSS

Wavelets come into play as a sparsifying transform. Applying a wavelet transform on both sides of (6.1) does not affect the mixing matrix and the model structure is preserved. Also, moving the data to a wavelet representation does not affect its information content. However, the statistical distribution of the data coefficients in the new representation is different: wavelets are known to lead to sparse i.i.d. representations of structured data. Further, the local (coefficient wise) signal to noise ratio depends on the choice of a representation. A wavelet transform tends to grab the informative coherence between pixels while averaging the noise contributions, thus enhancing structures in the data. Although the standard ICA model (6.1) is for a noiseless setting, the derived methods can be applied to real data. Performance will depend on the detectability of significant coefficients i.e. on the sparsity of the statistical distribution of the coefficients. Moving to a wavelet representation will often lead to more robustness to noise.

Once the data has been transformed to a proper representation (e.g. wavelets but also ridgelets and curvelets in the case of strongly anisotropic 2D or 3D data), WJADE (resp. WFastICA) consists in applying the standard JADE (resp. FastICA) method to the new multichannel coefficients. Once the mixing matrix is estimated, the initial source

maps are obtained using the adequate inverse transform after some non linear denoising or thresholding of the coefficients if necessary.

6.5 Sparse Blind Source Separation: the GMCA method

6.5.1 Morpho-Spectral Diversity

Extending the redundant representation framework to the multichannel case requires defining what a multichannel overcomplete representation is. Let us assume in this section that $\mathbf{A} = [\varphi_{\nu,1}, \cdots, \varphi_{\nu,N_c}] \in \mathbb{R}^{N_c \times N_s}$ is a *known spectral* dictionary, and $\mathbf{\Phi} = [\varphi_1, \cdots, \varphi_T] \in \mathbb{R}^{N \times T}$ is a *spatial* or *temporal* dictionary[3]. We assume that each source s_i can be represented as a (sparse) linear combination of atoms in $\mathbf{\Phi}$; $s_i = \mathbf{\Phi}\alpha_i$. Let $\boldsymbol{\alpha}$ the $N_s \times T$ matrix whose rows are α_i^{T}.

From Eq. (6.1), the multichannel noiseless data $\mathbf{Y}$ can be written as

$$\mathbf{Y} = \mathbf{A}\boldsymbol{\alpha}\mathbf{\Phi}^{\mathrm{T}} = \sum_{i=1}^{N_s} \sum_{j=1}^{T} \left(\varphi_{\nu,i}\varphi_j^{\mathrm{T}}\right) \alpha_i[j] \ . \tag{6.9}$$

Consequently, each column in of $\mathbf{Y}$ reads

$$\mathbf{Y}[., l] = (\mathbf{A} \otimes \mathbf{\Phi}[l, .]) \, \mathrm{vect}(\boldsymbol{\alpha}) \ , \quad \forall \ l = 1, \cdots, N \ , \tag{6.10}$$

and finally

$$\mathrm{vect}(\mathbf{Y}) = (\mathbf{A} \otimes \mathbf{\Phi}) \, \mathrm{vect}(\boldsymbol{\alpha}) \ , \tag{6.11}$$

where $\otimes$ is the tensor (Kronecker) product and the operator vect stacks the columns of its argument in a long 1D vector. This latter equation brings a clear and simple insight: the sparsity of the sources in $\mathbf{\Phi}$ translates into sparsity of the multichannel data $\mathbf{Y}$ in the multichannel tensor product dictionary $\mathbf{\Psi} = \mathbf{A} \otimes \mathbf{\Phi}$.

The multichannel dictionary $\mathbf{\Psi}$ can also be seen as concatenation of multichannel atoms $\mathbf{\Psi}^{(ij)} = \varphi_{\nu,i}\varphi_j^{\mathrm{T}}$ which are rank-one matrices obtained from each atomic spectrum $\varphi_{\nu,i}$ and each spatial elementary atom φ_j (see Eq. (6.9)).

Some of the popular recovery results in sparse component analysis algorithm rely on the mutual coherence of the dictionary (Elad and Bruckstein 2002; Gribonval and Nielsen 2003; Donoho and Elad 2003). In the multichannel case a quantity of that kind can be defined. In fact, by standard properties of the tensor product, one can easily show that the Gram matrix of a tensor product is the tensor product of the Gram matrices. Thus the mutual coherence of the multichannel dictionary $\mathbf{\Psi}$ is:

$$0 \leq \mu_{\mathbf{\Psi}} = \max\left\{\mu_{\mathbf{A}}, \mu_{\mathbf{\Phi}}\right\} < 1 \ . \tag{6.12}$$

This expression of mutual coherence is instructive as it tells us that multichannel atoms can be distinguished based on their spatial or spectral morphology. In other words, discriminating two multichannel atoms Ψ_{ij} and $\Psi_{i'j'}$ may put on different faces:

[3]The adjectives *spectral* and *spatial* that characterize the dictionaries are not formal. Owing to the symmetry of the multichannel sparse decomposition problems, $\mathbf{A}$ and $\mathbf{\Phi}$ have no formal difference. In practice and more particularly in multi/hyperspectral imaging, $\mathbf{A}$ will refer to the dictionary of physical spectra and $\mathbf{\Phi}$ to the dictionary of image/signal waveforms. In the BSS problem, $\mathbf{A}$ is unknown.

- Spatial or temporal (respectively spectral) diversity: in this case $i = i'$ and $j \neq j'$ (respectively $i \neq i'$ and $j = j'$). These atoms have the same spectrum (respectively, spatial shape) but one can discriminate between them based on their spatial (respectively, spectral) diversity. From (6.12), their coherence is lower than $\mu_{\mathbf{\Phi}}$ (respectively $\mu_{\mathbf{A}}$). Disentangling these multichannel atoms can equivalently be done in 5.
- Both diversities: $i \neq i'$ and $j \neq j'$, this seems to be a more favorable scenario to differentiate the atoms as they do not share neither the same spectrum nor the same spatial (or temporal) "shape". Note that from (6.12), the coherence between these atoms in this case is lower than $\mu_{\mathbf{A}}\mu_{\mathbf{\Phi}} \leq \max\{\mu_{\mathbf{A}}, \mu_{\mathbf{\Phi}}\}$.

6.5.2 Multichannel Sparse Decomposition

We embark from Eq. (6.9), where the multichannel dictionary $\mathbf{\Psi}$ is supposed to be overcomplete, i.e. $NN_c < TN_s$. The goal is to recover the sparsest solution $\boldsymbol{\alpha}$ from $\mathbf{Y}$ which requires solving:

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^{N_s \times T}} \sum_{i=1}^{N_s} \|\alpha_i\|_0 \quad \text{s.t.} \quad \mathbf{Y} = \mathbf{A}\boldsymbol{\alpha}\mathbf{\Phi}^{\mathrm{T}}. \tag{6.13}$$

As justified in 5.2, this combinatorial problem can be replaced by its convex relaxation substituting the ℓ_1 norm for the ℓ_0 pseudo-norm, hence giving:

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^{N_s \times T}} \sum_{i=1}^{N_s} \|\alpha_i\|_1 \quad \text{s.t.} \quad \mathbf{Y} = \mathbf{A}\boldsymbol{\alpha}\mathbf{\Phi}^{\mathrm{T}}. \tag{6.14}$$

As (6.11) is a vectorized monochannel form of (6.9), what we are trying so do is actually to find the sparsest solution of a monochannel underdetermined system of linear equations where the solution is sparse in an overcomplete tensor product dictionary. Recovery properties of monochannel sparse decomposition by ℓ_1 minimization were overviewed in chapter 5. Therefore, if one is able to translate those identifiability criteria in the language of tensor product dictionaries, then we are done.

In particular, the coherence-based sparse recovery criterion given in (Donoho and Huo 2001) is trivial to adapt owing to (6.12). Indeed, if $\mathbf{Y}$ is k-sparse in the multichannel dictionary $\mathbf{\Psi}$ with $k < C(\mu_{\mathbf{\Psi}}^{-1} + 1)$ for some $C > 0$ (typically $C = 1/2$), and the dictionary is sufficiently incoherent (both spectrally and spatially), then the solution of Eq. (6.14) is unique, is a point of equivalence of Eq. (6.13) and Eq. (6.14), and the recovery is stable to bounded noise on $\mathbf{Y}$.

Above, we addressed the multichannel sparse decomposition problem without assuming any constraint on the sparsity pattern of the different channels. It is worth however pointing out that sparse recovery conditions from multichannel measurements can be refined if some structured sparsity is hypothesized. For instance, for structured multichannel representation (e.g. sources with disjoint supports) (Gribonval and Nielsen 2008) provided coherence-based sufficient recovery conditions by solving Eq. (6.14). One should note that despite apparent similarities, the multichannel sparse decomposition problem discussed here is conceptually different from the one targeting *simultaneous* sparse recovery of multiple measurements vectors (MMV) considered by several authors, see e.g.

(Cotter et al. 2005; Malioutov et al. 2005; Tropp 2006; Chen and Huo 2006; Argyriou et al. 2008; Bach 2008; Gribonval et al. 2008; Eldar and Mishali 2009; Lounici et al. 2009; Negahban and Wainwright 2009). The latter are not aware of any mixing process via $\mathbf{A}$, and their goal is to recover $\boldsymbol{\alpha}$ from MMV $\mathbf{Y} = \boldsymbol{\alpha}\boldsymbol{\Phi}^{\mathrm{T}}$ in which the vectors α_i, i.e. rows of $\boldsymbol{\alpha}$, have a common sparsity pattern. However the MMV model can also be written $\mathrm{vect}(\mathbf{Y}^{\mathrm{T}}) = (\boldsymbol{\Phi} \otimes \mathbf{I})\,\mathrm{vect}(\boldsymbol{\alpha}^{\mathrm{T}})$ as in Eq. (6.11). The most widely used approach to solve the simultaneous sparse recovery problem with joint sparsity is to minimize a mixed $\ell_p - \ell_q$ norm of the form $\sum_{j=1}^{T} \left(\|\boldsymbol{\alpha}[., j]\|_p^q \right)^{1/q}$ for $p \geq 1, 0 \leq q \leq +\infty$.

6.5.3 Generalized Morphological Component Analysis

We now turn to the BSS problem and highlight the role of sparsity and morphological diversity as a source of contrast to solve it. Towards this goal, we assume that the sources are sparse in the spatial dictionary $\boldsymbol{\Phi}$ that is the concatenation of K orthonormal bases $(\boldsymbol{\Phi}_k)_{k=1,\cdots,K}$: $\boldsymbol{\Phi} = [\boldsymbol{\Phi}_1, \cdots, \boldsymbol{\Phi}_K]$. The restriction to orthonormal bases is only formal and the algorithms to be presented later still work in practice even with redundant sub-dictionaries $\boldsymbol{\Phi}_k$.

The Generalized Morphological Component Analysis framework assumes a priori that each source is modeled as the linear combination of K morphological components where each component is sparse in a specific basis:

$$\begin{aligned} \forall i \in \{1, \cdots, N_s\}; \qquad s_i &= \sum_{k=1}^{K} x_{i,k} = \sum_{k=1}^{K} \mathbf{\Phi}_k \alpha_{i,k} \\ &= \mathbf{\Phi}\alpha_i \qquad \text{where } \alpha_i = \left[\alpha_{i,1}^{\mathrm{T}}, \cdots, \alpha_{i,K}^{\mathrm{T}}\right]^{\mathrm{T}} \end{aligned} \tag{6.15}$$

GMCA seeks an unmixing scheme, through the estimation of $\mathbf{A}$, which leads to the sparsest sources $\mathbf{S}$ in the dictionary $\mathbf{\Phi}$. This is expressed by the following optimization problem written in the augmented Lagrangian form:

$$\min_{\mathbf{A}, \alpha_{1,1}, \cdots, \alpha_{N_s,K}} \frac{1}{2} \left\| \mathbf{Y} - \mathbf{A}\boldsymbol{\alpha}\mathbf{\Phi}^{\mathrm{T}} \right\|_{\mathrm{F}}^2 + \lambda \sum_{i=1}^{N_s} \sum_{k=1}^{K} \|\alpha_{i,k}\|_p^p \quad \text{s.t.} \quad \|a_i\|_2 = 1 \ \forall i \in \{1, \cdots, N_s\} \tag{6.16}$$

where typically $p = 0$ or its relaxed convex version with $p = 1$, and $\|\mathbf{X}\|_{\mathrm{F}} = \left(\mathrm{trace}(\mathbf{X}^{\mathrm{T}}\mathbf{X})\right)^{1/2}$ is the Frobenius norm. The unit ℓ_2-norm constraint on the columns of $\mathbf{A}$ avoids the classical scale indeterminacy of the product $\mathbf{AS}$ in (6.2). The reader may have noticed that the MCA problem in chapter 5 is a special case of the GMCA problem Eq. (6.16) when there is only one source $N_s = 1$ and one channel $N_c = 1$ (no mixing). Thus GMCA is indeed a multichannel generalization of MCA.

The program (6.16) is a notoriously difficult non-convex optimization problem even for convex penalties when $p \geq 1$. More conveniently, following (6.1), the product $\mathbf{AS}$ can be split into $N_s \cdot K$ multichannel morphological components: $\mathbf{AS} = \sum_{i,k} a_i x_{i,k}^{\mathrm{T}} = \sum_{i,k} (a_i \alpha_{i,k}^{\mathrm{T}}) \mathbf{\Phi}_k^{\mathrm{T}}$. Based on this decomposition, and inspired by the block-coordinate relaxation as for MCA, GMCA yields an alternating minimization algorithm to estimate iteratively one term at a time (Bobin et al. 2007a). We will show shortly that the estimation of each morphological component $x_{i,k} = \mathbf{\Phi}_k \alpha_{i,k}$ assuming $\mathbf{A}$ and $x_{\{i',k'\} \neq \{i,k\}}$ are fixed is obtained by simple hard or soft thresholding for $p = 0$ and $p = 1$.

Define the $(i,k)^{\text{th}}$ multichannel marginal residual by:

$$\mathbf{R}_{i,k} = \mathbf{Y} - \sum_{i' \neq i} \sum_{k' \neq k} a_{i'} x_{i',k'}^{\mathrm{T}} \tag{6.17}$$

as the part of the data $\mathbf{Y}$ unexplained by the multichannel morphological component $a_i x_{i,k}^{\mathrm{T}}$. Estimating $x_{i,k} = \mathbf{\Phi}_k \alpha_{i,k}$, assuming $\mathbf{A}$ and the other components $x_{(i',k') \neq (i,k)}$ are fixed, leads to the component-wise optimization problem:

$$\min_{x_{i,k} \in \mathbb{R}^N} \frac{1}{2} \left\| \mathbf{R}_{i,k} - (a_i \alpha_{i,k}^{\mathrm{T}}) \mathbf{\Phi}^{\mathrm{T}} \right\|_{\mathrm{F}}^2 + \lambda \|\alpha_{i,k}\|_p^p \tag{6.18}$$

Since here $\mathbf{\Phi}_k$ is an orthogonal matrix, with calculations using proximity operators, it can be shown that the unique solution of Eq. (6.18) is obtained by a hard ($p = 0$) or soft ($p = 1$) thresholding. Hence, the closed-form estimate of the morphological component $x_{i,k}$ is:

$$\tilde{x}_{i,k} = \Delta_{\mathbf{\Phi}_k, \lambda'} \left(\frac{1}{\|a_i\|_2^2} \mathbf{R}_{i,k}^{\mathrm{T}} a_i \right) \tag{6.19}$$

where $\lambda' = \lambda / \|a_i\|_2^2$ for soft thresholding and $\lambda' = \sqrt{2\lambda}/\|a_i\|_2$ for hard thresholding. The operator $\Delta_{\mathbf{D},\lambda}(x)$ consists of (i) computing the coefficients of x in the dictionary $\mathbf{D}$, (ii) thresholding (soft or hard) the obtained coefficients with the threshold λ, and (iii) reconstructing from thresholded coefficients:

$$\Delta_{\mathbf{D},\lambda}(x) = \mathbf{D}\,\mathrm{Thresh}_\lambda\left(\mathbf{D}^{\mathrm{T}}x\right) \tag{6.20}$$

Thresh_λ is either a hard or a soft thresholding. When $\mathbf{\Phi}_k$ is redundant, (6.19) is only the first iteration of a forward-backward splitting recursion, and which should be used when $\mathbf{\Phi}_k$ is overcomplete. However in practice (6.19) can still be used to save computation time.

Now, considering $\{a_{i'}\}_{i'\neq i}$ and all morphological components as fixed, and recalling that $N_c \geq N_s$, updating the column a_i is then just a least-squares estimate

$$\tilde{a}_i = \frac{1}{\|s_i\|_2^2}\left(\mathbf{Y} - \sum_{i'\neq i} a_{i'} s_{i'}^{\mathrm{T}}\right) s_i \tag{6.21}$$

where $s_i = \sum_{k=1}^{K} x_{i,k}$. This estimate is then projected onto the unit sphere to meet the unit ℓ_2-norm constraint in Eq. (6.16). The GMCA algorithm is summarized in Algorithm 10.

Algorithm 10 GMCA algorithm.

Task: Sparse Blind Source Separation.

Parameters: The data $\mathbf{Y}$, the dictionary $\mathbf{\Phi} = [\mathbf{\Phi}_1 \cdots \mathbf{\Phi}_K]$, number of iterations N_{iter}, number of sources N_s and channels N_c, stopping threshold $\lambda_{\min}$, threshold update schedule.

Initialization: $x_{i,k}^{(0)} = 0$ for all (i,k), $\mathbf{A}^{(0)}$ random and threshold λ_0.

Main iteration:

for $t = 1$ **to** N_{iter} **do**

  for $i = 1, \cdots, N_s$ **do**

    for $k = 1, \cdots, K$ **do**

      Compute the marginal residuals:

$$\mathbf{R}_{i,k}^{(t)} = \mathbf{Y} - \sum_{(i',k')\neq(i,k)} a_{i'}^{(t-1)} x_{i',k'}^{(t-1)^{\mathrm{T}}}.$$

      Estimate the current component $x_{i,k}^{(t)}$ via thresholding with threshold λ_t:

      $x_{i,k}^{(t)} = \Delta_{\mathbf{\Phi}_k,\lambda_t}\left(\mathbf{R}_{i,k}^{(t)^{\mathrm{T}}} a_i^{(t-1)}\right)$.

    Update ith source $s_i^{(t)} = \sum_{k=1}^{K} x_{ik}^{(t)}$.

    Update a_i assuming $a_{i'\neq i}^{(t)}$ and the morphological components $x_{i,k}^{(t)}$ are fixed :

    $a_i^{(t)} = \frac{1}{\|s_i^{(t)}\|_2^2}\left(\mathbf{Y} - \sum_{i'\neq i}^{N_s} a_{i'}^{(t-1)} s_{i'}^{(t)^{\mathrm{T}}}\right) s_i^{(t)}$ and normalize to a unit ℓ_2 norm.

  Update the threshold λ_t according to the given schedule.

  if $\lambda_t \leq \lambda_{\min}$ **then** stop.

Output: Estimated sources $\left(s_i^{(N_{\mathrm{iter}})}\right)_{i=1,\cdots,N_s}$ and mixing matrix $\mathbf{A}^{(N_{\mathrm{iter}})}$.

For $p = 1$ and fixed threshold λ, Algorithm 10 can be shown to converge to a stationary point, see (Tseng 2001; Bobin et al. 2008). This point is not guaranteed to be even a

local minimum of the energy, and this is even less clear for $p = 0$. Thus, in the same vein as MCA, GMCA relies on a salient-to-fine strategy using a varying threshold to mitigate the problem of sensitivity to initialization. More precisely, GMCA first computes coarse versions of the morphological components for any fixed source s_i. These raw sources are estimated from their most significant coefficients in $\mathbf{\Phi}$. Then, the corresponding column a_i is estimated from the most significant features of s_i. Each source and its corresponding column of $\mathbf{A}$ are then alternately and progressively refined as the threshold decreases towards $\lambda_{\min}$. This particular iterative thresholding scheme provides robustness to noise and initialization by working first on the most significant features in the data and then progressively incorporating smaller details to finely tune the model parameters. GMCA can be used with either linear or exponential decrease of the threshold as for MCA.

If $\mathbf{A}$ were known and fixed, the GMCA would be equivalent to performing an MCA sparse decomposition of $\mathbf{Y}$ in the tensor product multichannel dictionary $\mathbf{A} \otimes \mathbf{\Phi}$. But as GMCA also updates the mixing matrix at each iteration, it is able to learn the spectral part of the multichannel dictionary directly from the data.

The Thresholding Strategy

Hard or soft thresholding? In practice, it was observed that hard thresholding leads to better results (Bobin et al. 2006, 2007a). Furthermore, if $\mathbf{A}$ is known and no noise contaminates the data, GMCA with hard thresholding will enjoy the sparse recovery guarantees given in (Bobin et al. 2007b, 2009), with the proviso that the morphological components are contrasted and sparse in a sufficiently incoherent multichannel dictionary $\mathbf{A} \otimes \mathbf{\Phi}$.

Handling additive Gaussian noise. The GMCA algorithm is well suited to deal with data contaminated with additive Gaussian noise (see the next section for a Bayesian interpretation). For instance, assume that the noise $\mathbf{E}$ in (6.2) is additive white Gaussian in each channel, i.e. its covariance matrix $\mathbf{\Sigma_E}$ is diagonal, and let $\sigma_{\mathbf{E}}$ be its standard deviation supposed equal for all channels for simplicity. Then, Algorithm 10 can be applied as described above with $\lambda_{\min} = \tau\sigma_{\mathbf{E}}$, where τ is chosen as in denoising methods, typically taking its value in the range $[3, 4]$. This attribute of GMCA makes it a suitable choice for use in noisy BSS. GMCA not only manages to separate the sources, but also succeeds in removing additive noise as a by-product.

6.5.4 The Bayesian Perspective

GMCA can be interpreted from a Bayesian standpoint. For instance, let us assume that the entries of the mixtures $(y_i)_{i=1,\cdots,N_c}$, the mixing matrix $\mathbf{A}$, the sources $(s_i)_{i=1,\cdots,N_s}$ and the noise matrix $\mathbf{E}$ are random processes. We assume that the noise $\mathbf{E}$ is zero-mean Gaussian where the noise vector ε_i in each channel is white, but the noise between channels is possibly correlated with known covariance matrix $\mathbf{\Sigma_E}$. This means that the log-likelihood function takes the form:

$$LL(\mathbf{Y}\big|\mathbf{S}, \mathbf{A}, \mathbf{\Sigma_E}) = \frac{1}{2} \|\mathbf{Y} - \mathbf{AS}\|^2_{\mathbf{\Sigma_E}} \ , \ \text{where} \ \|\mathbf{X}\|^2_{\mathbf{\Sigma_E}} = \text{trace}\left(\mathbf{X}^{\mathrm{T}}\mathbf{\Sigma_E}^{-1}\mathbf{X}\right) \tag{6.22}$$

We further assume that the uniform prior is imposed on entries of $\mathbf{A}$. Other priors on $\mathbf{A}$ could be imposed; e.g. known fixed column for example. As far as the sources are concerned, they are known from (6.15) to be sparse in the dictionary $\mathbf{\Phi}$. Thus their coefficients $\boldsymbol{\alpha} = [\alpha_1, \cdots, \alpha_{N_s}]^{\mathrm{T}}$ will be assumed as drawn independently from a leptokurtic PDF with heavy tails such as the generalized Gaussian distribution form:

$$\mathrm{pdf}_{\boldsymbol{\alpha}}(\alpha_{1,1}, \ldots, \alpha_{N_s,K}) \propto \prod_{i=1}^{N_s} \prod_{k=1}^{K} \exp\left(-\lambda_{i,k} \left\|\alpha_i\right\|_{p_{i,k}}^{p_{i,k}}\right) \\ 0 \leq p_{i,k} < 2 \; \forall (i,k) \in \{1, \cdots, N_s\} \times \{1, \cdots, K\} \quad (6.23)$$

Putting together the log-likelihood function and the priors on $\mathbf{A}$ and $\boldsymbol{\alpha}$, the MAP estimator leads to the following optimization problem:

$$\min_{\mathbf{A}, \alpha_{1,1}, \cdots, \alpha_{N_s,K}} \frac{1}{2} \left\|\mathbf{Y} - \mathbf{A}\boldsymbol{\alpha}\mathbf{\Phi}^{\mathrm{T}}\right\|_{\mathbf{\Sigma_E}}^2 + \sum_{i=1}^{N_s} \sum_{k=1}^{K} \lambda_{i,k} \left\|\alpha_{i,k}\right\|_{p_{i,k}}^{p_{i,k}} \quad (6.24)$$

This problem has strong similarity with that of Eq. (6.16). More precisely, if the noise is homoscedastic and decorrelated between channels (i.e. $\mathbf{\Sigma_E} = \sigma_{\mathbf{E}}^2 \mathbf{I}$), if the shape parameters $p_{i,k}$ of the generalized Gaussian distribution prior are all equal to p and the scale parameters are all taken as $\lambda_{i,k} = \lambda / \sigma_{\mathbf{E}}^2$, and if the columns of $\mathbf{A}$ are assumed uniform on the unit sphere, then Eq. (6.24) is exactly Eq. (6.16). Note that in the development above, the independence assumption in (6.23) does not necessarily entail independence of the sources. Rather it means that there are no a priori assumptions that indicate any dependency between the sources.

6.5.5 The Fast GMCA Algorithm

The goal here is to speed up the GMCA algorithm. As a warm-up, assume that the dictionary $\mathbf{\Phi}$ is no longer redundant and reduces to a single orthobasis (i.e. $K = 1$). Let us denote $\boldsymbol{\beta} = \mathbf{Y}\mathbf{\Phi}$ the matrix where each of its rows stores the coefficients of each channel y_i. The optimization problem Eq. (6.16) then becomes (we omit the ℓ_2 constraint on $\mathbf{A}$ to lighten the notation):

$$\min_{\mathbf{A}, \boldsymbol{\alpha}} \frac{1}{2} \left\|\boldsymbol{\beta} - \mathbf{A}\boldsymbol{\alpha}\right\|_{\mathrm{F}}^2 + \lambda \sum_{i=1}^{N_s} \left\|\alpha_i\right\|_p^p \quad (6.25)$$

where $p = 0$ or $p = 1$. The GMCA algorithm no longer needs to apply the analysis and synthesis operators at each iteration as only the channels $\mathbf{Y}$ have to be transformed once in $\mathbf{\Phi}$. Clearly, this case is computationally much cheaper.

However, this is rigorously valid only for an orthobasis dictionary, and no orthonormal basis is able to sparsely represent large variety of signals and yet we would like to use very sparse signal representations which motivated the use of redundancy in the first place. Arguments supporting the substitution of (6.25) for (6.16) for a redundant dictionary $\mathbf{\Phi}$ were given in (Bobin et al. 2007a, 2008). The idea is to first compute the sparsest representation of each channel y_i in the redundant dictionary $\mathbf{\Phi}$ using an appropriate (non-linear) decomposition algorithm (e.g. BP, MCA). Now, $\boldsymbol{\beta}$ denotes the matrix where each row

contains the sparse decomposition of the corresponding channel. Because the channels are linear mixtures of the sources via the mixing matrix $\mathbf{A}$, the key argument developed by (Bobin et al. 2007a) is that the sparse decomposition algorithm must preserve linear mixtures. Descriptively, the sparsest decomposition provided by the algorithm when applied to each channel must be equal to the linear combination of the sparsest decompositions of the sources. This statement is valid if the sources and the channels are identifiable, meaning that they verify sufficient conditions so that their unique sparsest representation can be recovered by the decomposition algorithm. For instance, if MCA is used, then it is sufficient as in (Bobin et al. 2007b, 2009) that the channels and the sources be sparse enough in an incoherent dictionary $\mathbf{\Phi}$, and their morphological components be sufficiently contrasted. See (Bobin et al. 2007a, 2008) for details.

Hence, under these circumstances, a fast GMCA algorithm can be designed to solve (6.25) by working in the transform domain after decomposing each observed channel y_i in $\mathbf{\Phi}$ using a sparse decomposition algorithm such as MCA. There is an additional important simplification when substituting problem (6.25) for (6.16). Indeed, since $N_c \geq N_s$ (i.e. overdetermined BSS), it turns out that (6.25) is a multichannel overdetermined least-squares fit with ℓ_0/ℓ_1-sparsity penalization. We again use an alternating minimization scheme to solve for $\mathbf{A}$ and $\boldsymbol{\alpha}$:

- Update the coefficients: when $\mathbf{A}$ is fixed, since the quadratic term is strictly convex ($\mathbf{A}$ has full column-rank), the marginal optimization problem can be solved by a general form of the forward-backward splitting iteration (Chen and Rockafellar 1997):
$$\boldsymbol{\alpha}^{(t+1)} = \mathrm{Thresh}_{\mu\lambda}\left(\boldsymbol{\alpha}^{(t)} + \mu\mathbf{\Xi}\mathbf{A}^{\mathrm{T}}(\boldsymbol{\beta} - \mathbf{A}\boldsymbol{\alpha}^{(t)})\right) \tag{6.26}$$
where $\mathbf{\Xi}$ is a relaxation matrix such that the spectral radius of $(\mathbf{I} - \mu\mathbf{\Xi}\mathbf{A}^{\mathrm{T}}\mathbf{A})$ is bounded above by 1, and the step-size $0 < \mu \leq 1/\|\mathbf{\Xi}\mathbf{A}\mathbf{A}^{\mathrm{T}}\|$. Taking $\mathbf{\Xi} = (\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}$ ($\mathbf{A}^{\mathrm{T}}\mathbf{A}$ is non-singular and a kind of Newton's method ensues) yields the closed-form
$$\tilde{\boldsymbol{\alpha}} = \mathrm{Thresh}_{\lambda}\left(\mathbf{A}^{+}\boldsymbol{\beta}\right) \tag{6.27}$$
where $\mathrm{Thresh}_{\lambda}$ is a thresholding operator (hard for $p = 0$ and soft for $p = 1$).

- If $\boldsymbol{\alpha}$ is fixed, and since $\boldsymbol{\alpha}$ is full row-rank, the mixing matrix $\mathbf{A}$ is given by the least-squares estimate:
$$\tilde{\mathbf{A}} = \boldsymbol{\beta}\boldsymbol{\alpha}^{\mathrm{T}}\left(\boldsymbol{\alpha}\boldsymbol{\alpha}^{\mathrm{T}}\right)^{-1} = \boldsymbol{\beta}\boldsymbol{\alpha}^{+} \tag{6.28}$$
and the columns of $\tilde{\mathbf{A}}$ are then normalized.

Note that the latter two-step estimation scheme has a flavor of the alternating sparse coding/dictionary learning algorithm presented by (Aharon et al. 2006; Peyré et al. 2007) in a different framework.

This two-stage iterative process leads to the accelerated version of GMCA summarized in Algorithm 11.

In the same vein as in Section 6.5.3, the coarse-to-fine process is also at the heart of this fast version of GMCA with the threshold that decreases with increasing iteration count. This again brings robustness to noise and initialization.

Algorithm 11 Fast GMCA algorithm.

Task: Sparse Blind Source Separation.
Parameters: The data $\mathbf{Y}$, the dictionary $\mathbf{\Phi} = [\mathbf{\Phi}_1 \cdots \mathbf{\Phi}_K]$, number of iterations N_{iter}, number of sources N_s and channels N_c, stopping threshold $\lambda_{\min}$, threshold update schedule.
Initialization:

- $\boldsymbol{\alpha}^{(0)} = 0$, $\mathbf{A}^{(0)}$ a random matrix.
- Apply the MCA Algorithm 8 with $\mathbf{\Phi}$ to each data channel y_i to get $\boldsymbol{\beta}$.
- Set threshold $\lambda_0 = \max_{i,l} |\boldsymbol{\beta}[i,l]|$.

Main iteration:
for $t = 1$ **to** N_{iter} **do**

- Update the coefficients $\boldsymbol{\alpha}$: $\boldsymbol{\alpha}^{(t+1)} = \text{Thresh}_{\lambda_t}\left(\mathbf{A}^{(t)^+}\boldsymbol{\beta}\right)$.
- Update the mixing matrix $\mathbf{A}$: $\mathbf{A}^{(t+1)} = \boldsymbol{\beta}\boldsymbol{\alpha}^{(t+1)^+}$, normalize columns to a unit ℓ_2 norm.
- Update the threshold λ_t according to the given schedule.

if $\lambda_t \leq \lambda_{\min}$ **then** stop.

Reconstruct the sources: $\tilde{s}_i = \sum_{k=1}^{K} \mathbf{\Phi}_k \alpha_{i,k}^{(N_{\text{iter}})}, i = 1, \cdots, N_s$.
Output: Estimated sources $\left(\tilde{s}_i\right)_{i=1,\cdots,N_s}$ and mixing matrix $\mathbf{A}^{(N_{\text{iter}})}$.

Chapter 7

Statistics on the Sphere and Non-Gaussianities Detection

7.1 Introduction

The search for non-Gaussian signatures in the cosmic microwave background (CMB) temperature fluctuation maps furnished by MAP[1] (Komatsu et al. 2003), and to be furnished by PLANCK[2], is of great interest for cosmologists. Indeed, the non-Gaussian signatures in the CMB can be related to very fundamental questions such as the global topology of the universe (Riazuelo et al. 2002), superstring theory, topological defects such as cosmic strings (Bouchet et al. 1988), and multi-field inflation (Bernardeau and Uzan 2002). The non-Gaussian signatures can, however, have a different but still cosmological origin. They can be associated with the Sunyaev-Zel'dovich (SZ) effect (Sunyaev and Zeldovich 1980) (inverse Compton effect) of the hot and ionized intra-cluster gas of galaxy clusters (Aghanim and Forni 1999; Cooray 2001), with the gravitational lensing by large scale structures (Bernardeau et al. 2003), or with the reionization of the universe (Aghanim and Forni 1999; Castro 2003). They may also be simply due to foreground emission (Jewell 2001), or to non-Gaussian instrumental noise and systematics (Banday et al. 2000).

All these sources of non-Gaussian signatures might have different origins and thus different statistical and morphological characteristics. It is therefore not surprising that a large number of studies have recently been devoted to the subject of the detection of non-Gaussian signatures. Many approaches have been investigated: Minkowski functionals and the morphological statistics (Novikov et al. 2000; Shandarin 2002), the bispectrum (3-point estimator in the Fourier domain) (Bromley and Tegmark 1999; Verde et al. 2000; Phillips and Kogut 2001), the trispectrum (4-point estimator in the Fourier domain) (Kunz et al. 2001), wavelet transforms (Aghanim and Forni 1999; Forni and Aghanim 1999; Hobson et al. 1999; Barreiro and Hobson 2001; Cayón et al. 2001b; Jewell 2001; Starck et al. 2004a), and the curvelet transform (Starck et al. 2004a). In (Aghanim et al. 2003; Starck et al. 2004a), it was shown that the wavelet transform was a very powerful tool to detect the non-Gaussian signatures. Indeed, the excess kurtosis (4th moment) of the wavelet coefficients outperformed all the other methods (when the signal is characterized

[1] http://map.gsfc.nasa.gov/

[2] http://astro.estec.esa.nl/SA-general/Projects/Planck/

by a non-zero 4th moment). Based on kurtosis of wavelet coefficients, recent studies have reported non-Gaussian signatures in the WMAP data (Vielva et al. 2004; Mukherjee and Wang 2004; Cruz et al. 2005). The excess kurtosis is a widely used statistic, based on the 4th moment. The kurtosis measures a kind of departure of X from Gaussianity. The non-Gaussianty detector consists of first applying a multiscale transform (e.g., wavelet, or curvelet), and then calculating at each scale the kurtosis. In practice, missing data and instrumental effects may create an artificial kurtosis and it is very important to produce realistic simulations which present the same caracteristics as the observated data (e.g., missing data, noise, etc.). Then the kurtosis obtained from the data is compared to the kurtosis level expected from the simulations.

Finally, a major issue of the non-Gaussian studies in CMB remains our ability to disentangle all the sources of non-Gaussianity from one another. Recent progress has been made on the discrimination between different possible origins of non-Gaussianity. Namely, it was possible to separate the non-Gaussian signatures associated with topological defects (cosmic strings) from those due to the Doppler effect of moving clusters of galaxies (both dominated by a Gaussian CMB field) by combining the excess kurtosis derived from both the wavelet and the curvelet transforms (Starck et al. 2004a).

The wavelet transform is suited to spherical-like sources of non-Gaussianity, and a curvelet transform is suited to structures representing sharp and elongated structures such as cosmic strings. Each provides an adapted non-Gaussian estimator, namely the normalised mean excess kurtosis. The combination of these transforms through the product of the normalized mean excess kurtosis of wavelet transforms by normalized mean excess kurtosis of curvelet transforms highlights the presence of the cosmic strings in a mixture CMB+SZ+CS. Such a combination gives information about the nature of the non-Gaussian signals. The sensitivity of each transform to a particular shape makes it a very strong discriminating tool (Starck et al. 2004a; Jin et al. 2005).

In order to illustrate this, we show in Fig. 7.1 a set of simulated maps. Primary CMB, kinetic SZ and cosmic string maps are shown respectively in Fig. 7.1 top left, top right and bottom left. The "simulated observed map", containing the three previous components, is displayed in Fig. 7.1 bottom right. The primary CMB anisotropies dominate all the signals except at very high multipoles (very small angular scales). The wavelet function is overplotted on the kinetic Sunyaev-Zel'dovich map and the curvelet function is overplotted on cosmic string map.

CMB data are different from other astronomical data sets in the sense that they are not sparse (typical sparse data are stars or/and galaxies on top of a smooth background). After a component separation processing (see chapter 6), the CMB data are not completely free of contaminations. Point sources still need to be detected and removed. Once we believe the data are clean enough, we want to check if the distribution of CMB temperature fluctuations is Gaussian by using robust statistical Gaussianity tests.

7.2 Point Sources on a Gaussian Background

Several methods have been proposed in the last years for point source detection in the CMB such as the the Mexican Hat wavelet (Cayón et al. 2000, 2001b), the pseudo-filter (Sanz et al. 2001), or the biparametric scale-adaptive filter (López-Caniego et al. 2005). A simple and robust technique, which maximizes the signal-to-noise ratio is the Matched

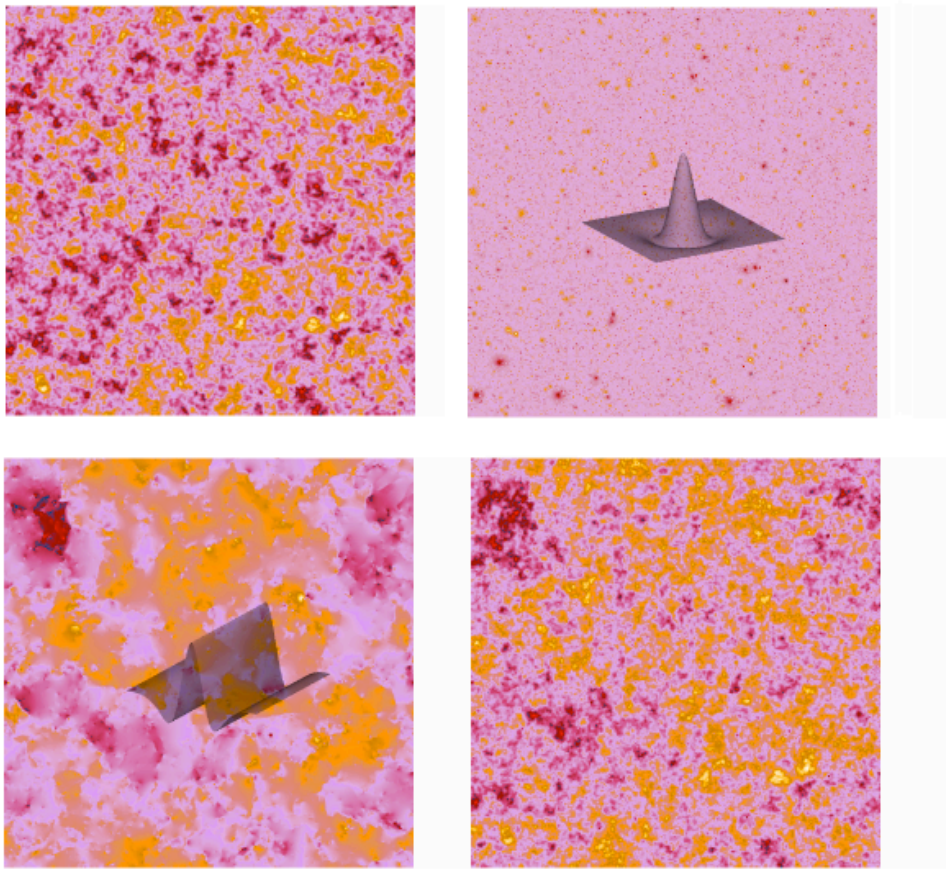


Figure 7.1: Top, primary Cosmic Microwave Background anisotropies (left) and kinetic Sunyaev-Zel'dovich fluctuations (right). Bottom, cosmic string simulated map (left) and simulated observation containing the previous three components (right). The wavelet function is overplotted on the Sunyaev-Zel'dovich map and the curvelet function is overplotted on cosmic string map.

Filter (Vio et al. 2002). Assuming an isotropic point spread function (PSF) with known power sprectum $\tau(q)$ and the CMB with power spectrum $P(q)$, the Matched Filter is (Vio et al. 2002):

$$\widehat{\psi}_{MF}(q) = \frac{1}{2\pi\alpha}\,\frac{\tau(q)}{P(q)}, \qquad \alpha \equiv \int_0^{+\infty} q\frac{\tau^2}{P}\,dq \tag{7.1}$$

with minimum variance

$$\sigma^2 = \frac{1}{2\pi\alpha} \tag{7.2}$$

If the PSF is unknown (or space-variant), the Mexican Hat wavelet may be a good alternative. It consists of convolving the data with the wavelet function $\psi_{a,b}(x) = \psi(\frac{x-b}{a})$, where $\psi(x) = \frac{1}{\sqrt{2\pi}}(1-x^2)e^{-x^2/2}$. a is the scale parameter and b the position parameter. A fast implementation is obtained by using the Fourier transform to perform the convolution products ($\widehat{\psi}_a(q) = \frac{2}{\sqrt{\pi}}(qa)^2 e^{-\frac{1}{2}(qa)^2}$) (López-Caniego et al. 2005).

7.3 Detecting Faint Non-Gaussian Signals Superposed on a Gaussian Signal

The superposition of a non-Gaussian signal with a Gaussian signal can be modeled as $Y = N + G$, where Y is the observed image, N is the non-Gaussian component and G is the Gaussian component. We are interested in using transform coefficients to test whether $N \equiv 0$ or not.

7.3.1 Hypothesis Testing and Likelihood Ratio Test (LRT).

Transform coefficients of various kinds [Fourier, wavelet, curvelet, etc.] have been used for detecting non-Gaussian behavior in numerous studies. Let $X_1, X_2, \ldots, X_n$ be the transform coefficients of Y; we model these as

$$X_i = \sqrt{1-\lambda} \cdot z_i + \sqrt{\lambda} \cdot w_i, \qquad 0 < \lambda < 1 \tag{7.3}$$

where $\lambda > 0$ is a parameter, $z_i \overset{iid}{\sim} N(0,1)$ are the transform coefficients of the Gaussian component G, $w_i \overset{iid}{\sim} W$ are the transform coefficients of the non-Gaussian component N, and W is some unknown symmetrical distribution. Here without loss of generality, we assume the standard deviation for both z_i and w_i are 1.

Phrased in statistical terms, the problem of detecting the existence of a non-Gaussian component is equivalent to discriminating between the hypotheses:

$$H_0 : \quad X_i = z_i \tag{7.4}$$

$$H_1 : X_i = \sqrt{1-\lambda} \cdot z_i + \sqrt{\lambda} \cdot w_i, \qquad 0 < \lambda < 1 \tag{7.5}$$

and $N \equiv 0$ is equivalent to $\lambda \equiv 0$. We call H_0 the *null hypothesis* H_0, and H_1 the *alternative hypothesis*.

When both W and λ are known, then the optimal test for Problem (7.4) - (7.4) is simply the Neyman-Pearson Likelihood ratio test (LRT), (Lehmann 1986, Page 74). The size of $\lambda = \lambda_n$ for which reliable discrimination between H_0 and H_1 is possible can be derived using asymptotics. If we assume that the tail probability of W decays algebraically,

$$\lim_{x \to \infty} x^{\alpha} P\{|W| > x\} = C_{\alpha}, \qquad C_{\alpha} \text{ is a constant} \tag{7.6}$$

(we say W has a power-law tail), and we calibrate λ to decay with n, so that increasing amounts of data are offset by increasingly hard challenges:

$$\lambda = \lambda_n = n^{-r} \tag{7.7}$$

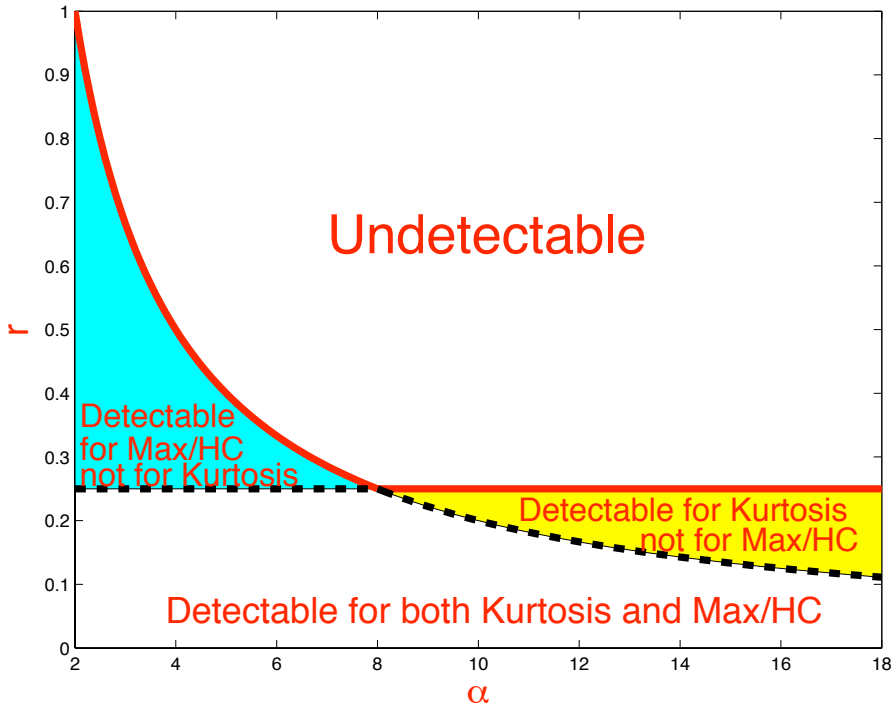


Figure 7.2: Detection Boundary in the $\alpha - r$ plane. The solid curve is the detection boundary of LRT, above which is not possible to detect, and below which it is possible to reliably detect, the dotted line segment and solid line segment together is the detection boundary for Kurtosis, the dotted curve and the solid curve together is the detection boundary of Max/HC. Right panel: detectable regions for Kurtosis, Max/HC.

then there is a *threshold effect* for the detection problem (7.4) - (7.4). In fact, define:

$$\rho_1^*(\alpha) = \begin{cases} 2/\alpha, & \alpha \le 8 \\ 1/4, & \alpha > 8 \end{cases} \tag{7.8}$$

then as $n \to \infty$, LRT is able to reliably detect for large n when $r < \rho_1^*(\alpha)$, and is unable to detect when $r > \rho_1^*(\alpha)$; this is proved in (Donoho and Jin 2004b). Since LRT is optimal, it is not possible for any statistic to reliably detect when $r > \rho_1^*(\alpha)$. We call the curve $r = \rho_1^*(\alpha)$ in the α-r plane the *detection boundary*; see Figure 7.2.

In fact, when $r < 1/4$, asymptotically LRT is able to reliably detect whenever W has a finite 8-th moment, even without the assumption that W has a power-law tail. Of course, the case that W has an infinite 8-th moment is more complicated, but if W has a power-law tail, then LRT is also able to reliably detect if $r < 2/\alpha$.

Despite its optimality, LRT is not a practical procedure. To apply LRT, one needs to specify the value of λ and the distribution of W, which seems unlikely to be available. We need non-parametric detectors, which can be implemented without any knowledge of λ or W, and depend on X_i's only. In the next section, we are going to introduce three non-parametric detectors: excess kurtosis, Max and Higher Criticism (HC).

7.4 Kurtosis, HC from Wavelet and Curvelet Coefficients

7.4.1 Kurtosis

For a statistic T_n, the p-value is the probability of seeing equally extreme results under the null hypothesis:

$$p = P_{H_0}\{T_n \geq t_n(X_1, X_2, \ldots, X_n)\}$$

here P_{H_0} refers to probability under H_0, and $t_n(X_1, X_2, \ldots, X_n)$ is the observed value of statistic T_n. Notice that the smaller the p-value, the stronger the evidence against the null hypothesis. A natural decision rule based on p-values rejects the null when $p < \alpha$ for some selected level α, and a convenient choice is $\alpha = 5\%$. When the null hypothesis is indeed true, the p-values for any statistic are distributed as uniform $U(0, 1)$. This implies that the p-values provide a common scale for comparing different statistics.

We now introduce two statistics for comparison.

Excess Kurtosis (κ_n). Excess kurtosis is a widely used statistic, based on the 4-th moment. For any (symmetrical) random variable X, the kurtosis is:

$$\kappa(X) = \frac{EX^4}{(EX^2)^2} - 3$$

The kurtosis measures a kind of departure of X from Gaussianity, as $\kappa(z) = 0$. Empirically, given n realizations of X, the excess kurtosis statistic is defined as:

$$\kappa_n(X_1, X_2, \ldots, X_n) = \sqrt{\frac{n}{24}}\left[\frac{\frac{1}{n}\sum_i X_i^4}{(\frac{1}{n}\sum_i X_i^2)^2} - 3\right] \tag{7.9}$$

When the null is true, the excess kurtosis statistic is asymptotically normal:

$$\kappa_n(X_1, X_2, \ldots, X_n) \rightarrow_w N(0, 1), \qquad n \rightarrow \infty$$

thus for large n, the p-value of the excess kurtosis is approximately:

$$\tilde{p} = \bar{\Phi}^{-1}(\kappa_n(X_1, X_2, \ldots, X_n))$$

where $\bar{\Phi}(\cdot)$ is the survival function (upper tail probability) of $N(0, 1)$.

It is proved in (Donoho and Jin 2004b) that the excess kurtosis is asymptotically optimal for the hypothesis testing of (7.4) - (7.4) if

$$E[W^8] < \infty$$

However, when $E[W^8] = \infty$, even though kurtosis is well-defined ($E[W^4] < \infty$), there are situations in which LRT is able to reliably detect but excess kurtosis completely fails. In fact, by assuming (7.6) - (7.7) with an $\alpha < 8$, if (α, r) falls into the blue region of Figure 7.2, then LRT is able to reliably detect, however, excess kurtosis completely fails. This shows that in such cases, excess kurtosis is not optimal; see (Donoho and Jin 2004b).

7.4.2 Max

The largest (absolute) observation is a classical and frequently-used non-parametric statistic:

$$M_n = \vee(|X_1|, |X_2|, \ldots, |X_n|)$$

under the null hypothesis,

$$M_n \approx \sqrt{2 \log n}$$

and moreover, by normalizing M_n with constants c_n and d_n, the resulting statistic converges to the Gumbel distribution E_v, whose cdf is $e^{-e^{-x}}$:

$$\frac{M_n - c_n}{d_n} \rightarrow_w E_v$$

where approximately

$$d_n = \frac{\sqrt{6} S_n}{\pi}, \qquad c_n = \bar{X} - 0.5772 d_n$$

here $\bar{X}$ and S_n are the sample mean and sample standard deviation of $\{X_i\}_{i=1}^n$ respectively. Thus a good approximation of the p-value for M_n is:

$$\tilde{p} = \exp(-\exp(-\frac{M_n - c_n}{d_n}))$$

We have tried the above experiment for $n = 244^2$, and found that taking $c_n = 4.2627$, $d_n = 0.2125$ gives a good approximation.

Assuming (7.6) - (7.7) and $\alpha < 8$, or $\lambda = n^{-r}$ and that W has a power-law tail with $\alpha < 8$, it is proved in (Donoho and Jin 2004b) that Max is optimal for hypothesis testing (7.4) - (7.4). Recall if we further assume $\frac{1}{4} < r < \frac{2}{\alpha}$, then asymptotically, excess kurtosis completely fails; however, Max is able to reliably detect and is competitive to LRT.

On the other hand, recall that excess kurtosis is optimal for the case $\alpha > 8$. In comparison, in this case, Max is not optimal. In fact, if we further assume $\frac{2}{\alpha} < r < \frac{1}{4}$, then excess kurtosis is able to reliably detect, but Max will completely fail.

In Figure 7.2, we compared the detectable regions of the excess kurtosis and Max in the α-r plane.

7.4.3 Higher Criticism

The Higher Criticism statistic (HC), was proposed in (Donoho and Jin 2004a). To define HC first we convert the individual X_i's into p-values for individual z-tests. Let $p_i = P\{N(0,1) > X_i\}$ be the i^{th} p-value, and let $p_{(i)}$ denote the p-values *sorted in increasing order*; the Higher Criticism statistic is defined as:

$$HC_n^* = \max_i \left| \sqrt{n}[i/n - p_{(i)}] / \sqrt{p_{(i)}(1 - p_{(i)})} \right|$$

or in a modified form:

$$HC_n^+ = \max_{\{i:\, 1/n \le p_{(i)} \le 1 - 1/n\}} \left| \sqrt{n}[i/n - p_{(i)}] / \sqrt{p_{(i)}(1 - p_{(i)})} \right|$$

we let HC_n refer either to HC_n^* or HC_n^+ whenever there is no confusion. The above definition is slightly different from (Donoho and Jin 2004a), but the ideas are essentially the same.

With an appropriate normalization sequence:

$$a_n = \sqrt{2 \log \log n}, \qquad b_n = 2 \log \log n + 0.5 \log \log \log n - 0.5 \log(4\pi)$$

the distribution of HC_n converges to the Gumbel distribution E_v^4, whose cdf is $\exp(-4\exp(-x))$, (Shorack and Wellner 1986):

$$a_n HC_n - b_n \rightarrow_w E_v^4$$

so the p-values of HC_n are approximately:

$$\exp(-4\exp(-[a_n HC_n - b_n])) \tag{7.10}$$

For moderately large n, in general, the approximation in (7.10) is accurate for the HC_n^+, but not for HC_n^*.

A brief remark comparing Max and HC. Max only takes into account the few largest observations, HC takes into account those outliers, but also moderate large observations. As a result, in general HC is better than Max, especially when we have unusually many moderately large observations. However, when the actual evidence lies in the middle of the distribution both HC and Max will be very weak.

7.5 Experiments

Fig. 7.3 shows, top left and right, two images with respectively Gaussians and lines. We have created a set of simulated images by adding a Gaussian white noise with different standard deviations to these two images. The Signal to Noise Ratio (SNR) varies between 0 and 1. For the image with lines, the SNR is defined as the pixel values along the lines divided by the noise standard deviation, and for the image with Gaussians, the SNR is defined as the maximum of the Gaussians divided by the noise standard deviation. Fig. 7.3 shows, bottom left and right, the two noisy images with a SNR equal to 1. Hence, for each SNR value, we have thirty realizations of the noise, and we have calculated the kurtosis at the different scales of both the curvelet and the wavelet coefficients. These kurtosis values were normalized by the standard deviation of the kurtosis obtained from the wavelet and the curvelet transform of thirty Gaussian white noise realizations. Finally we kept for each SNR the maximum normalized kurtosis along the scales. Fig. 7.4 left (resp. right) shows the normalized kurtosis values using the wavelet transform (resp. the curvelet transform) for the two images (i.e. lines and Gaussians) versus the SNR. Continuous error bars correspond to 1σ level and dashed error bars correspond to 2σ level. We can clearly see that the detection power of the wavevet transform is much larger than the detection power of the curvelet transform for detecting non-Gaussianities due to isotropic features, while curvelets are more powerful than wavelets for detecting anisotropic features.

7.6 Conclusions

The kurtosis of the wavelet coefficients is very often used in astronomy for the detection of non-Gaussianities in the CMB. It has been shown (Starck et al. 2004a) that it is also

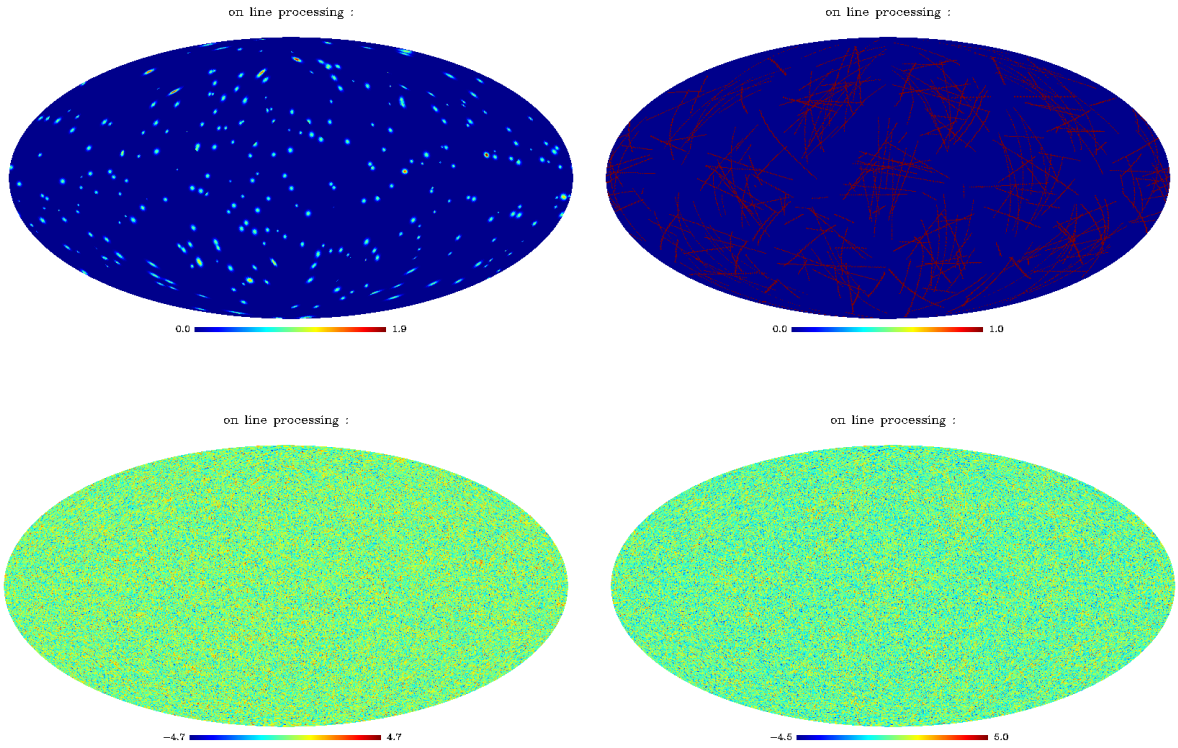


Figure 7.3: Top, image with Gaussians and image with lines. Bottom, same images but with an additional Gaussian noise. The SNR is equal to 1.

possible to separate the non-Gaussian signatures associated with cosmic strings from those due to SZ effect by combining the excess kurtosis derived from these both the curvelet and the wavelet transform. It has been shown that kurtosis is asymptotically optimal in the class of weakly dependent symmetric non-Gaussian contamination with finite 8-th moments, while HC and MAX are asymptotically optimal in the class of weakly dependent symmetric non-Gaussian contamination with infinite 8-th moment (Jin et al. 2005). Hence depending on the nature of the non-Gaussianity, a statitic is better than another one. This is a motivation for using several statistics rather than a single one for analysing CMB data. The case of the detection of cosmic string contaminations has been studied on simulated maps, and it has been shown that kurtosis outperforms clearly Max/HC (Jin et al. 2005).

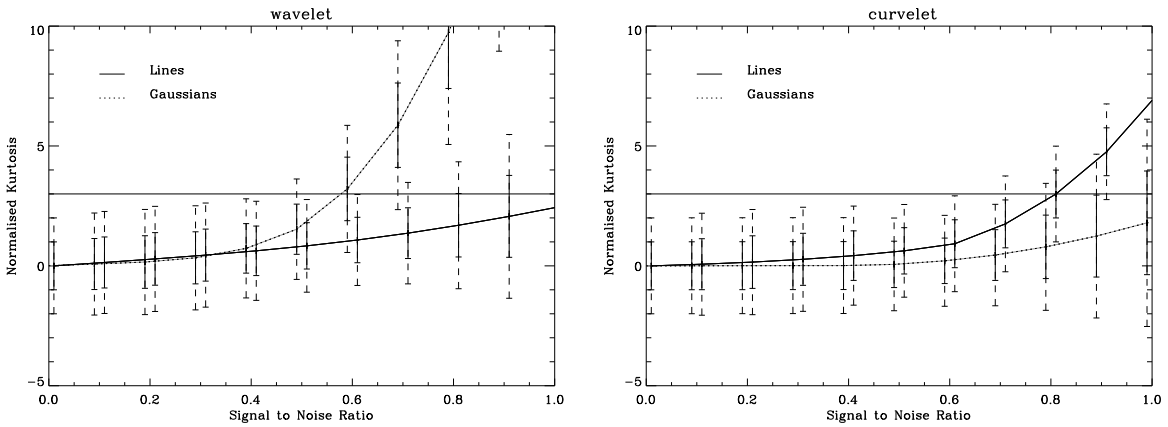


Figure 7.4: Normalised kurtosis value versus the SNR for the wavelet coefficients (left) and the curvelet coefficients (right). The continuous error bars correspond to one σ and the dashed error bars correspond to 2σ.

Chapter 8

IDL Routines

8.1 General functions for spherical maps

8.1.1 Reading a spherical map from a file : mrs_read

Read a spherical map, either in HEALPIX or in GLESP format.

USAGE: map = mrs_read(file)

where

- *file* : Input string, name of the file to be read. The pathname can be included in the string, by default 'file.fits' is equivalemt to './file.fits'
- *map* : Output IDL array of Healpix map or Glesp image IDL structure, map read. For a Healpix map, the map is setted to the NESTED format after reading.

Example:

- healmap = mrs_read('my_file_healpix.fits')
  Read the map stored into the file 'my_file_healpix.fits' and load it into healmap.

8.1.2 Writing a spherical map into a file : mrs_write

Write a spherical map, either in HEALPIX with the NESTED format or in GLESP format.

USAGE: mrs_write, file, map, ring=ring

where

- *file* : Input string, name of the file to be writen. The pathname can be included in the string, by default 'file.fits' is equivalemt to './file.fits'
- *map* : Input IDL array of Healpix map or Glesp image IDL structure, to be writen. For a Healpix map, the map is assumed to be in the NESTED format.
- *ring* : scalar, if set convert the Healpix map data to the RING format for the writing.

Example:

- mrs_write, 'my_file_healpix.fits', healpixdata, /ring
  Write the map healpixdata into the file 'my_file_healpix.fits' as a ring image.

8.1.3 Ploting a spherical map on the screen : mrs_tv

Visualization of a Healpix image (nested data representation) or a GLESP image.

USAGE: mrs_tv, Data, graticule=graticule, gif=gif, png=png, title=title, colt=colt, nobar=nobar, Healpix=Healpix, PS=PS, log=log, min=min, max=max, pxsize=pxsize, big=big, x=x, pol=pol, units=units

where

- *Data* : Input IDL array of Healpix map or Glesp image IDL structure, map to be visualized.
- *min* : float, Data image is visualized with the new min set.
- *max* : float, Data image is visualized with the new max set.
- *log* : scalar, if set plot the image in log scale, Data must be positive.
- *graticule* : int, Mollview Healpix command graticule keyword, see mollview Healpix documentation for more details.
- *png* : string, if set write to the disk a PNG file with the filename given by the png keyword.
- *gif* : string, if set write to the disk a GIF file with the filename given by the gif keyword.
- *PS* : string, if set write to the disk a Postcript file with the filename given by the ps keyword.
- *title* : string, if set add the string as a title of the plot in the Healpix representation.
- *unit* : string, name of the image's unit to be plotted with the Look Up Table in the Healpix representation.
- *colt* : int, IDL Color table.
- *nobar* : scalar, if set do not plot the Look Up Table in the Healpix representation.
- *healpix* : scalar, if set then convert the GLESP image into a Healpix one, and use the Healpix representation for visualization. This keyword is not active for maps already in Healpix representation.
- *pxsize* : int, set the number of horizontal pixel on the plot (it will be the same in vertical), default value is 800.
- *big* : scalar, if set pxsize is setted to 1500.

- *pol* : int, keyword for plotting polarized map T,Q,U. Default value is 0 (no polarized map), see mollview Healpix documentation for more details.
- *x* : scalar, if set start interactive plot with the mapview prog, nside max=1024, data will be resized if greater.

8.1.4 Resizing a spherical map: mrs_resize

Resize map either in Healpix (nested format) or Glesp representation

USAGE: resize_map = mrs_resize(map, nside=nside, nx=nx, np=np, ViaAlm=ViaAlm)

where

- *map* : Input IDL array of Healpix map or Glesp image IDL structure to be transformed.
- *resize_map* : Output IDL array of Healpix map or GLESP image IDL structure. Healpix input map and output resized map are in nested format.
- *nside* : int, the new nside parameter of the output healpix resized map.
- *nx* : int = number of rings of the resized Glesp map.
- *np* : int = max number of pixel on a ring of the resized Glesp map.
- *ViaAlm* : scalar, if set use alm transform for the resizing, otherwise, use interpolation. Ignored if nside keyword value is lower than imag nside, Healpix only.

Examples:

- map2 = mrs_resize(map, nside = 256)
  resize an Healpix map.
- map2 = mrs_resize(map, nx = 512, np = 1024)
  resize a Glesp map.

8.1.5 Spliting of a spherical map into patches: mrs_split

Decompose an healpix map (nested format) into a cube of small patches.

USAGE: PatchTrans = mrs_split(Imag, frac=frac, nx=nx, exrec=exrec, SizePatchDegrees=SizePatchDegrees, PixelSizeParam=PixelSizeParam)

where

- *Imag* : Input IDL array of Healpix map to be to be decomposed into patches.
- *PatchTrans* : Output IDL structure with the following fields:

  - *NMaps* : long, number of patches.
  - *map_hd* : string, header of a small patch.
  - *Nx* : long, size of each patch in the x axis.
  - *Ny* : long, size of each patch in the y axis.
  - *Lon* : float array[NMaps], Longitude of each patches if the position (Nx/2,Ny/2) is the center of the patches.
  - *Lat* : float array[NMaps], Latitude of each patches if the position (Nx/2,Ny/2) is the center of the patches.
  - *Map* : float array[Nx, Nx, NMaps], cube of the patches.
  - *PixelSize* : float, pixel size in arc minute.
  - *MapSize* : float, map size in arc minute.
  - *Frac* : float, overlapping factor between patches.
  - *Nside* : long, nside parameter of the input imag.
- *Frac* : float, overlapping factor between patches. Default is 0.05
- *Nx* : long, size of each patch along both x and y axis. Default is automatically estimated.
- *SizePatchDegrees* : float, size (in degrees) of each patch. Default is 10 degrees.
- *exrec* : scalar, if set the pixel size is smaller in order to have an exact reconstruction using mrs_invsplit
- *PixelSizeParam* : float, if exrec is setted, the pixel size is multiplied by this factor (< 1) in order to smaller pixel for exact reconstruction.

Examples:

- Patch = mrs_split(map)
  Decompose an image into patches with default options.

8.1.6 Reconstruction of a spherical map from its patches: mrs_invsplit

Reconstruct a healpix map (nested format) from a cube of small patches obtained by the routine mrs_split.

USAGE: Map = mrs_invsplit(PatchTrans)

where

- *PatchTrans* : Output IDL structure with the following fields:
  - *NMaps* : long, number of patches.
  - *map_hd* : string, header of a small patch.
  - *Nx* : long, size of each patch in the x axis.

  - *Ny* : long, size of each patch in the y axis.
  - *Lon* : float array[NMaps], Longitude of each patches if the position (Nx/2,Ny/2) is the center of the patches.
  - *Lat* : float array[NMaps], Latitude of each patches if the position (Nx/2,Ny/2) is the center of the patches.
  - *Map* : float array[Nx, Nx, NMaps], cube of the patches.
  - *PixelSize* : float, pixel size in arc minute.
  - *MapSize* : float, map size in arc minute.
  - *Frac* : float, overlapping factor between patches.
  - *Nside* : long, nside parameter of the input imag.
- *Map* : Output IDL array of Healpix map reconstructed from the patches.

Examples:

- map = mrs_invsplit(Patch)
  Reconstruct an image from patches.

8.1.7 Transformations of a spherical map : mrs_trans

Compute a transform (wavelet, curvelet...) on a map on the sphere in the Healpix representation (nested data representation).

The transform can be:

1. Spherical Harmonic Transform.
2. Orthogonal wavelet (per face).
3. A trou undecimated wavelet transform (per face).
4. Pyramidal isotropic wavelet on the sphere.
5. Undecimated wavelet transform on the sphere (using a spline wavelet, Meyer wavelet or Needlet wavelet).
6. Ridgelet transform on the sphere.
7. Curvelet transform on the sphere.
8. Discrete Cosine Transform (per face).

The input image must be in the HEALPix pixel representation (nested data representation). The output is a IDL structure.

USAGE: mrs_trans, InImag, Trans, NbrScale=NbrScale, Alm=Alm, AT=AT, Cur=Cur, UWT=UWT, PyrWT=PyrWT, OWT=OWT, Rid=Rid, DCT=DCT, lmax=lmax, FirstBlockSize=FirstBlockSize, DifInSH=DifInSH, MeyerWave=MeyerWave, NeedletWave=NeedletWave, B_NeedletParam=B_NeedletParam, Overlap=Overlap

where

- *Imag* : Input 1D IDL array of healpix map in nested format. Input image to be transformed.
- *Trans* : Output IDL structure with the following fields:
  - *NbrScale* : int, number of scales.
  - *nside* : int, Healpix nside parameter.
  - *lmax* : int, maximum l value in the Spherical Harmonic Space.
  - *npix* : long, number of pixels of the input image.
  - *MeyerWave* : int, 1 if the keyword MeyerWave used, otherwise 0.
  - *NeedletWave*: int , 1 if the keyword NeedletWave is used, otherwise 0
  - *B_NeedletParam*: int, B_NeedletParam (default is 2).
  - *DifInSH* : int, 1 if the keyword DifInSH used, otherwise 0.
  - *pyrtrans* : int, 1 if a pyramidal decomposition has been applied, otherwise 0.
  - *DEC* : IDL structure, transformation result (depends on the chosen transform).
  - *TransChoice* : string, code of the chosen transform.
  - *TabCodeTransform* : string array, array of transforms codes. TabCodeTransform = ['T_ALM', 'T_OWT', 'T_AT', 'T_PyrWT', 'T_UWT', 'T_Ridgelet', 'T_CUR', 'T_DCT']
  - *TransName* : string, transform's name.
  - *TransTypeName* : string array, array of transforms names. TransTypeName = ['Alm','Bi-Orthogonal WT', 'A_Trous WT', 'Pyramidal WT', 'Undecimated WT', 'Ridgelet Transform', 'Curvelet', 'DCT']
- *NbrScale* : int, number of scales of the wavelet transforms.
- *Alm* : scalar, if set perform a Spherical Harmonic Transform. If no transform is selected, it will be the default transformation.
- *Cur* : scalar, if set perform a curvelet transform.
- *uwt* : scalar, if set perform an undecimated isotropic wavelet transform.
- *PyrWT* : scalar, if set perform a pyramidal isotropic wavelet transform.
- *owt* : scalar, if set perform a bi-orthogonal wavelet transform on each face.
- *AT* : scalar, if set perform a A trou undecimated wavelet transform on each face.
- *Rid* : scalar, if set perform a Ridgelet transform on the sphere.
- *DCT* : scalar, if set perform a Discrete Cosine Transform (per face).
- *Overlap* : int, if equal to 1 if blocks are overlapping, only used with curvelet transform.

- *FirstBlockSize* : int, block size in the ridgelet transform at the finest scale (default is 16), only used with curvelet transform.
- *lmax* : int, maximum l value in the Spherical Harmonic Space (for isoptropic wavelet transform only).
- *DifInSH* : Input keyword parameter. If set, the wavelet coefficients are computed as the difference between two resolutions in the spherical harmonics representation. Otherwise, the wavelet coefficients are computed as the difference between two resolutions in direct space. Only used with keyword uwt or PyrWT.
- *MeyerWave* : If set, use Meyer wavelets and set the keyword DifInSH. Only used with keyword uwt or PyrWT.

Examples:

- mrs_trans, Imag, WT, NbrScale=5, /uwt
  Compute the undecimated wavelet transform of the map Imag with five scales. The result is stored in WT.

8.1.8 Reconstructions of a spherical map : mrs_rec

Compute a inverse transform (wavelet, curvelet, ...) to get a map on the sphere in the Healpix representation (nested data representation) from its decomposition obtained by mrs_trans.

USAGE: mrs_rec, Trans, Rec

where

- *Trans* : Input IDL structure, see mrs_trans for more details.
- *Rec* : Output 1D IDL array of healpix map in nested format. Reconstructed image.

Examples:

- mrs_trans, Imag, WT, NbrScale=5, /uwt
  Compute the undecimated wavelet transform of the map Imag with five scales. The result is stored in WT.
- mrs_rec, WT, RecIma
  Reconstruct the image.

8.2 Spherical Harmonics

8.2.1 ALM transform : mrs_almtrans

Computes the spherical harmonic transform, using the Healpix representation (nested data representation by default) or the GLESP data representation.

USAGE: mrs_almtrans, Imag, Trans, lmax=lmax, ring=ring, tab=tab, complex=complex, psp=psp, norm=norm

where

- *Imag* : Input IDL array of Healpix map or Glesp image IDL structure. Input image to be transformed.
- *Trans* : Output IDL structure with the following fields:
  - *ALM* : array of the ALM coefficients

    ALM = fltarray[*,2] list of the real part (ALM[*,0]) and imaginary part (ALM[*,1]) of the ALM. This is the default storage.
    ALM = cfarr[*] list of the ALM in complex values format if the keyword complex is set.
    ALM = fltarray[NbrMaxM, NbrMaxL, 2] table of the real part (ALM[*,*,0]) and imaginary part (ALM[*,*,1]) of the ALM if the keyword tab is set.
    ALM = cfarr[NbrMaxM, NbrMaxL] table of the ALM in complex values format if the keywords complex and tab are both setted. By default, NbrMaxM = NbrMaxL = lmax+1
  - *complex_alm* : int, 0 (default value) if ALM array contains real and imaginary part separated. 1 if ALM is a complex array. 2 if ALM array contains the power spectrum and the phase.
  - *PixelType* : int, 0 for a Healpix input map and 1 for a GLESP input map.
  - *tab* : int, 0 for default ALM representation as a list (i.e. 1D IDL array) and 1 for 2D representation as a table (i.e. l for the first dimension and m for the second).
  - *nside* : int, Healpix nside parameter, only used in Healpix representation, otherwise, 0.
  - *lmax* : int, maximum l value in the Spherical Harmonic Space.
  - *npix* : long, number of pixels of the input image (12*nside*nside for Healpix).
  - *nx* : int = number of rings (Glesp parameter), otherwise, 0.
  - *np* : int = max number of pixel on a ring (Glesp parameter), otherwise, 0.
  - *x_sky* : float array with COS(THETA) for each ring (Glesp parameter), otherwise, 0.
  - *y_sky* : long array number of pixels/ring (Glesp parameter), otherwise, 0.
  - *TabNbrM* : int array[NbrMaxL], max number of m value for a given l, only used if keyword tab is set otherwise, 0.

  - *index* : long array, indicies of the ALM coefficients, used only if keyword tab is not set.
  - *NormVal* : float, normalization value applied to the alm coefficients (only if keyword norm used).
  - *norm* : int, 0 if no normalization has been aplied, else 1.

- *Lmax* : int, Number of spherical harmonics computed in the decomposition. For a Healpix map, default is 3*nside and should be between 2*nside and 4*nside. For a GLESP map, default is: min([Imag.nx/2, Imag.np/4]).
- *ring* : scalar, if set the input Healpix map is supposed to be in RING representation, ignored with GLESP images.
- *Tab* : scalar, if set, ALM coefficients in Trans.alm are stored in a 2D array: Trans.alm[m,l] where m = 0..Trans.TabNbrM[l]-1 and l = 0..lmax-1
- *complex* : scalar, if set Trans.alm will contain complex values instead of the real and imaginary parts.
- *psp* : scalar, if set Trans.alm will contain the power spectrum and the phase instead of the real and imaginary parts. This is ignored if keyword complex is also set.
- *norm* : scalar, if set, a normalization is performed to the alm coefficients.

Examples:

- mrs_almtrans, Imag, Output
  Compute the spherical harmonics transform of an image, the result is stored in Output.

8.2.2 ALM inverse transform : mrs_almrec

Computes the inverse spherical harmonic transform, using the Healpix representation (nested data representation by default) or the GLESP representation.

USAGE: mrs_almrec, Trans, Imag, to_healpix=to_healpix, to_glesp=to_glesp, pin=pin, nx=nx, np=np, pixel_window=pixel_window

where

- *Trans* : Input IDL structure of ALM coefficients, see mrs_almtrans above for details.
- *Imag* : Output IDL array of Healpix map (if Trans.pixeltype=0) or Glesp image IDL structure (if Trans.pixeltype=1). Image reconstructed, Healpix images are only reconstructed in nested representation.
- *to_healpix* : int, if to_healpix=1, the reconstructed image will be in Healpix format instead of GLESP. If Trans.pixeltype=0 (already healpix), generates an error. If to_healpix ¿ 1, force the output to be in Healpix format with nside = to_healpix (must be a valid value)

- *to_glesp* : scalar, if set the reconstructed image will be in GLESP format instead of healpix. If Trans.pixeltype=1 (already GLESP), generates an error.
- *nx* : int, new number of rings for the reconstructed GLESP image (should be equal to 4*nside), nx must be larger than 2*lmax.
- *np* : int, new number of pixels for the central ring of the reconstructed GLESP image (should be equal to 8*nside) np must be larger than 4*lmax.
- *pixel_window* : scalar, if set the image is convolved by the healpix pixel window (only for Healpix map).
- *pin* : int, GLESP parameter -1 for equal aera (default), 0 for iso lat/lon (Theta-Phi image), 1 for triangular ...

Examples:

- mrs_almtrans, Imag, Output
  Compute the spherical harmonics transform of an image, the result is stored in Output.
- mrs_almrec, Output, Rec
  Reconstruct the image.

8.2.3 Power spectrum exctraction from ALM : mrs_alm2spec

Compute the power spectrum $C(l)$ from the $A_{l,m}$ coefficients.

$$C(l) = \frac{1}{2l+1} \sum_{m=-l}^{m=l} \mid A_{l,m} \mid^2 \tag{8.1}$$

USAGE: spec = mrs_alm2spec(ALM, StdPS=StdPS)

where

- *ALM* : Input IDL structure of ALM coefficients, see mrs_almtrans above for details.
- *spec* : Output 1D IDL float array[ALM.lmax+1]: Power spectrum $C(l)$ extracted from the ALM coefficients.
- *StdPS* : 1D IDL float array[ALM.lmax + 1]: estimated standard deviation of the C_l coefficients.

Examples:

- mrs_almtrans, Imag, Output
  Compute the spherical harmonics transform of an image, the result is stored in Output.
- spec = mrs_alm2spec(Output, StdPS=StdPS)
  Compute the power spectrum of the image and it's associated standard deviation.

8.2.4 Power spectrum exctraction from an image : mrs_powspec

Computes the power spectrum of a map, using the Healpix representation (nested data representation by default) or the GLESP representation. If the keyword log is set, it is the log-power spectrum which is returned. If the global variable DEF_NORM_POWSPEC equal to 1 or if the keyword /set_norm is set, then a normalization is performed, so that a Gaussian randomn noise with variance equal to 1 has a power spectrum equal to 1.

USAGE: spec = mrs_powspec(Imag, plot=plot, lplot=lplot, log=log, IndL=IndL, PowSpecIma=PowSpecIma, StdPS=StdPS, set_norm=set_norm, nonorm=nonorm, NormVal=NormVal, alm=alm, lmax=lmax)

where

- *Imag* : Input IDL array of Healpix map or Glesp image IDL structure. Input image whose power spectrum will be extracted.
- *spec* : Output 1D IDL float array[Lmax+1]: Power spectrum $C(l)$ extracted from the map.
- *plot* : scalar, if set the power spectrum is plotted.
- *lplot* : scalar, if set the power spectrum multiplied by l(l+1) is plotted.
- *log* : scalar, if set the log power spectrum is calculated instead of the power spectrum.
- *Lmax* : int, number of spherical harmonics computed in the decomposition and size of the computed spectrum (Lmax+1). For a Healpix map, default is 3*nside and should be between 2*nside and 4*nside. For a GLESP map, default is: min([Imag.nx/2, Imag.np/4]).
- *nonorm* : scalar, if set no normalisation is applied on the ALM computed.
- *set_norm* : scalar, if set a l2 normalization if perform, so a Gaussian randomn noise with variance equal to 1 will have a power spectrum equal to 1.
- *IndL* : Optional output 1D IDL int array [Lmax+1]. Contains the multiplicative l(l+1) values.
- *StdPS* : 1D IDL float array[Lmax+1]: estimated standard deviation of the C_l coefficients.
- *NormVal* : float, normalization value applied to the alm coefficients.
- *alm* : IDL structure of ALM coefficients, result of the alm transform of the input image (see mrs_almtrans) with options lmax, /tab, /psp, norm=? .
- *PowSpecIma* : 2D IDL float array: Power spectrum of the input data for l and m.

Examples:

- P = mrs_powspec(Imag)
  Compute the power spectrum of the image.

8.2.5 Wiener filtering of a map in spherical harmonics space : mrs_wiener

Perform wiener filtering of a map, in the spherical harmonic space, using the Healpix representation (nested data representation by default) or the GLESP representation. $Alm\_output = Alm\_input * WienerFilter$ where:

$$WienerFilter = \frac{P^* S}{\mid PP^* \mid^2 S + N}$$

with

- P, instrumental beam (i.e. PSF, by default P = 1)
- S, a priori Signal Power Spectrum (default, power spectrum of the data)
- N, Noise power spectrum

If the keyword ALM is set (it should be a structure (see mrs_almtrans) containg the ALM coefficients of the data, then the first parameter is not used and the alm are not calculated in this routine. The Wiener is optimal in the sens of the Least mean square error of the reconstructed map. The power spectrum of the reconstructed map is biased (i.e. $PowSpectrum(WienerMap) = \frac{PowSpectrum(RealMap)}{(1+N/(P^2 S))}$). If the Cole keyword is set, then the Wiener filtering is replaced by the Cole filtering and the power spectrum of the reconstructed map is unbiased, but the estimation is not optimal anymore for the east mean square error criterion. The Cole filter is:

$$ColeFilter = \Big(\frac{S}{(\mid PP^* \mid^2 S + N)} \Big)^{\frac{1}{2}}$$

If the keyword DataPrior is set, then the vector given by SignalPrior corresponds to an a priori on the power spectrum of the signal multiplied by P^2

USAGE: mrs_wiener, Imag, NoiseSpectrum, Recons, alm=alm, lmax=lmax, SignalPrior=SignalPrior, Spec1D=Spec1D, WienerFilter=WienerFilter, Psf=Psf, Cole=Cole, bias=bias

where

- *Imag* : Input IDL array of Healpix map or Glesp image IDL structure, image to be denoised.
- *NoiseSpectrum* : float or IDL 1D float array: variance or power spectrum of the noise.
- *Recons* : Output IDL array of healpix map or Glesp image IDL structure, denoised image.
- *SignalPrior* : Input IDL 1D float array, power spectrum of the expected signal. By default, it is estimated from the data.
- *DataPrior* : scalar, if set the vector given by SignalPrior corresponds to an a priori on the power spectrum of the signal multiplied by P^2

- *Psf* : Input IDL 1D float array, instrumental beam i.e. PSF[l] = Spherical Harmonics $a_{l,0} = \ldots = a_{l,l+1}$ of the intrumental beam (i.e. Point Spread Funciton).
- *alm* : input/output ALM structure (see mrs_almtrans). If this keyword is set, the input image is not used, and the ALM given by this keyword are used instead. The denoised ALM are stored in the structure. **ALM MUST have been calculated with the keywords "/tab" and "/norm" and not "/psp" or "/complex"**
- *Spec1D* : Output IDL 1D float array, estimated power spectrum of the denoised image.
- *WienerFilter* : Output IDL 1D float array, Wiener filter.
- *Lmax* : int, maximum l used in the calculation of the ALM. This kewyord is not used if the keyword ALM is set.
- *Cole* : scalar, if set the Wiener filter is replaced by the Cole filter.
- *bias* : Output float or IDL 1D float array, estimated bias on the power spectrum.

8.2.6 Iterative Wiener filtering of a map: mrs_itwiener

Perform wiener filtering of a spherical map when the noise is non-stationary using the Healpix representation (nested data representation by default).

If the keyword Frg is set, it also estimates the contribution coming from extra foreground residuals.

USAGE: mrs_itwiener, Imag, Powspec, Recons, Noise=Noise, fNoise=fNoise, niter=niter,cole=cole,StartWiener=StartWiener, NbrScale=NbrScale, BS=BS,Frg=Frg,RMSmoothing=RMSmoothing,tol=tol,verbose=verbose

where

- *Imag* : Input IDL array of Healpix map, image to be denoised.
- *Powspec* : float or IDL 1D float array: theoretical power spectrum of the image.
- *Recons* : Output IDL array of healpix denoised image.
- *Noise* :Input Noise realization from which the noise statistics are measured.
- *fNoise* : Output Noise realization (apply the same filtering on the data and on the input noise realization).
- *niter* : scalar, number of iterations. Default is 100.
- *Cole* : scalar, if set the Wiener filter is replaced by the Cole filter.
- *StartWiener* : if set, the algorithm is initialized with the global Wiener filter (i.e. in the spherical harmonics).
- *NbrScale* scalar: number of wavelet scales used for analysis. Default is 4.

- *BS* scalar: minimal patch size used to compute the noise variance. Default is 16.
- *Frg* scalar: if set, also estimates the contribution coming from extra foreground residuals. Default is no.
- *RMSmoothing* scalar: if set, applies a smoothing of the estimated noise variance maps. Default is no.
- *tol* : scalar, convergence precision. Default is $1e^{-6}$.

8.3 Wavelets

8.3.1 Mexican Hat Wavelet Transform : mrs_wtmexhat

Convolves an input spherical map with the mexican hat wavelet function at a given scale.

USAGE: Scale = mrs_wtmexhat(Image, ScaleParameter)

where

- *Image* : IDL array of Healpix map. Input image to be transformed
- *ScaleParameter* : float = Scale parameter

Examples:

- coef = mrs_wtmexhat(Image, $\frac{\sqrt{3}}{3}$)
  Convolve the data with a mexican hat wavelet function, with a scale parameter equal to $\frac{\sqrt{3}}{3}$ which corresponds to an angular spread in θ of about $\frac{\pi}{6}$.

8.3.2 bi-orthogonal wavelet transform : mrs_owttrans

Computes the bi-orthogonal wavelet transform on the sphere with the filter bank 7/9 (L_2 normalization), using the HEALPix pixel representation (nested data representation). The wavelet transform is applied successively on the 12 faces of the Healpix image. The output is an IDL structure.

USAGE: mrs_owttrans, Imag, Trans, NbrScale=NbrScale, Opt=Opt

where

- *Image* : Input IDL array of a Healpix map to be transformed.
- *Trans* : Output IDL structures with the following fields:
  - *NbrScale* : Number of scales of the wavelet transform.
  - *Coef*: 3D IDL array $[*, *, 12]$ which contains the wavelet coefficients. $COEF[*, *, f]$ is the wavelet transform of the face f (f=0..11).

  - *Nx* : int. number of pixels on the side of the Healpix patch, nside
  - *Ny* : int, same as Nx.

- *NbrScale* : int, input optional parameter specifying the number of scales of the wavelet transform (default is 4)
- *Opt* : string: if package MR1 is also installed, extra keyword used by mr_transform.pro for the computation of the wavelet transform

Examples:

- mrs_owttrans, Imag, WT, NbrScale=5
  Compute the bi-orthogonal wavelet transform with five scales.
- tvscl, WT.coef[*,*,f]
  plot the wavelet transform of the f$^{\text{th}}$ face, for $f \in \{0 \dots 11\}$.

8.3.3 bi-orthogonal wavelet reconstruction : mrs_owtrec

Reconstructs an image on the Sphere from its bi-orthogonal wavelet transform.

USAGE: mrs_owtrec, WT_Struct, result

where

- *WT_Struct* : Input IDL structure Wavelet transform structure.
- *Result* : Output 1D array of an Healpix image (nested format).

Examples:

- mrs_owttrans, Imag, WT, NbrScale=5
  Compute the bi-orthogonal wavelet transform with five scales.
- mrs_owtrec, WT, RecIma
  Wavelet reconstruction.

8.3.4 Undecimated Isotropic Wavelet Transform : mrs_wttrans

Computes the undecimated isotropic wavelet transform on the sphere, using the Healpix pixel representation (nested data representation) or Glesp Data representation. The wavelet function is zonal and its spherical harmonics coefficients $a_{l,0}$ follow a cubic box-spline profile. If DifInSH is set, wavelet coefficients are derived in the Spherical Harmonic Space, otherwise (default) they are derived in the direct space.

USAGE: mrs_wttrans, Imag, Trans, NbrScale=NbrScale,
Healpix_with_Glesp=Healpix_with_Glesp, lmax=lmax,
MeyerWave=MeyerWave, NeedletWave=NeedletWave,
B_NeedletParam=B_NeedletParam, DifInSH=DifInSH

where

- *Imag* : Input IDL array of Healpix map or Glesp image IDL structure. Input image be transformed.
- *Trans* : Output IDL structures with the following fields:
  - *NbrScale* : int, number of scales
  - *nside* : int, Healpix nside parameter (0 for a Glesp image)
  - *lmax* : int, maximum l value in the Spherical Harmonic Space
  - *npix* : long, Number of pixels of the input image
  - *Healpix_with_Glesp* : int, 1 if the keyword Healpix_with_Glesp used, otherwise 0
  - *UseGLESP* : int, 1 if the input image was in Glesp format, otherwise 0
  - *MeyerWave* : int, 1 if the keyword MeyerWave used, otherwise 0
  - *NeedletWave*: int , 1 if the keyword NeedletWave is used, otherwise 0
  - *B_NeedletParam*: int, B_NeedletParam (default is 2).
  - *DifInSH* : int, 1 if the keyword DifInSH used, otherwise 0
  - *nx* : int, number of rings (Glesp parameter), otherwise, 0
  - *np* : int, max number of pixel on a ring (Glesp parameter), otherwise, 0
  - *x_sky* : float array with COS(THETA) for each ring (Glesp parameter), otherwise, 0
  - *y_sky* : long array number of pixels/ring (Glesp parameter), otherwise, 0
  - *Coef* : fltarr[npix, NbrScale] wavelet transform of the data

    Coef[*, 0] = wavelet coefficients of the finest scale (highest frequencies).
    Coef[*, NbrScale-1] = coarsest scale (lowest frequencies).
- *NbrScale* : int, optional input parameter specifying the number of scales (default is 4).
- *Lmax* : int, optional input parameter specifying the maximum multipole number l in the spherical harmonics decomposition (default is $3 \times$ nside, should be between $2 \times$ nside and $4 \times$ nside).
- *DifInSH* : Input keyword parameter. If set, the wavelet coefficients are computed as the difference between two resolutions in the spherical harmonics representation. Otherwise, the wavelet coefficients are computed as the difference between two resolutions in direct space.
- *MeyerWave* : If set, use Meyer wavelets and set the keyword DifInSH.
- *Healpix_with_Glesp* : If set, a copy of Imag is done in Glesp format in order to compute the wavelet transform.

Examples:

- mrs_wttrans, Imag, Trans, NbrScale=5
  Undecimated Wavelet transform with five scales.
- tvs, Trans.coef[*,0]
  Visualization of the first scale.

8.3.5 Undecimated Isotropic Wavelet Reconstruction : mrs_wtrec

Reconstructs an image on the sphere from its wavelet coefficients obtained with the undecimated isotropic wavelet transform on the sphere, described right above.

USAGE: mrs_wtrec, Trans, Rec, filter=filter

where

- *Trans*: input IDL structures (see mrs_wttrans).
- *Rec* : Output IDL array of healpix map: reconstructed image from the wavelet coefficients or Glesp image IDL structure if UseGlesp=1.
- *filter* : Input keyword parameter. Use filters for the reconstructions. If this keyword is not set, the reconstructed image is obtained by a simple addition of all wavelet scales. Automatically applied if keyword MeyerWave, NeedletWave, or DifInSH were set at the wavelet decomposition.

Examples:

- mrs_wttrans, Imag, Trans, NbrScale=5
  Undecimated Wavelet transform with five scales.
- mrs_wtrec, Trans, RecIma
  Reconstruction of the image from its wavelet coefficients.

8.3.6 Undecimated Isotropic Wavelet Packet Transform : mrs_wptrans

Computes the undecimated isotropic wavelet packet transform on the sphere with Meyer wavelets, using the Healpix representation (nested data representation) or the Glesp representation.

USAGE: mrs_wptrans, Imag, Trans, NbrScale=NbrScale

where

- *Imag* : Input IDL array of Healpix map or Glesp image IDL structure. Input image be transformed.
- *Trans* : Output IDL structures with the following fields:

  - *NbrScale* : int, number of scales
  - *nside* : int, Healpix nside parameter (0 for a Glesp image)
  - *lmax* : int, maximum l value in the Spherical Harmonic Space
  - *npix* : long, Number of pixels of the input image
  - *UseGLESP* : int, 1 if the input image was in Glesp format, otherwise 0
  - *nx* : int, number of rings (Glesp parameter), otherwise, 0
  - *np* : int, max number of pixel on a ring (Glesp parameter), otherwise, 0
  - *x_sky* : float array with COS(THETA) for each ring (Glesp parameter), otherwise, 0
  - *y_sky* : long array number of pixels/ring (Glesp parameter), otherwise, 0
  - *Coef* : fltarr[npix, NbrScale] wavelet transform of the data
- *NbrScale* : int, optional output parameter specifying the number of scales.
- **The image reconstruction is done by using the procedure mrs_wtrec**

Examples:

- mrs_wptrans, Imag, Trans
  Undecimated Wavelet Packet transform.
- mrs_wtrec, Trans, RecIma
  Reconstruction of the image from its wavelet coefficients.

8.3.7 Undecimated Wavelet Transform with "A Trou" Algorithm : mrs_attrans

Compute the isotropic wavelet transform on the sphere, using the Healpix pixel representation (nested data representation) and using the "à trous" algorithm. The wavelet transform is applied successively on the 12 faces of the Healpix image. The output is a IDL structure.

USAGE: mrs_attrans, Imag, Trans, NbrScale=NbrScale, Opt=Opt, modif=modif, healpix=healpix

where

- *Imag* : Input IDL array of Healpix map or Glesp image IDL structure. Input image be transformed.
- *Trans* : Output IDL structures with the following fields:
  - *NbrScale* : int, number of scales of the wavelet transform.
  - *Coef* : 4D IDL float array [*,*,*,12] : Wavelet coefficients cube containing all wavelet coefficients.

    Coef[x, y, j, f] = wavelet coefficients at face f (f=0..11), position x, y and scale j

  - *Nx* : int, number of pixels on the side of the Healpix patch, nside
  - *Ny* : int, same as Nx

- *NbrScale* : int, number of scales of the wavelet transform, default is 4
- *Opt* : string, if package MR1 is installed, extra keyword used by mr_transform.pro
- *modif* : scalar, if set, add extra smoothing with spline filtering
- *healpix* : scalar, if set, change ATTrans.coef to a 2D array [c, j] by reordering wavelet coefficients at scale j as a Healpix NESTED map.

Examples:

- mrs_attrans, Imag, WT, NbrScale=5
  Undecimated Wavelet transform with five scales and "A TROU" algorithm.
- tvscl, WT.coef[*,*,0,f]
  Plot the fth face of the first scale of the wavelet transform.

8.3.8 Undecimated Wavelet Transform with "A Trou" Algorithm reconstruction : mrs_atrec

Reconstructs an image on the sphere from its wavelet coefficients obtained with the undecimated wavelet transform on the sphere with the "A TROU" algorithm, described right above.

USAGE: mrs_atrec, Trans, Rec

where

- *Trans*: input IDL structures with the following fields:
  - *NbrScale* : int, number of scales of the wavelet transform.
  - *Coef* : 4D IDL float array [*,*,*,12] : Wavelet coefficients cube containing all wavelet coefficients.

    Coef[x, y, j, f] = wavelet coefficients at face f (f=0..11), position x, y and scale j
  - *Nx* : int, number of pixels on the side of the Healpix patch, nside
  - *Ny* : int, same as Nx
- *Rec* : Output IDL array of healpix map in nested format: reconstructed image from the wavelet coefficients.

Examples:

- mrs_attrans, Imag, Trans, NbrScale=5
  Undecimated Wavelet transform with five scales.
- mrs_atrec, Trans, RecIma
  Reconstruction of the image from its wavelet coefficients.

8.3.9 Pyramidal Wavelet Transform : mrs_pwttrans

Computes the pyramidal wavelet transform on the sphere, using the Healpix pixel representation (nested data representation) or Glesp Data representation. The wavelet function is zonal and its spherical harmonics coefficients $a_{l,0}$ follow a cubic box-spline profile.

USAGE: mrs_pwttrans, Imag, Trans, NbrScale=NbrScale, lmax=lmax, DifInSH=DifInSH, MeyerWave=MeyerWave

where

- *Imag* : Input IDL array of Healpix map or IDL structure of a Glesp map. Input image be transformed.
- *Trans* : Output IDL structures with the following fields:
  - *UseGLESP* : int, 1 if the input image was in Glesp format, otherwise 0
  - *NbrScale* : int, number of scales
  - *nside* : int, Healpix nside parameter, only present with healpix input image
  - *npix* : long, Number of pixels of the input image
  - *lmax* : int, maximum l value in the Spherical Harmonic Space at the first scale
  - *Tab_lmax* : int array Tab_lmax[j], lmax at scale j+1, j=0..NbrScale-1
  - *Tab_nside* : int array Tab_nside[j], nside parameter of the scale j+1, j=0..NbrScale-1 (Healpix input map) or nx number of rings, Glesp parameter of the scale j+1, j=0..NbrScale-1 (Glesp input map)
  - *nx* : int, number of rings at the first scale, Glesp parameter, only present with glesp input image
  - *np* : int, max number of pixel on a ring at the first scale, Glesp parameter, only present with glesp input image
  - *Scalej* : j th scale (j=1..NbrScale), j = 1 is the finest scale (highest frequencies), with j = NbrScale, coarsest resolution. A IDL array of healpix map or IDL structure of a Glesp map
  - *MeyerWave* : int = 1 if the keyword MeyerWave used, otherwise 0
  - *DifInSH* : int = 1 if the keyword DifInSH used, otherwise 0
- *NbrScale* : Optional input parameter specifying the number of scales in the decomposition (default is 4).

- *lmax* : Optional input parameter specifying the maximum multipole number l in the spherical harmonics decomposition (default is $3 \times$ nside, should be between $2 \times$ nside and $4 \times$ nside).
- *DifInSH* : Compute be difference between two resolution in the spherical harmonic space instead of the direct space.
- *MeyerWave* : If set, use Meyer wavelets and set the keyword DifInSH

Examples:

- mrs_pwttrans, Imag, Trans, NbrScale=5
  Pyramidal Wavelet transform with five scales.
- tvs, Trans.Scale1
  Visualization of the first scale.

8.3.10 Pyramidal Wavelet Reconstruction : mrs_pwtrec

Computes the inverse pyramidal wavelet transform on the sphere.

USAGE: mrs_pwtrec, Trans, Rec, filter=filter

where

- *Trans* : Input IDL structures with the following fields:
  - *UseGLESP* : int, 1 if the input image was in Glesp format, otherwise 0
  - *NbrScale* : int, number of scales
  - *nside* : int, Healpix nside parameter, only present with healpix input image
  - *npix* : long, Number of pixels of the input image
  - *lmax* : int, maximum l value in the Spherical Harmonic Space at the first scale
  - *Tab_lmax* : int array Tab_lmax[j], lmax at scale j+1, j=0..NbrScale-1
  - *Tab_nside* : int array Tab_nside[j], nside parameter of the scale j+1, j=0..NbrScale-1 (Healpix input map) or nx number of rings, Glesp parameter of the scale j+1, j=0..NbrScale-1 (Glesp input map)
  - *nx* : int, number of rings at the first scale, Glesp parameter, only present with glesp input image
  - *np* : int, max number of pixel on a ring at the first scale, Glesp parameter, only present with glesp input image
  - *Scalej* : j th scale (j=1..NbrScale), j = 1 is the finest scale (highest frequencies), with j = NbrScale, coarsest resolution. A IDL array of healpix map or IDL structure of a Glesp map
  - *MeyerWave* : int = 1 if the keyword MeyerWave used, otherwise 0
  - *DifInSH* : int = 1 if the keyword DifInSH used, otherwise 0

- *Rec* : IDL array of Healpix map or IDL structure of a Glesp map. Output reconstructed image.

- *filter* : Optional inout keyword. If set, conjugate filters are used in the reconstruction. Otherwise, the reconstructed image is obtained by a simple interpolation/addition of all wavelet scales. When computing direct transform (mrs_pwttrans function), if the keywords DifInSH or MeyerWave were setted, filter is automatically used.

Examples:

- mrs_pwttrans, Imag, Trans, NbrScale=5
  Pyramidal Wavelet transform with five scales.

- mrs_pwtrec, Trans, RecIma
  Reconstruction of the image from its wavelet coefficients.

8.3.11 Extract a Wavelet Scale : mrs_wtget

Returns a scale of the wavelet transform obtained by the command *mrs_wttrans* or by the command *mrs_pwttrans*.

USAGE: Scale = mrs_wtget(Trans, ScaleNumber, Face=Face, NormVal=NormVal)

where

- *Trans* : Input IDL structure containing the wavelet transform.

- *ScaleNumber* : integer scale number. The scale number must be between 0 and Trans.NbrScale-1

- *Face* : optional input keyword parameter. If set, the routine returns a Cube[*,*,0:11] containing the twelve faces of the HEALPix representation.

- *NormVal* : float, normalization coefficient in that band.

Examples:

- mrs_pwttrans, Imag, Trans, NbrScale=5
  Pyramidal Wavelet transform with five scales.

- Band1 = mrs_wtget(Trans,0)
  Extract the first wavelet scale.

8.3.12 Insert a band into Wavelet Transform : mrs_wtput

Replaces a map of coefficients at a given scale in the wavelet transform obtained by the command *mrs_wttrans* or by the command *mrs_pwttrans*.

USAGE: mrs_wtput, Trans, Scale, ScaleNumber, Face=Face

where

- *Trans* : Input IDL structure containing the wavelet transform.
- *Scale* : IDL array, the wavelet scale we want to insert in the specified decomposition.
- *ScaleNumber* : integer. Specifies the scale number to be replaced by the given Scale map. The scale number must be between 0 and Trans.NbrScale − 1.
- *Face* : If set, the routine put into Trans a Cube[*,*,0:11] containing the twelve faces of the HEALPix representation
- *NormVal* : float: Normalization value of the band.

Examples:

- mrs_pwttrans, Imag, Trans, NbrScale=5
  Pyramidal Wavelet transform with five scales.
- Band1 = mrs_wtget(Trans, 0)
  Extract the first wavelet scale.
- Band1_thres = mrs_absthreshold(Band1, mrs_sigma(Band1))
  Hard thresholding of the scale.
- mrs_wtput, Trans, Band1_thres, 0
  Insert the new wavelet scale.

8.3.13 Visualization of the wavelet scales : mrs_wttv

Visualization of the wavelet transform obtained by the command *mrs_wttrans* or by the command *mrs_pwttrans*. If the keyword WRITE is set to a string, then all the scales are written on the disk as PNG files, and the string is used as a prefix for the file name of the different scales.

USAGE: mrs_wttv, Trans, Tit=Tit, write=write, graticule=graticule, min=min, max=max, big=big

where

- *Trans* : Input IDL structure containing the wavelet transform.
- *Tit* : string: Title of the plot.

- *write* : string: Prefix filename. If set, write to disk each scale of the wavelet transform in PNG format.
- *graticule* : this is the GRATICULE keyword in the HEALPix command MOLLVIEW.
- *min* : float Mollview Healpix command min, new min value to be used for the display.
- *max* : float Mollview Healpix command max, new max value to be used for the display.
- *big* : bool if set, Mollview Healpix keyword pxsize is set to 1500

Examples:

- mrs_pwttrans, Imag, Trans, NbrScale=5
  Pyramidal Wavelet transform with five scales.
- mrs_wttv, Trans
  Visualization of all scales.

8.4 Ridgelet

8.4.1 Ridgelet transform : mrs_ridtrans

Compute the ridgelet transform on the sphere using the Healpix pixel representation (nested data representation). The standard ridgelet transform is applied on the 12 faces of the Healpix image. The output is an IDL structure. A band at scale j ($j = 0 \ldots$NBRSCALE-1) can be extracted using the function *mrs_ridget(Rid, j)* (e.g. Scale2 = mrs_ridget(RidTrans, 2)) and a band can be inserted in the transformation using the routine *mrs_ridput* (e.g. mrs_ridput, RidTrans, Scale2, 2).

USAGE: mrs_ridtrans, Imag, RidTrans, NbrScale=NbrScale, overlap=overlap, blocksize=blocksize, Opt=Opt

where

- *Image* : Input IDL HEALPix array containing the input map.
- *RidTrans* : Output IDL structure with the following fields:
  - *NbrScale* : long, Number of scales of the ridgelet transform.
  - *Coef* : Ridgelet coefficients. If MR1 package is installed, it is a 3D IDL array [*,*,12] containing all ridgelet coefficients. If MR1 package is not installed, it is a IDL array of 12 structures (one for each Healpix face).
  - *Bsize* : long, Block size used in the ridgelet transform.
  - *nxb* : long, Number of blocks in the x-axis direction.

  - *nyb* : long, Number of blocks in the y-axis direction.
  - *Overlap* : long, is equal to 1 if blocks are overlapping.
  - *TabNorm* : float Array[0 : NBRSCALE-1]: Normalization value for each scale.

- *NbrScale* : int, number of scales in the ridgelet decomposition. By default it is automatically estimated.

- *Overlap* : if this keyword is set, the blocks in the local ridgelet transform overlap by half their size, otherelse the blocks do not overlap.

- *Blocksize* : long, this is the size of the square blocks on which the local ridgelet transform is computed. If not set, then the blocksize is taken to be half the size of the individaul faces ie blocksize = nside/2. Blocksize is required to be a power of two smaller than nside which is also a power of two. There is no testing of this requirement.

- *Opt* : string, sets options to used if the mre package is available. If mre is not available, then Opt is useless, and only procedures in mrs are used.

Examples:

- mrs_ridtrans, Imag, Rid
  Compute the ridgelet transform

- mrs_ridtrans, Imag, Rid, /overlap, blocksize=32
  Ditto, but using a overlapping blocks of size 32.

8.4.2 Ridgelet reconstruction : mrs_ridrec

Reconstructs an image on the Sphere from its ridgelet transform (see mrs_ridtrans).

USAGE: mrs_ridrec, Rid_Struct, Result

where

- *Rid_Struct* : Input IDL structure, Ridgelet transform structure (see mrs_ridtrans).

- *Result* : Output 1D array of an Healpix image (nested format).

Examples:

- mrs_ridtrans, Imag, Rid
  Compute the ridgelet transform

- mrs_ridrec, Rid, RecIma
  Ridgelet reconstruction.

8.4.3 Extract a ridgelet band : mrs_ridget

Extracts a ridgelet band from the ridgelet transform (see mrs_ridtrans). A specific normalization can be applied to the local ridgelet coefficients. Indeed, after applying the ridgelet transform to all blocks, we obtain a set of n_b blocks $R_i(a, b, \theta)$ $(i = 1 \dots n_b)$, and for each scale, orientation and position (a, b, θ), we extract the vector $V_{a,b,\theta}(i)$. Then the normalization consists in dividing the ridgelet coefficients $R_i(a, b, \theta)$ $(i = 1..n_b)$ by their MAD value (Median Absolution Deviation) defined by MAD = median($\mid x \mid$)/0.6745 (Rousseeuw and Croux 1993). Hence, we normalize the ridgelet coefficients by the following expression:

$$\bar{R}_i(a, b, \theta) = \frac{R_i(a, b, \theta)}{\mathrm{MAD}(V_{a,b,\theta})} \tag{8.2}$$

If the keyword NormMad is set, the normalization is applied.

USAGE: Result = mrs_ridget(Rid_Struct, ScaleRid, NormMad=NormMad, ImaMean=ImaMean, ImaMad=ImaMad)

where

- *Rid_Struct* : Input IDL structure. Ridgelet transform structure (see mrs_ridtrans).
- *ScaleRid* : int, input Ridgelet band number.
- *NormMad* : scalar, if set, normalize the coefficients by the Median Absolution Deviation of all coefficients at a give position in the block.
- *ImaMean* : 2D IDL float array: Image containing the mean value for all coefficients at a given position in the block.
- *ImaMad* : 2D IDL float array: Image containing the normalization parameters.
- *Result* : 4D IDL float array[*,*,*,12], output extracted band.

Examples:

- mrs_ridtrans, Imag, Rid
  Compute the ridgelet transform
- Band = mrs_ridrec(Rid,0)
  Extract the first scale.

8.4.4 Insert a band into Ridgelet Transform : mrs_ridput

Insert a band in the ridgelet transform (see mrs_ridtrans).

USAGE: mrs_ridput, Rid_Struct, Band, ScaleRid

where

- *Rid_Struct* : Input/Output IDL structure. Ridgelet transform structure (see mrs_ridtrans).
- *Band* : IDL 4D array: input band to insert in the ridgelet transform.
- *ScaleRid* : int, input Ridgelet band number.

Examples:

- mrs_ridtrans, Imag, Rid
  Compute the ridgelet transform
- Band = mrs_ridget(Rid,0)
  Extract the first scale.
- Band[*] = 0 Set the band to zero.
- mrs_ridput, Rid_Struct, Band, 0
  Reinsert the modified band.

8.5 Curvelet

8.5.1 Curvelet transform : mrs_curtrans

Computes the curvelet transform on the sphere, using the Healpix pixel representation (nested data representation). A band of the curvelet transform is defined by two number, the 2D WT scale number and the ridgelet scale number. The output is an IDL structure. A band at wavelet scale j (j=0...NBRSCALE-1) and ridgelet scale j_1 can be extracted using the function mrs_curget(Curtrans, j, j_1) (ex: Scale2_1 = mrs_curget(CurTrans, 2, 1)) and a band can be inserted in the transformation using the routine mrs_curput (ex: mrs_curput, CurTrans, Scale2_1, 2, 1). By default, the pyramidal curvelet is applied. If the keyword UNDEC is set, then the standard undecimated curvelet transform is applied.

USAGE: mrs_curtrans, Imag, CurTrans, Opt=Opt, lmax=lmax, NbrScale=NbrScale, Overlap=Overlap, Undec=Undec, FirstBlockSize=FirstBlockSize, Silent=Silent

where

- *Image* : Input IDL Healpix array of the map to be transformed.
- *RidTrans* : Output IDL structure with the following fields:
  - *NbrScale* : int, Number of scales of the ridgelet transform.
  - *TabBlockSize* : int array TABBLOCKSIZE[j], Block size in the ridgelet transform at scale j, j = 0...NBRSCALE-1.
  - *TabNbrScaleRID* : int array TABNBRSCALERID[j], number of ridgelet band at scale j.
  - *TabNorm* : 2D IDL array : Normalization array.
  - *RidScale1* : IDL structure : ridgelet transform of the first wavelet scale (see mrs_ridtrans for details).
  - *RidScalej* : IDL structure : ridgelet transform of the j^{th} wavelet scale, $j = 1 \ldots$NBRSCALE-1.
  - *LastScale* : 1D IDL float array: Healpix image of the coarsest scale.

  - *WT* : IDL structure: Wavelet structure (for internal use only).
  - *PyrTrans* : int, equal to 1 for a pyramidal curvelet transform and 0 otherwise .

- *NbrScale* : int, Number of scales in the 2D wavelet transform (default 4).
- *Undec* : int, if set, an undecimated curvelet transform is used instead of the pyramidal curvelet transform.
- *FirstBlockSize* : int Block size in the ridgelet transform at the finest scale (default value is 16).
- *Lmax* : int, Number of used spherical harmoniques used in the wavelet transform (default is $3 \times$ nside, should be between $2 \times$ nside and $4 \times$ nside).
- *Overlap* : int, if set blocks in the internal ridgelet transform are overlapping.
- *Opt* : string, optionnal parameters used by mrs_ridtrans (see mrs_ridtrans).
- *Silent* : scalar, if set, verbose mode disabled

Example:

- mrs_curtrans, Imag, Cur
  Compute the curvelet transform

8.5.2 Curvelet reconstruction : mrs_currec

Reconstructs an image on the Sphere from its curvelet transform (see mrs_curtrans).

USAGE: mrs_currec, Cur_Struct, result

where

- *Cur_Struct* : Input IDL structure, Curvelet transform structure (see mrs_curtrans).
- *Result* : Output 1D IDL array of the reconstructed Healpix image (nested format).

Examples:

- mrs_curtrans, Imag, Cur
  Compute the curvelet transform
- mrs_currec, Cur, RecIma
  Curvelet reconstruction.

8.5.3 Extract a curvelet band : mrs_curget

Extracts a curvelet band from the curvelet transform (see mrs_curtrans). If the keyword NormMad is set, a normalization is applied (see mrs_ridget).

USAGE: Result = mrs_curget(Cur_Struct, ScaleWT2D, ScaleRid, NormMad=NormMad, ImaMean=ImaMean, ImaMad=ImaMad)

where

- *Cur_Struct* : Input IDL structure, Curvelet transform structure (see mrs_curtrans).
- *ScaleWT2D* : int, specifies in which 2D WT scale to get the curvelet band.
- *ScaleRid* : int, specifies which ridgelet band of the specified wavelet scale corresponds to the requested curvelet band.
- *NormMad* : scalar, if set, normalize the coefficients by the Median Absolution Deviation of all coefficients at a given position in the block.
- *ImaMean* : 2D IDL array: Image containing the mean value for all coefficients at a given position in the block.
- *ImaMad* : 2D IDL array: Image containing the normalization parameters.
- *Result* : 4D IDL float array[*,*,*,12], extracted band

Example:

- mrs_curtrans, Imag, Cur
  Compute the ridgelet transform
- Band = mrs_ridrec(Rid,0,0)
  Extract the first scale.

8.5.4 Insert a band into the Curvelet Transform : mrs_curput

Inserts a band back in the curvelet transform (see mrs_curtrans).

USAGE: mrs_curput, Cur_Struct, Band, ScaleWT2D, ScaleRid

where

- *Cur_Struct* : Input/Output IDL structure, Curvelet transform structure (see mrs_curtrans).
- *Band* : 4D IDL float array[*,*,*,12], input band to be inserted in the curvelet transform.
- *ScaleWT2D* : int, specifies in which 2D WT scale to put the given curvelet band.
- *ScaleRid* : int, specifies which ridgelet band of the specified wavelet scale is to be replaced by the given curvelet band.

Examples:

- mrs_curtrans, Imag, Cur
  Compute the curvelet transform
- Band = mrs_curget(Rid, 0, 0)
  Extract the first scale.
- Band[*] = 0 Set the band to zero.
- mrs_curput, Cur_Struct, Band, 0, 0
  Reinsert the modified band.

8.6 Denoising

8.6.1 Wavelet filtering : mrs_wtfilter

Wavelet denoising of an image on the sphere (Healpix pixel nested representation) or Glesp Data representation. By default, the noise is assumed to follow a Gaussian distribution. If the keyword SigmaNoise is not set, then the noise standard deviation is automatically estimated. If the keyword MAD is set, then a correlated Gaussian noise is considered, and the noise level at each scale is derived from the Median Absolution Deviation (MAD) method. If the keyword KillLastScale is set, the coarsest resolution is set to zero. If the "Pyr" keyword is used, then the pyramidal WT is used instead of the undecimated WT. If the "atrou" keyword is used, then the "a trou" WT is used instead of the undecimated WT. If the keyword CYCLE is set, the denoising is performed three times, by shifting the data by $\pi/4$ and $-\pi/4$, denoising the shifted version, and averaging the unshifted denoising maps. This procedure allows us to remove the block effect which may appear on the border of the Healpix faces. The thresholded wavelet coefficients can be obtained using the keyword Trans. If the input keyword NITER is set, then an iterative algorithm is applied and if the POS keyword is also set, then a positivity constraint is added.

USAGE: mrs_wtfilter, Imag, Filter, NbrScale=NbrScale, NSigma=NSigma, SigmaNoise=SigmaNoise, lmax=lmax, TabNSigma=TabNSigma, mad=mad, localmad=localmad, WinMinSize=WinMinSize, Soft=Soft, niter=niter, pos=pos, Pyr=Pyr, cycle=cycle, Trans=Trans, KillLastScale=KillLastScale, FirstScale=FirstScale, fdr=fdr, Use_FdrAll=Use_FdrAll, mask=mask, FilterLast=FilterLast, atrou=atrou, OutMask=OutMask

where

- *Image* : Input IDL Healpix array or Glesp IDL structure containing the input map.
- *Filter* : Output IDL Healpix array or Glesp IDL structure containing the output filtered map.
- *NbrScale* : int = Number of scales (default is 4).
- *NSigma* : float = Level of thresholding (default is 3).

- *TabNSigma* : float array = Level of thresholding at each scale
- *SigmaNoise* : float = Noise standard deviation. Default is automatically estimated.
- *mad* : if set, then the noise level is derive at each scale using the MAD of the wavelet coefficient. MAD = median (ABS(WaveletScale)) / 0.6745.
- *localmad* : int if set, similar to keyword MAD but with one value for each patch of the image and each instead of one value for the full image at each scale.
- *WinMinSize* : int minimal size of patches (default is 8).
- *KillLastScale* : if set, the last scale is set to zero.
- *niter* : number of iterations used in the reconstruction.
- *pos* : if set, the solution is assumed to be positive.
- *Pyr* : if set, a pyramidal WT is used instead of the the undecimated WT.
- *cycle* : int: if set, then a cycle spanning is applied.
- *FirstScale* : int: Consider only scales larger than FirstScale. Default is 1 (i.e. all scales are used).
- *Soft* : if set, use soft thresholding instead of hard thresholding.
- *fdr* : float between 0 (default) and 1 (max, if greater or equal to 1, set to 0.05), used to estimate a threshold level instead of a NSigma threshold, threshold is applied from scale j=FirstScale to the last.
- *Use_FdrAll* : same as fdr but applied to all scales.
- *FilterLast* : if set, the last scale is filtered.
- *mask* : IDL array of healpix map, input mask applied.
- *lmax* : int = maximum l value in the Spherical Harmonic Space.
- *atrou* : if set, a "a trou" WT is used instead of the the undecimated WT.
- *Trans* : IDL structure: Thresholded wavelet decomposition of the input image.
- *OutMask* : IDL array of healpix map, part of Imag that were set to 0 via the filtering, including keyword mask if used.
- **localmad, cycle and atrou keywords don't work with Glesp images, an error will be generated**

Examples:

- mrs_wtfilter, Imag, Filter, NbrScale=5
  Wavelet filtering with five scales.
- mrs_wtfilter, Imag, Filter, NbrScale=5, Nsigma=5
  Ditto, but using a 5 sigma threshold.

8.6.2 Curvelet filtering : mrs_curfilter

Curvelet denoising of an image on the sphere (Healpix pixel representation). By default Gaussian noise is considered. If the keyword SigmaNoise is not set, then the noise standard deviation is automatically estimated. If the keyword MAD is set, then a correlated Gaussian noise is considered, and the noise level at each scale is derived from the Median Absolution Deviation (MAD) method. If the keyword KillLastScale is set, the coarsest resolution is set to zero. If the UNDEC keyword is used, then a undecimated decomposition is used instead of the pyramidal WT. The threshold curvelet coefficients can be obtained using the keyword Trans. If the input keyword NITER is set, then an iterative algorithm is applied and if the POS keyword is also set, then a positivity constraint is added. If the keyword CYCLE is set, the denoising is performed three times, by shifting the data by $\pi/4$ and $-\pi/4$, denoising the shifted version, and averaging the unshifted denoising maps. This procedure allows us to remove the block effect which may appear on the border of the Healpix faces.

USAGE: mrs_curfilter, Image, Filter, NbrScale=NbrScale, NSigma=NSigma, SigmaNoise=SigmaNoise, mad=mad, Trans=Trans, niter=niter, pos=pos, Undec=Undec, KillLastScale=KillLastScale, FirstBlockSize=FirstBlockSize, cycle=cycle, FirstScale=FirstScale

where

- *Image* : Input IDL Healpix array containing the input map.
- *Filter* : Output IDL Healpix array containing the output filtered map.
- *NbrScale* : int = Number of scales (default is 4).
- *NSigma* : float = Level of thresholding (default is 3).
- *SigmaNoise* : float = Noise standard deviation. Default is automatically estimated.
- *MAD* : if set, then the noise level is derive at each scale using the MAD of the wavelet coefficient. MAD = median (ABS(WaveletScale)) / 0.6745.
- *KillLastScale* : if set, the last scale is set to zero.
- *FirstBlockSize* : int Block size in the ridgelet transform at the finest scale (default value is 16).
- *niter* : int, number of iterations used in the reconstruction.
- *pos* : if set, the solution is assumed to be positive.
- *Undec* : if set, an undecimated WT is used instead of the the pyramidal WT.
- *cycle* : int, if set a cycle spanning is applied.
- *FirstScale* : int, consider only scales larger than FirstScale. Default is 1 (i.e. all scales are used).
- *Trans* : Output optional IDL structure for storing the thresholded curvelet decomposition of the input image.

Examples:

- mrs_curfilter, Imag, Filter, NbrScale=5
  Pyramidal Curvelet filtering with five scales.
- mrs_curfilter, Imag, Filter, NbrScale=5, Nsigma=5
  Ditto, but using a 5 sigma threshold.

8.6.3 Combined filtering : mrs_cbfilter

Combined filtering using Wavelet and Curvelet of an image on the sphere (Healpix pixel representation). By default, Gaussian noise is considered and if the keyword SigmaNoise is not set, then the noise standard deviation is automatically estimated. If the keyword MAD is set, then a correlated Gaussian noise is considered, and the noise level at each scale is derived from the Median Absolution Deviation (MAD) method. If the keyword KillLastScale is set, the coarsest resolution is set to zero. If the "undec" keyword is used, then a undecimated decomposition is used instead of the pyramidal WT. An iterative algorithm is applied and the keyword NITER gives the number of iterations (10 iterations by default).

USAGE: mrs_cbfilter, Image, Filter, NbrScale=NbrScale, NSigma=NSigma, SigmaNoise=SigmaNoise, mad=mad, niter=niter, pos=pos, Undec=Undec, KillLastScale=KillLastScale, FirstBlockSize=FirstBlockSize, FirstScale=FirstScale

where

- *Image* : Input IDL Healpix array containing the input map.
- *Filter* : Output IDL Healpix array containing the output filtered map.
- *NbrScale* : int = Number of scales (default is 4).
- *NSigma* : float = Level of thresholding (default is 3).
- *SigmaNoise* : float = Noise standard deviation. Default is automatically estimated.
- *MAD* : if set, then the noise level is derive at each scale using the MAD of the wavelet coefficient. MAD = median (ABS(WaveletScale)) / 0.6745.
- *KillLastScale* : if set, the last scale is set to zero.
- *niter* : int, number of iterations used in the reconstruction (default value is 10).
- *pos* : if set, the solution is assumed to be positive.
- *Undec* : if set, an undecimated WT is used instead of the the pyramidal WT.
- *FirstBlockSize* : int Block size in the ridgelet transform at the finest scale (default value is 16).
- *FirstScale* : int, consider only scales larger than FirstScale. Default is 1 (i.e. all scales are used).

Examples:

- mrs_cbfilter, Imag, Filter, NbrScale=5
  Pyramidal Curvelet and pyramidal wavelet filtering with five scales.
- mrs_cbfilter, Imag, Filter, NbrScale=5, Nsigma=5
  Ditto, but using a 5 sigma threshold.

8.7 Blind Source Separation

8.7.1 Blind source separation using JADE : mrs_jade

Apply the ICA method JADE (Cardoso 1999) on data in different settings : the mixed multichannel data gathered from m sensors, may consist of either 1D time series, 2D flat images or spherical maps. A mask can be specified to indicate missing or invalid pixels. The components to be separated are all assumed to be independently and identically distributed random fields in the specified representation. The possible representations offered here are 'initial' or 'wavelet'. The chosen wavelet transform is an orthogonal wavelet transform or an extension of it to the sphere.

USAGE: mrs_jade, data, topology, nb_sources, sources, demixingmat, domain = domain, mask = mask, nb_scales=nb_scales

where

- *data* : either an IDL 2D array of size m*T in the **'1D'** case, or an IDL 3D array of size tx*ty*m in the flat **'2D'** case, or an array of strings giving the filenames of m spherical data maps in the Healpix nested format in the **'Sphere'** case.
- *topology* : string = either **'1D'** or **'2D'** or **'Sphere'**. Specifies the topology of the maps in the multichannel data to be processed. This is clearly redundant information but makes things simpler. The specified 'topology' and the structure of the input data should obviously agree.
- *nb_sources* : integer = number of independent sources one wants to recover from the data. The number of sources should be less than or equal to the number of channels m.
- *sources* : either an IDL 2D array of size nb_sources*T in the '1D' case, or an IDL 3D array of size tx*ty*nb_sources in the flat '2D' case, or an array of strings giving the predefined filenames of nb_sources spherical data maps in the Healpix nested format in the 'Sphere' case.
- *demixingmat* : IDL array of size nb_sources * m. Inverse or pseudo inverse of the mixing matrix, used to estimate the source processes from the data according to 'sources = demixingmat * data'.
- *domain* : string = either 'initial' or 'wavelet'. Specifies the representation in which the source separation algorithm JADE should be run i.e. the representation in which the cumulant statistics should be computed (default is 'initial').

- *mask* : either a length T IDL array in the '1D' case, or an IDL array of size tx*ty in the flat '2D' case, or a string giving the filename of a spherical map in the Healpix nested format in the 'Sphere' case. The specified mask should be the same size as one of the data maps. A mask is an array of 0s and 1s where 0 indicates an invalid data sample, and 1 indicates a valid data sample.

  IF A MASK IS SPECIFIED, THE DATA HAVE TO BE MULTIPLIED BY THE MASK PRIOR TO CALLING THE MRS_JADE ROUTINE.

- *nb_scales* : int = number of scales in the wavelet transform including the smooth array (default is nb_scales = 4). There is no verification that it is a valid number of scales for the given data.

Examples:

- mrs_jade, data, 'Sphere', 4, sources, demixingmat

  Recovering four independent sources from a set of spherical mixture maps using Jade.

- mrs_jade, data, '1D', 3, sources, demixingmat, domain = 'wavelet', mask = 'the-mask.fits', nb_scales=5

  Recovering three independent sources from a set of 1D mixtures using Jade in a wavelet representation on five scales, with missing samples specified by a mask.

8.7.2 Blind source separation using fastICA : mrs_fastica

Apply the ICA method fastICA (Hyvärinen et al. 2001) on data in different settings: the mixed multichannel data gathered from m sensors may consist of either 1D time series, 2D flat images or spherical maps. A mask can be specified to indicate missing or invalid pixels. The components to be separated are all assumed to be independently and identically distributed random fields in the specified representation. The possible representations offered here are 'initial' or 'wavelet'. The chosen wavelet transform is an orthogonal wavelet transform or an extension of it to the sphere.

USAGE: mrs_fastica, data, topology, nb_sources, sources, demixingmat, domain = domain, mask = mask, nb_scales=nb_scales

where

- *data* : either an IDL 2D array of size m*T in the **'1D'** case, or an IDL 3D array of size tx*ty*m in the flat **'2D'** case, or an array of strings giving the filenames of m spherical data maps in the Healpix nested format in the **'Sphere'** case.

- *topology* : string = either **'1D'** or **'2D'** or **'Sphere'**. Specifies the topology of the maps in the multichannel data to be processed. This is clearly redundant information but makes things simpler. The specified 'topology' and the structure of the input data should obviously agree.

- *nb_sources* : integer = number of independent sources one wants to recover from the data. The number of sources should be less than or equal to the number of channels m.
- *sources* : either an IDL 2D array of size nb_sources*T in the '1D' case, or an IDL 3D array of size tx*ty*nb_sources in the flat '2D' case, or an array of strings giving the predefined filenames of nb_sources spherical data maps in the Healpix nested format in the 'Sphere' case.
- *demixingmat* : IDL array of size nb_sources * m. Inverse or pseudo inverse of the mixing matrix, used to estimate the source processes from the data according to 'sources = demixingmat * data'.
- *domain* : string = either 'initial' or 'wavelet'. Specifies the representation in which the source separation algorithm fastICA should be run (default is 'initial').
- *mask* : either a length T IDL array in the '1D' case, or an IDL array of size tx*ty in the flat '2D' case, or a string giving the filename of a spherical map in the HEALPix nested format in the 'Sphere' case. The specified mask should be the same size as one of the data maps. A mask is an array of 0s and 1s where 0 indicates an invalid data sample, and 1 indicates a valid data sample.

  IF A MASK IS SPECIFIED, THE DATA HAS TO BE MULTIPLIED BY THE MASK PRIOR TO CALLING THE MRS_FASTICA ROUTINE.
- *nb_scales* : int = number of scales in the wavelet transform including the smooth array (default is nb_scales = 4). There is no verification that it is a valid number of scales for the given data.

Examples:

- mrs_fastica, data, 'Sphere', 4, sources, demixingmat

  Recovering four independent sources from a set of spherical mixture maps using fastICA.
- mrs_fastica, data, '1D', 3, sources, demixingmat, domain = 'wavelet', mask = 'the-mask.fits', nb_scales=5

  Recovering three independent sources from a set of 1D mixtures using fastICA in a wavelet representation on five scales, with missing samples specified by a mask.

8.7.3 Handling missing/masked data through wavelet scales : mrs_mask

When gaps exist in a signal or a map, some wavelet coefficients located outside the initial mask are affected. The extent of the influence of the mask depends on scale. The purpose of this function is to apply the specified wavelet transform to the specified mask and to return a mask on each scale where 1s correspond to valid coefficients (i.e. coefficient which are contaminated by the mask but below some threshold) and 0s correspond to

contaminated coefficients. Implemented for three different topologies and "two" different transforms (ie undecimated a trous algorithm or orthogonal transform): the undecimated transform is the one used in mrs_smica whereas the orthogonal transform is used in mrs_jade.

USAGE: mrs_mask, mask, topology, wt_type, nb_scales, mask_out, nlmax=nlmax

where

- *mask* : either a length T IDL array in the '1D' case, or a tx*ty IDL array in the flat '2D' case, or an IDL array in Healpix nested format in the 'Sphere' case.
- *topology* : string = either **'1D'** or **'2D'** or **'Sphere'**. Specifies the topology of the maps in the multichannel data to be processed. This is clearly redundant information but makes things simpler. The specified 'topology' and the structure of the input data should obviously agree.
- *wt_type* : string = either **'atrous'** or **'ortho'**. Specifies the wavelet transform type to be used, either respectively the undecimated a trous wavelet transform (and extensions in different topologies) or the orthogonal wavelet transform (and extensions in the different topologies).
- *nb_scales* : int = number of scales in the wavelet transform including the smooth array (there is no verification that it is a valid number of scales for the given data).
- *mask_out*:
  - if wt_type = 'atrous', this is either an nb_scales*T array in the 1D case, or a tx*ty*nb_scales array in the flat 2D case, or an npix*nb_scales array of nb_scales spherical masks in Healpix nested format where npix is the size of the initial mask in Healpix nested format.
  - if wt_type = 'ortho', this is either a length T array in the 1D case, or a tx*ty array in the flat 2D case, or an array the same size as the initial mask in Healpix nested format.
- *nlmax* : this is only used in the undecimated spherical wavelet transform so when the specified topology is 'Sphere' and the specified transform is 'atrous'. This is not an optional input in the 'Sphere' AND 'atrous' case. N.B.: The same value of nlmax should be used as in the corresponding spherical wavelet transform of the data maps on which the map is to be applied.

Examples:

- mrs_mask, mask, '1D', 'ortho', 5, mask_out

  Computing the mask to be used on each scale of a 1D orthogonal wavelet transform on five scales of the data.

- mrs_mask, mask, 'Sphere', 'atrous', 5, mask_out, nlmax = 512

  Computing the mask to be used on each scale of an isotropic undecimated spherical wavelet transform on five scales of the data.

8.7.4 Morphological Components Analysis on the sphere : mrs_mca

Apply the sparse component analysis method called Morphological Component Analysis, including a hard thresholding with linear decreasing threhold, on a spherical map in Healpix representation (NESTED format) using several basis decompositions and there transforms on the sphere selected in the following list:

```
1: Isotropic Undecimated Wavelet
2: Pyramidal Wavelet
3: Othogonal Wavelet Transform (on each face)
4: ALM
5: Dirac
6: Curvelet
7: DCT (on each face)
8: A Trou Wavelet (on each face)
9: CMBLET

Default transforms are Pyramidal Wavelet and ALM
```

With the selection of only one transform and the use of a mask, MCA will make an inpainting of the input data.

USAGE: mrs_mca, data_in, data_out, Bounded=Bounded, residual=residual, CstSigma=CstSigma, SelectTrans=SelectTrans, SigmaNoise=SigmaNoise, Positivity=Positivity, LastThreshold=LastThreshold, niter=niter, fit=fit, FirstThreshold=FirstThreshold, tabNameTrans=tabNameTrans, mad=mad, FirstWTDetectScale=FirstWTDetectScale, DCTblocksize=DCTblocksize, NbrScale=NbrScale, mom=mom, expo=expo, soft=soft, lmax=lmax, Mask=Mask, nomean=nomean

where

- *data_in* : Input 1D IDL array of a Healpix map, image to be analysed.
- *data_out* : Output 2D IDL array[*, NbTrans] of Healpix maps, components estimated from data_in. NbTrans is the number of selected transforms (via SelectTrans keyword), by default there are 2 transforms.
- *SelectTrans* : 1D int array with the code number of the selected transforms. SelectTrans[i] value must be between 1 and 9. Default value: SelectTrans = [2,4]
- *niter* : int, iteration number of the MCA algorithm. Default value is 10.

- *Mask* : 1D IDL array of Healpix map, mask applied to data_in. Inpainting on the masked areas.
- *expo* : scalar, if set use an exponential decreasing thresholding instead of linear decreasing thresholding.
- *mom* : scalar, if set use a linear decreasing thresholding with MOM as a first threshold.
- *mad* : scalar, if set use a linear decreasing thresholding with MAD as a first threshold.
- *fit* : scalar, if set fit the threshold levels to ALM decomposition of data_in.
- *soft* : scalar, if set use soft thresholding instead of hard thresholding.
- *SigmaNoise* : float, standard deviation of the noise, assumed gaussian. Default value is 1.
- *NbrScale* : int, number of scale decompositions for wavelet transfroms. Default value is 5.
- *lmax* : int, maximum l number of spherical harmonics. Default value is 3*nside, max value is 3000.
- *Bounded* : scalar, if set constraints the reconstructed components of data_out to be bounded by the min and max of data_in.
- *Positivity* : scalar, if set constraints the reconstructed components of data_out to be positive.
- *CstSigma* : scalar, if set and if a mask is applied, constraints the decompositions coefficients to have the same standard deviation inside and outside the masked area.
- *nomean* : scalar, if set remove the mean of the reconstructed components of data_out. Work only with keywords mask and CstSigma.
- *DCTblocksize* : int, size of the blocks for DCT transform (if selected). Default value is the nside parameter of data_in.
- *FirstWTDetectScale* : int, for isotropic wavelet only, set all wavelet coefficients from Scale ¡ FirstWTDetectScale to 0.
- *LastThreshold* : float, last threshold level. Default value is 0. input / output
- *FirstThreshold* : float, first threshold level. Default is automatically estimated. input / output
- *residual* : Output 1D IDL array of a Healpix map, final residual.
- *tabNameTrans* : Output string array, list of the possible transforms. tabNameTrans = ['Unknown', 'Isotropic Undecimated Wavelet', 'Pyramidal Wavelet', 'Orthogonal Wavelet Transform', 'ALM', 'DIRAC', 'Curvelet', 'DCT', 'a trous WT', 'CMBLET']

Examples:

- mrs_mca, Data, Components, SelectTrans=[2,4,5,6]
  Compute the MCA on a Healpix image data, considering 4 components: Curvelet, ALM, Pyramidal Wavelet and Dirac.

8.7.5 Generalized Morphological Components Analysis on the sphere : mrs_gmca

Apply the blind component separation method called GMCA on spherical maps using Healpix representation (NESTED format) with the orthogonal wavelet transform on the sphere.

USAGE: mrs_gmca, Data, NbrSource, RecSource, MixintMat, NbrIter, NbrScale=NbrScale, VARMEAN=VARMEAN, SpecConst=SpecConst, A0=A0, ForceGalactic=ForceGalactic, DustAlmConst=DustAlmConst, SyncAlmConst=SyncAlmConst, GMCA_mask=GMCA_mask

where

- *Data* : Input 2D IDL float array: Multichannel data on the Sphere (Healpix nested format). Data[*,i] is the ith channel.
- *NbrSource* : int, number of estimated sources. The number of source must be smaller or equal to the number of channels.
- *NbrIter* : int, number of iterations.
- *RecSource* : Output 2D IDL float array, multichannel reconstructed sources. RecSource[*,i] (i=0,NbrSource-1) is the ith source.
- *MixintMat* : Output 2D IDL float array, estimated mixing matrix (Data = MixintMat # RecSource). MixintMat is of size NumberChannels × NbrSource (MixintMat[0:NumberChannels-1, 0:NbrSource-1]).
- *NbrScale* : int, number of scales in the wavelet transform. Default value is 4.
- *VARMEAN* : 2D IDL float array, noise covariance matrix given on input, if not set, it is assumed it is identity matrix.
- *GMCA_mask* : 1D IDL array of Healpix map. optional mask applied to the data.
- *A0* : 1D IDL float array [0..NumberChannels-1]. If set, then the first column of the matrix is considered as known and does not need to be estimated. We have MixintMat[*, 0] = A0
- *SpecConst* : 2D IDL float array, initial mixing matrix with CMB and SZ spectral contraints (the components 0 and 1 of the mixing matrix are fixed).
- *ForceGalactic* : scalar, if SpecConst is also set then the values of the Dust and Synchrotron spectra (the components 2 and 3 of the mixing matrix) are also fixed.

- *DustAlmConst* : scalar, if set then it constraints the first 769 alm coefficients of the 2nd estimated sources (being the estimated dust).
- *SyncAlmConst* : scalar, if set then it constraints the first 769 alm coefficients of the 3rd estimated sources (being the estimated synchrotron).

Examples:

- mrs_gmca, Data, 4, Sources, mat, 100, NbrScale=5
  Compute the GMCA on a 2D data set considering 4 sources, 100 iterations and 5 wavelet scales.

8.7.6 A few more examples

Several scripts are included in the package giving examples of how to run the different source separation codes :

- test_mrs_jade.pro
- test_mrs_fastica.pro
- test_mrs_mask.pro

These scripts use the data files and masks provided with the package in $\$MRS/data$. These scripts use a procedure called test_data_sph.pro to generate synthetic noisy mixtures of the available component maps on the sphere.

8.8 Statistics

8.8.1 Compute several statistics : get_stat

Return statistical information relative to a given data set. The return value is an IDL array of 9 elements.

```
Tab[0] = standard deviation
Tab[1] = skewness
Tab[2] = Kurtosis
Tab[3] = Min
Tab[4] = Max
Tab[5] = HC
Tab[6] = HC^+
Tab[7] = Cumulant of order 5
Tab[8] = Cumulant of order 6

       If the keyword norm is set, the data are first normalized.
```

USAGE: TabStat = get_stat(Data, HCIma=HCIma, norm=norm, TabStatName=TabStatName, qpplot=qpplot, verb=verb, zeromean=zeromean, TabCumulant=TabCumulant)

where

- *Data* : IDL array. Input data to analyze.
- *Norm* : scalar, if set, the input data are normalized and centered (i.e. Data = (Data-Mean)/Sigma).
- *qpplot* : scalar, if set, plot the qpplot of the data.
- *verb* : scalar, if set, the calculated statistics are printed on the screen.
- *zeromean* : scalar, if set, the input data are supposed to have a zero mean and are not centered, it is ignored if keyword norm is set.
- *TabStatName* : Output IDL table of string = ["Sigma", "Skewness", "Kurtosis", "Min", "Max", "HC1", "HC2", "CUMULANT ORDER 5", "CUMULANT ORDER 6"]
- *TabCumulant* : IDL double array [0:5]: 6 first cumulants of Data (TabCumulant[c] = cumulant of order c+1)

Examples:

- TabStat = get_stat(Data, /verb)
  Compute statistical information about the data set Data.

8.8.2 Compute several statistics on the wavelet coefficients : mrs_wtstat

Return statistical information relative to the wavelet transform of a given data set. The return value is a 2D IDL array of 9 elements $\times$ Number of scales. For each scale j, we have:

```
Tab[0,j] = standard deviation
Tab[1,j] = skewness
Tab[2,j] = Kurtosis
Tab[3,j] = Min
Tab[4,j] = Max
Tab[5,j] = HC
Tab[6,j] = HC^+
Tab[7,j] = Cumulant of order 5
Tab[8,j] = Cumulant of order 6

If TabFile is set, then the statistic is computed on a set of images.
Tab[*,*,f] will be the statistic related to the file TabFile[f]
```

USAGE: TabStat = mrs_wtstat(Imag, NbrScale=NbrScale, undec=undec, TabAllSurvStat=TabAllSurvStat, wt=wt, TabStatName=TabStatName, TabSurvNu=TabSurvNu, survival=survival, verb=verb, TabFile=TabFile, TabSurvStat=TabSurvStat)

where

- *Imag* : IDL array of Healpix map. Input image to analyze.
- *NbrScale* : int = Number of scales. Default is 4.
- *undec* : scalar, if set, use an undecimated WT instead of the pyramidal WT.
- *verb* : scalar, if set, the calculated statistics are printed on the screen.
- *TabFile* : Input IDL table of string, list of file where the function read the maps to be analized, in that case, on output, Imag is the last map that had been proceed and the return value is a 3D array:
  Tab[i,j,f] statistic i for scale j and map TabFile[f].
- *survival* : scalar, if set, use the survival function and activate the keywords parameters TabSurvStat, TabAllSurvStat and TabSurvNu for the results.
- *TabStatName* : Output IDL table of string, TabStatName = ["Sigma", "Skewness", "Kurtosis", "Min", "Max", "HC1", "HC2", "CUMULANT ORDER 5", "CUMULANT ORDER 6"]
- *wt* : Output IDL structure, wavelet transform of the data (see mrs_wttrans mrs_pwttrans), if TabFile keyword is used, the last map proceed.
- *TabSurvStat* : Output 2D float array [*,j] survival value at scale j.
- *TabAllSurvStat* : Output 3D double array, use together with TabFile keyword, TabAllSurvStat[*,*,f] is TabSurvStat parameter for map TabFile[f].
- *TabSurvNu* : Output 2D float array [*,j] nu survival value at scale j.

Examples:

- TabStat = mrs_wtstat(Imag, NbrScale=5, /verb)
  Compute the pyramidal wavelet transform with 5 scales and compute statistical information relative to each scale of the wavelet transform.

8.8.3 Compute several statistics on the wavelet coefficients : mrs_owtstat

Return statistical information relative to the bi-orthogonal wavelet transform of a given data set. The return value is a 2D IDL array of 9 elements × (Number of scales - 1)*(d + 1) with d = 0, 1, 2 for the 3 directions or only d = 0 if the keyword isotropic is set. The coarserst scale is not used. For each scale j, we have:

```
Tab[0,j*(d+1)] = standard deviation
Tab[1,j*(d+1)] = skewness
Tab[2,j*(d+1)] = Kurtosis
Tab[3,j*(d+1)] = Min
Tab[4,j*(d+1)] = Max
Tab[5,j*(d+1)] = HC
Tab[6,j*(d+1)] = HC^+
Tab[7,j*(d+1)] = Cumulant order 5
Tab[8,j*(d+1)] = Cumulant order 6

with : d = 0 : horizontal band, d = 1 : vertical band, d = 2 : diagonal band

If TabFile is set, then the statistic is computed on a set of images.
Tab[*,*,f] will be the statistic related to the file TabFile[f]
```

USAGE: TabStat = mrs_owtstat(Imag, TabStatName=TabStatName, NbrScale=NbrScale, verb=verb, TabFile=TabFile, isotropic=isotropic, TabAllSurvStat=TabAllSurvStat, TabSurvStat=TabSurvStat, TabSurvNu=TabSurvNu, survival=survival)

where

- *Imag* : IDL array of Healpix map. Input image to analyze.
- *NbrScale* : int = Number of scales. Default is 4.
- *verb* : scalar, if set, the calculated statistics are printed on the screen.
- *isotropic* : scalar, if set, directional information is not taken into account.
- *TabFile* : Input IDL table of string, list of file where the function read the maps to be analized, in that case, on output, Imag is the last map that had been proceed and the return value is a 3D array:
  Tab[i,j,f] statistic i for scale j and map TabFile[f].
- *survival* : scalar, if set, use the survival function and activate the keywords parameters TabSurvStat, TabAllSurvStat and TabSurvNu for the results.
- *TabStatName* : Output IDL table of string, TabStatName = ["Sigma", "Skewness", "Kurtosis", "Min", "Max", "HC1", "HC2", "CUMULANT ORDER 5", "CUMULANT ORDER 6"]
- *TabSurvStat* : Output 2D float array [*,j] survival value at scale j.
- *TabAllSurvStat* : Output 3D double array, use together with TabFile keyword, TabAllSurvStat[*,*,f] is TabSurvStat parameter for map TabFile[f].
- *TabSurvNu* : Output 2D float array [*,j] nu survival value at scale j.

Examples:

- TabStat = mrs_owtstat(Data, NbrScale=5, /verb)
  Compute the bi-orthogonal wavelet transform with 5 scales and compute statistical information relative to each scale and each direction of the wavelet transform.

8.8.4 Compute several statistics on the ridgelet coefficients : mrs_ridstat

Return statistical information relative to the ridgelet transform of a given data set. The return value is a 2D IDL array of 9 elements × Number of scales. For each scale j, we have:

```
Tab[0,j] = standard deviation
Tab[1,j] = skewness
Tab[2,j] = Kurtosis
Tab[3,j] = Min
Tab[4,j] = Max
Tab[5,j] = HC
Tab[6,j] = HC^+
Tab[7,j] = Cumulant of order 5
Tab[8,j] = Cumulant of order 6

If TabFile is set, then the statistic is computed on a set of images.
Tab[*,*,f] will be the statistic related to the file TabFile[f]
```

If the keyword NormMad is set, the ridgelet coefficients are first normalized (see *mrs_ridget*).

USAGE: TabStat = mrs_ridstat(Data, TabStatName=TabStatName, NbrScale=NbrScale, BlockSize=BlockSize, NormMad=NormMad, verb=verb, Ridtrans=Ridtrans, survival=survival, TabFile=TabFile, TabSurvStat=TabSurvStat, TabAllSurvStat=TabAllSurvStat, TabSurvNu=TabSurvNu)

where

- *Data* : IDL array of Healpix map = Input data to analyze.
- *NbrScale* : int = Number of scales. Default value is automatically calculated.
- *BlockSize* : int = Block size used in the ridgelet transform. By default, BlockSize=nside/2.
- *verb* : scalar, if set, the calculated statistics are printed on the screen.
- *NormMad* : scalar, if set, a normalization is applied to the curvelet coefficient.
- *TabFile* : Input IDL table of string, list of file where the function read the maps to be analized, in that case, on output, Imag is the last map that had been proceed and the return value is a 3D array:
  Tab[i,j,f] statistic i for scale j and map TabFile[f].

- *Ridtrans* : IDL structure containing the ridgelet transform of Data.
- *survival* : scalar, if set, use the survival function and activate the keywords parameters TabSurvStat, TabAllSurvStat and TabSurvNu for the results.
- *TabStatName* : Output IDL table of string, TabStatName = ["Sigma", "Skewness", "Kurtosis", "Min", "Max", "HC1", "HC2", "CUMULANT ORDER 5", "CUMULANT ORDER 6"]
- *TabSurvStat* : Output 2D float array [*,j] survival value at scale j.
- *TabAllSurvStat* : Output 3D double array, use together with TabFile keyword, TabAllSurvStat[*,*,f] is TabSurvStat parameter for map TabFile[f].
- *TabSurvNu* : Output 2D float array [*,j] nu survival value at scale j.

Examples:

- TabStat = mrs_ridstat(Data, NbrScale=4, /verb)
  Compute the ridgelet transform with 4 scales and compute statistical information relative to each scale of the wavelet transform.

8.8.5 Compute several statistics on the curvelet coefficients : mrs_curstat

Return statistical information relative to the pyramidal curvelet transform of a given data set. The return value is a 2D IDL array of 9 elements × Number of scales. For each scale j, we have:

```
Tab[0,j] = standard deviation
Tab[1,j] = skewness
Tab[2,j] = Kurtosis
Tab[3,j] = Min
Tab[4,j] = Max
Tab[5,j] = HC
Tab[6,j] = HC^+
Tab[7,j] = Cumulant of order 5
Tab[8,j] = Cumulant of order 6

If TabFile is set, then the statistic is computed on a set of images.
Tab[*,*,f] will be the statistic related to the file TabFile[f]
```

If the keyword NormMad is set, the curvelet coefficients are first normalized (see *mrs_ridget*).

USAGE: TabStat = mrs_curstat(Data, NbrScale=NbrScale, verb=verb, Firstblocksize=Firstblocksize, NbrScale=NbrScale, normMad=normMad, TabFile=TabFile, TabStatName=TabStatName, survival=survival, TabSurvStat=TabSurvStat, TabAllSurvStat=TabAllSurvStat, TabSurvNu=TabSurvNu)

where

- *Data* : IDL array of Healpix map, Input data to analyze.
- *NbrScale* : int = Number of scales. Default is 4.
- *verb* : scalar, if set, the calculated statistics are printed on the screen.
- *NormMad* : scalar, if set, a normalization is applied to the curvelet coefficient.
- *Firstblocksize:* : int, First block size used in the curvelet transform.
- *TabFile* : Input IDL table of string, list of file where the function read the maps to be analized, in that case, on output, Imag is the last map that had been proceed and the return value is a 3D array:
  Tab[i,j,f] statistic i for scale j and map TabFile[f].
- *survival* : scalar, if set, use the survival function and activate the keywords parameters TabSurvStat, TabAllSurvStat and TabSurvNu for the results.
- *TabStatName* : Output IDL table of string, TabStatName = ["Sigma", "Skewness", "Kurtosis", "Min", "Max", "HC1", "HC2", "CUMULANT ORDER 5", "CUMULANT ORDER 6"]
- *TabSurvStat* : Output 2D float array [*,j] survival value at scale j.
- *TabAllSurvStat* : Output 3D double array, use together with TabFile keyword, TabAllSurvStat[*,*,f] is TabSurvStat parameter for map TabFile[f].
- *TabSurvNu* : Output 2D float array [*,j] nu survival value at scale j.

Examples:

- TabStat = mrs_curstat(Data, NbrScale=4, /verb)
  Compute the pyramidal curvelet transform with 4 scales and compute statistical information relative to each scale of the wavelet transform.

8.8.6 Compute several statistics on wavelet, ridgelet and curvelet coefficients : mrs_allstat

Return statistical information relative to several multiscales transforms of a given data set. The six used multiscale transforms are: the pyramidal wavelet transform, the isotropic undecimated wavelet transform, the ridgelet transform with a block size equals to 8, the ridgelet transform with a block size equals to 16, the ridgelet transform with a block size equals to 32 and the pyramidal curvelet transform.

All statistical informations are computed with the survival option. The return value is a IDL structure with several fields for six transforms statistics:

Each of the statistics fields is a 2D IDL array of 9 elements $\times$ Number of scales and for each scale j, we have:

```
Tab[0,j] = standard deviation
Tab[1,j] = skewness
Tab[2,j] = Kurtosis
Tab[3,j] = Min
Tab[4,j] = Max
Tab[5,j] = HC
Tab[6,j] = HC^+
Tab[7,j] = Cumulant of order 5
Tab[8,j] = Cumulant of order 6

If TabFile is set, then the statistic is computed on a set of images.
Tab[*,*,f] will be the statistic related to the file TabFile[f]
```

USAGE: StatData = mrs_allstat(Imag, NbrScale2D=NbrScale2D, TabFile=TabFile, TabStatName=TabStatName, normMad=normMad, verb=verb, iwt=iwt, owt=owt, rid8=rid8, rid16=rid16, rid32=rid32, cur=cur, all=all, save=save, TabTransformName=TabTransformName)

where

- *StatData* : IDL structure with the following fields:
  - *OWT* : IDL float array [9, NbrScale], statistics of the Orthogonal Wavelet Transform.
  - *OWTSurv* : TabSurvStat parameter for Orthogonal Wavelet Transform.
  - *OWTSurvNu* : TabSurvNu parameter for Orthogonal Wavelet Transform.
  - *IWT* : IDL float array [9, NbrScale], statistics of the Isotropic Undecimated Wavelet Transform.
  - *IWTSurv* : TabSurvStat parameter for Isotropic Undecimated Wavelet Transform.
  - *IWTSurvNu* : TabSurvNu parameter for Isotropic Undecimated Wavelet Transform.
  - *Rid8* : IDL float array [9, NbrScale], statistics of the Ridgelet Transform (Length=8).
  - *Rid8Surv* : TabSurvStat parameter for Ridgelet Transform (Length=8).
  - *Rid8SurvNu* : TabSurvNu parameter for Ridgelet Transform (Length=8).
  - *Rid16* : IDL float array [9, NbrScale], statistics of the Ridgelet Transform (Length=16).
  - *Rid16Surv* : TabSurvStat parameter for Ridgelet Transform (Length=16).
  - *Rid16SurvNu* : TabSurvNu parameter for Ridgelet Transform (Length=16).
  - *Rid32* : IDL float array [9, NbrScale], statistics of the Ridgelet Transform (Length=32).
  - *Rid32Surv* : TabSurvStat parameter for Ridgelet Transform (Length=32).
  - *Rid32SurvNu* : TabSurvNu parameter for Ridgelet Transform (Length=32).

  - *Cur* : IDL float array [9, NbrScale], statistics of the Curvelet Transform.
  - *CurSurv* : TabSurvStat parameter for Curvelet Transform.
  - *CurSurvNu* : TabSurvNu parameter for Curvelet Transform.
  - *TabStatName* : IDL table of string: TabStatName = ["Sigma", "Skewness", "Kurtosis", "Min", "Max", "HC1", "HC2", "CUMULANT ORDER 5", "CUMULANT ORDER 6"]
  - *TabTransformName* : IDL table of string: TabTransformName=['Orthogonal Wavelet', 'Isotropic Undecimated Wavelet', 'Ridgelet Transform (Length=8)', 'Ridgelet Transform (Length=16)', 'Ridgelet Transform (Length=32)', 'Curvelet']
  - If a transform is not chosen, the three corresponding parameters fields are set to 0.

- *Imag* : IDL array of Healpix map. Input image to analyze.
- *NbrScale2D* : int = Number of scales. Default is 4.
- *verb* : scalar, if set, the calculated statistics are printed on the screen.
- *NormMad* : scalar, if set, a normalization is applied to the ridgelet and curvelet coefficients.
- *TabFile* : Input IDL table of string, list of file where the function read the maps to be analized, in that case, on output, Imag is the last map that had been proceed and the return value is a 3D array:
  Tab[i,j,f] statistic i for scale j and map TabFile[f].
- *iwt* : scalar, if set, the statistics of the Isotropic Undecimated Wavelet Transform are computed.
- *owt* : scalar, if set, the statistics of the Orthogonal Wavelet Transform are computed.
- *rid8* : scalar, if set, the statistics of the Ridgelet Transform (Length=8) are computed.
- *rid16* : scalar, if set, the statistics of the Ridgelet Transform (Length=16) are computed.
- *rid32* : scalar, if set, the statistics of the Ridgelet Transform (Length=32) are computed.
- *cur* : scalar, if set, the statistics of the Curvelet Transform are computed.
- *all* : scalar, if set, the statistics of all the 6 transforms are computed.
- *save* : scalar, if set, the results are saved in separate files.
- *TabStatName* : Output IDL table of string, TabStatName = ["Sigma", "Skewness", "Kurtosis", "Min", "Max", "HC1", "HC2", "CUMULANT ORDER 5", "CUMULANT ORDER 6"]
- *TabTransformName* : Output IDL table of string: TabTransformName=['Orthogonal Wavelet', 'Isotropic Undecimated Wavelet', 'Ridgelet Transform (Length=8)', 'Ridgelet Transform (Length=16)', 'Ridgelet Transform (Length=32)', 'Curvelet']

Examples:

- TabStat = mrs_allstat(Data, NbrScale2D=4, /verb, /all)
  Compute the six transforms with 4 scales and compute statistical information relative to each scale and transform.

8.9 Tutorial

8.9.1 Undecimated Wavelet Transform on the Sphere

The code to generate the undecimated wavelet transform of the Mars image of Fig. 3.8 is as follows.

```
; read the data
m = mrs_read('mars_topo_mola_hpx_128.fits')

; compute the undecimated wavelet transform with 5 scales
mrs_wttrans, m, w,nbrscale=5

; Display and write the figures to the disk
tvs, m, tit='Mars topographic map', png='fig_mars.png'
tvs, w.coef[*,0], tit='Mars topographic map: scale 1', $
                   png='fig_mars_scale1.png'
tvs, w.coef[*,1], tit='Mars topographic map: scale 2', $
                   png='fig_mars_scale2.png'
tvs, w.coef[*,2], tit='Mars topographic map: scale 3', $
                   png='fig_mars_scale3.png'
tvs, w.coef[*,3], tit='Mars topographic map: scale 4', $
                   png='fig_mars_scale4.png'
tvs, w.coef[*,4], tit='Mars topographic map: scale 5', $
                   png='fig_mars_scale5.png'
```

8.9.2 Pyramidal Wavelet Transform on the Sphere

The code to generate the pyramidal wavelet transform of the Mars image of Fig. 3.11 is as follows.

```
; read the data
e = mrs_read('earth_healpix_128.fits')

; compute the pyramidal wavelet transform with 5 scales
mrs_pwttrans, e, we, nbrscale=5

; Display and write the figures to the disk
mrs_wttv, we, write='fig_earth'
```

Denoising

In the denoising experiment of Fig. 4.1 and Fig. 4.2, we have added Gaussian noise to the astronomical simulated synchrotron emission map. The code to generate the figures is as follows.

```
; read the image
s = rims('sync_res128.fits')

; add Gaussian noise
n = randomn(seed, N_ELEMENTS(s))
SigmaNoise = 5.
s1 = s + n* SigmaNoise

; Denoising using the undecimated WT on the sphere at 4sigma
Nsig = 4.
 mrs_wtfilter, s1, fwt4, nsigma= Nsig, nbrscale=5, SigmaNoise=SigmaNoise

; Denoising using the curvelet transform
mrs_curfilter, s1, fct4, nsigma= Nsig, nbrscale=5, SigmaNoise=SigmaNoise

; Denoising using the combined denoising
mrs_cbfilter, s1, fcb4, nsigma= Nsig, nbrscale=5, SigmaNoise=SigmaNoise

; Display and write the figure to the disk
tvs, s, /log, tit='Synchrotron emission', png='fig_sync.png'
tvs, s1 > 30, /log, tit='Synchrotron emission + noise', $
                   png='fig_sync_noise5.png'

tvs , fwt4 > 30, /log, title='Undecimated Wavelet Denoising (4sigma)', $
                    png='fig_sync_wtfilter5.png'
tvs , fct4 > 30, /log, title='Curvelet Denoising (4sigma)',  $
                    png='fig_sync_curfilter5.png'
tvs , fcb4 > 30, /log, title='Combined Filtering (4sigma)',
                    png='fig_sync_cbfilter5.png'

tvs , s1- fwt4,   title='Residual undecimated Wavelet Denoising (4sigma)', $
                    png='fig_sync_resi_wtfilter5.png'
tvs , s1 - fct4,  title='Residual  curvelet Denoising (4sigma)',  $
                    png='fig_sync_resi_curfilter5.png'
tvs , s1  - fcb4,  title='Residual  combined Filtering (4sigma)',  $
                    png='fig_sync_resi_cbfilter5.png'

; Print the standard deviation (error) between the true image
; and the denoised images
print, 'Err WT = ', sigma(s-fwt4) , ', Err Cur  = ', sigma(s-fct4),  ',  $
                  Err Comb = ', sigma(s-fcb4)
```

We find the outcome here to be:

```
==>  Err WT = 1.25,  Err Cur  = 1.07,  Err Combined = 0.86
```

Part II

SparsePOL/V1.1 : Polarized Spherical Wavelets and Curvelets

Chapter 9

Polarized Data

9.1 Introduction

Polarized maps are a special kind of multi-dimentionnal data with strong links between their components. The polarization is of great importance in physics or astrophysics as it's analysis denotes fundamentals characteristics of the observed object or phenomenum. An example of great interest is today the cosmic microwave background (Zaldarriaga 1998; Kovac et al. 2002).

The statistical analysis of the slight intensity fluctuations in the primordial cosmic microwave background radiation field, for which evidence was found for the first time in the early 1990's in the observations made by COBE (Smoot et al. 1992), is a major issue in modern cosmology as these are strongly related to the cosmological scenarios describing the properties and evolution of our Universe. In the Big Bang model, the observed CMB anisotropies are an imprint of primordial fluctuations in baryon-photon density from a time when the temperature of the Universe was high enough above 3000 K for matter and radiation to be tightly coupled. At that time, the attraction of gravity and the repulsive radiation pressure were opposed, thus generating so-called acoustic oscillations in the baryon-photon fluid, causing peaks and troughs to appear in the power spectrum of the spatial anisotropies of the CMB. With the Universe cooling down as it expanded, matter and radiation finally decoupled. Photons were set free in a nearly transparent Universe, while the density fluctuations collapsed under the effect of gravity into large scale structures such as galaxies or clusters of galaxies. Due to the expansion of the Universe, the CMB photons are now observed in the microwave range but are still distributed according to an almost perfect black body emission law. Another major result was the measurement of the polarization state and anisotropies of the CMB radiation field by DASI (Kovac et al. 2002). Only a fraction of the total CMB radiation is polarized so that extremely sensitive instruments are needed. Polarization of the CMB radiation is a consequence of the Thomson scattering of photons on electrons. But for the outgoing population of photons to be polarized, the radiation incident on the scatterer needs to be anisotropic and have a quadrupole moment. The statistics of the CMB polarization anisotropies are also a source of information for cosmology. Inference of cosmological parameters from the joint statistics of the CMB anisotropies should benefit from both the complementarity and the redundancy of the information carried by the additional measurement of CMB polarization. Hence the full-sky maps with unprecedented sensitivity and angular reso-

lution of both temperature and polarization anisotropies of the CMB to be delivered by the upcoming Planck Surveyor satellite experiment are awaited with excitement.

9.2 Representation of polarized data

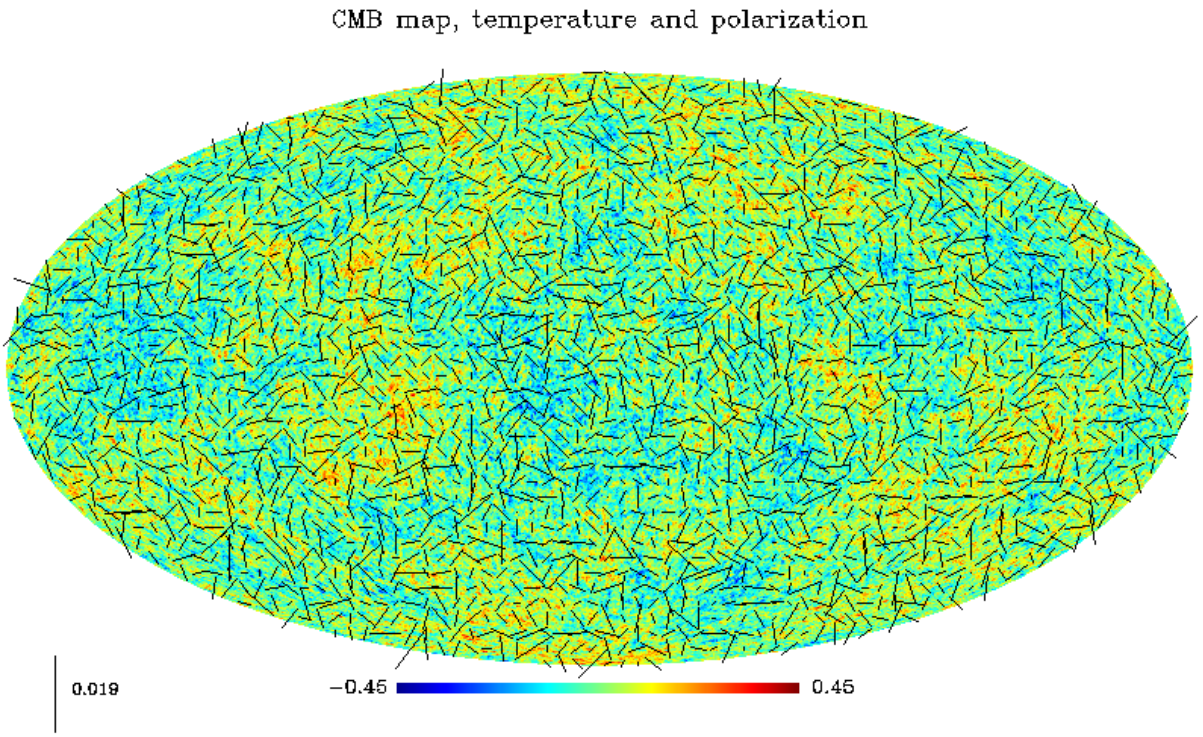


Figure 9.1: Simulated CMB map, temperature and polarization.

Full-sky CMB polarization data consists of measurements of the Stokes parameters so that in addition to the temperature T map, Q and U maps are given as well. The fourth Stokes parameter commonly denoted V is a measure of circular polarization. In the case of CMB which is not expected to have circularly polarized anisotropies, V vanishes. The former three quantities, T, Q and U then fully describe the linear polarization state of the CMB radiation incident along some radial line of sight : T is the total incoming intensity, Q is the difference between the intensities transmitted by two perfect orthogonal polarizers the directions of which define a reference frame in the tangent plane, and U is the same as Q but with polarizers rotated 45 degrees in that tangent plane. Clearly, Q and U are not invariant through a rotation of angle ϕ of the local reference frame around the line of sight. In fact, it is easily shown that :

$$\begin{aligned} Q' &= \cos(2\phi)Q + \sin(2\phi)U \\ U' &= \cos(2\phi)U - \sin(2\phi)Q \end{aligned} \tag{9.1}$$

which can also be written $Q' \pm iU' = e^{\mp i\phi}(Q \pm iU)$ which by definition expresses the fact that the quantities $Q \pm iU$ are spin-2 fields on the sphere. The suitable generalization of

the Fourier representation for such fields is the spin-2 spherical harmonics basis denoted ${}_{\pm 2}Y_{\ell m}$, in which we can expand :

$$Q \pm iU = \sum_{\ell,m} {}_{\pm 2}a_{\ell m \pm 2} Y_{\ell m} \tag{9.2}$$

Fig. 9.1 show an example of CMB polarized map.

Chapter 10

Multiscale Methods for polarized maps on the Sphere

10.1 Multiscale Representation

The easiest way to build a multiscale transform for polarized data is to use the Healpix[1] representation (Górski et al. 2005), and to apply a bi-orthogonal wavelet transform on each face of the Healpix map, separately for Q and U (Starck et al. 2009b). Fig. 10.1 shows the flow-graph of this Q-U orthogonal wavelet transform (QU-OWT). Recall that the base resolution of the Healpix representation divides the sphere into twelve curvilinear quadrilateral faces of equal area placed on three rings around the poles and equator. Each face is subsequently divided into $nside^2$ pixels of exactly equal surface but with varying shape. It follows that Q and U are reconstructed at position k from their wavelet coefficients $w^Q_{j,p}$, $w^U_{j,p}$, $c^Q_{J,p}$ and $c^U_{J,p}$ according to :

$$\begin{aligned} Q_k &= \sum_p c^Q_{J,p}\phi_{j,k}(p) + \sum_p \sum_{j=1}^{J} \psi_{j,k}(p) w^Q_{j,p} \\ U_k &= \sum_p c^U_{J,p}\phi_{j,k}(p) + \sum_p \sum_{j=1}^{J} \psi_{j,k}(p) w^U_{j,p} \end{aligned} \tag{10.1}$$

which can also be written as:

$$\begin{aligned} (Q \pm iU)_k &= \sum_p (c^Q_{J,p} \pm i c^U_{J,p})\phi_{j,k}(p) + \\ &\quad \sum_p \sum_{j=1}^{J} \psi_{j,k}(p)(w^Q_{j,p} \pm i w^U_{j,p}) \end{aligned} \tag{10.2}$$

This wavelet transform is not redundant *i.e.* the decomposition has the same number of coefficients as the input data, and it is invertible so that the Q and U maps can be reconstructed exactly.

[1] http://healpix.jpl.nasa.gov

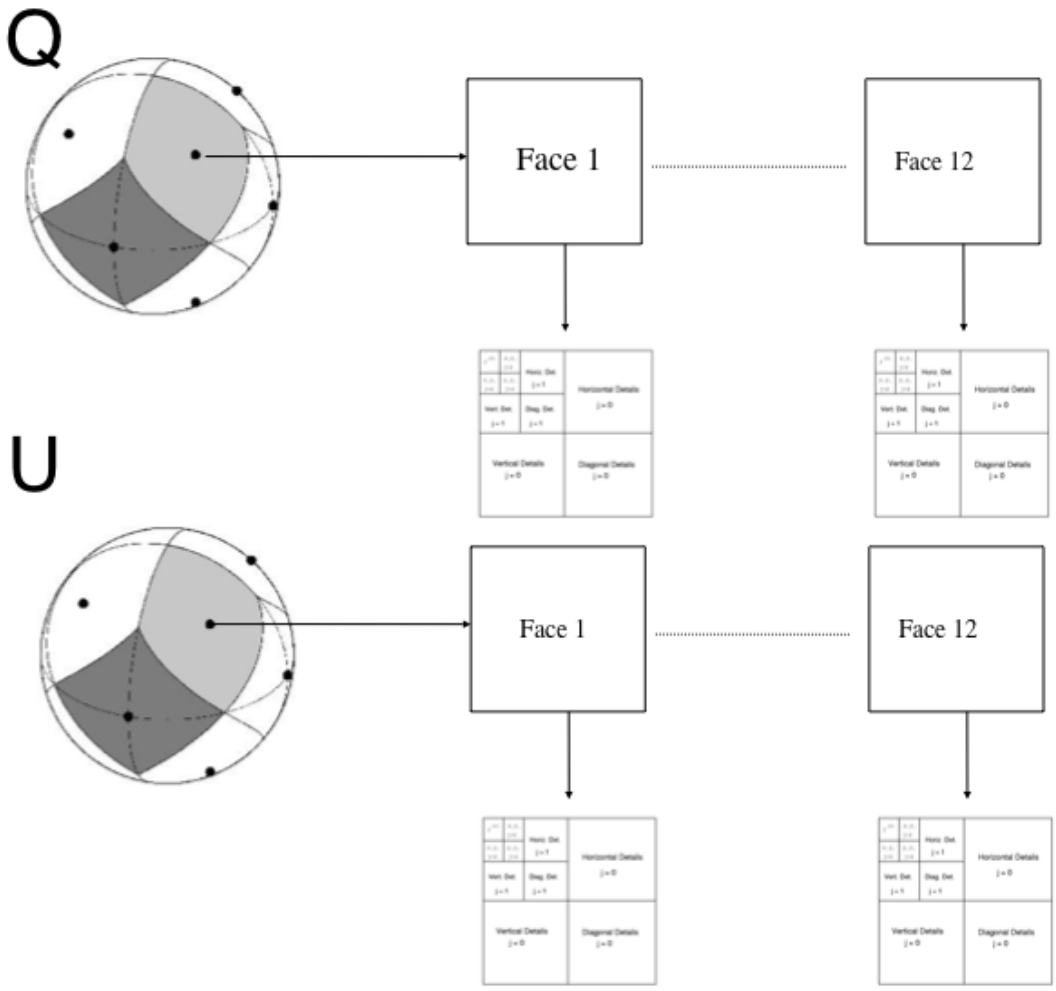


Figure 10.1: Q-U orthogonal Wavelet Transform.

When we apply such a decomposition, we implicitly use a dictionary Φ on which we project the data. As discussed previously, the shape of the dictionary elements, also called atoms, is very important to have an efficient analysis of the data. In the case of polarized data, it is not straightforward to imagine these shapes from Eq. (10.2). In order to visualize them, we can perform a backprojection *i.e.* we apply the inverse wavelet transform to sets of wavelet coefficients where only one coefficient is different from zero. Repeating the same experiment, changing only the scale and position of the non-zero coefficient allows us to view the different atoms in the dictionary related to the QU-OWT transform that we use. Fig. 10.2 shows examples of backprojections of Q wavelet coefficients (top) and backprojections of U wavelet coefficients (bottom). The shapes of the individual atoms do not look close to the astronomical patterns we would expect in our data. Therefore, this decomposition may not be optimal to analyze polarized astronomical data, although this would need to be confirmed in practice. The following sections describe other polarized wavelet transforms with different morphological properties.

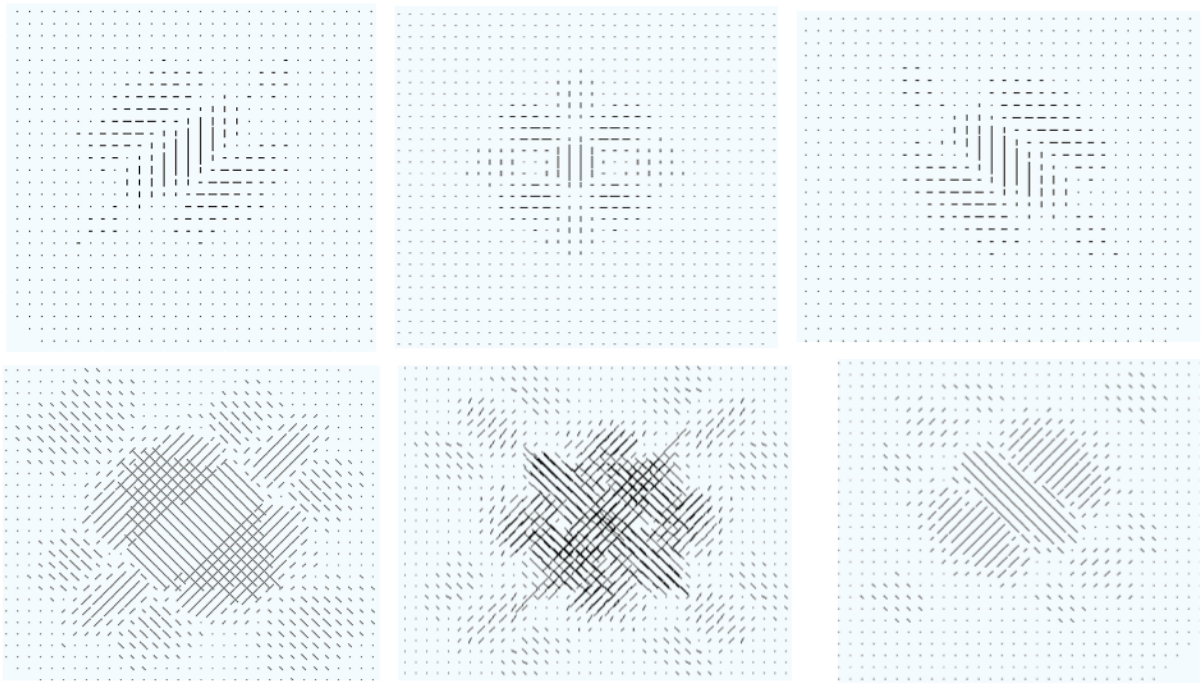


Figure 10.2: *Top :* examples of backprojections of Q-wavelet coefficients. *Bottom :* examples of backprojections of U-wavelet coefficients.

10.2 Module-phase non linear multiscale transform

10.2.1 Introduction

Given a polarized map in the standard Q-U representation, consider a different point of view and define the modulus M and phase P maps as follows :

$$\forall k, \quad M_k = \sqrt{Q_k^2 + U_k^2} \tag{10.3}$$

$$\forall k, \quad P_k = \exp(i\theta_k) \text{ where } tan(\theta_k) = U_k/Q_k \tag{10.4}$$

Because the smoothness of the Q and U maps should result in some smoothness of the modulus map M and the phase map P, one may consider devising a multiscale modulus/phase decomposition of the spin 2 field $\mathcal{V} = [Q\,U]$.

The specificity of the modulus/phase decomposition of $\mathcal{V}$ is twofold : i) the modulus field is non-negative and ii) the phase field takes its values on the unit circle S^1. Recently, (Rahman et al. 2005) introduced a multiscale analysis technique for manifold valued data that will be described in the following paragraph. We then define the modulus/phase (MP) multiscale transform as follows (Starck et al. 2009b):

1. Apply a classical multiscale transform (*i.e.* wavelets) to the modulus map M.
2. Apply the multiscale analysis technique for manifold valued data described in (Rahman et al. 2005) to the phase map P.

10.2.2 Decimated MP-multiscale transform

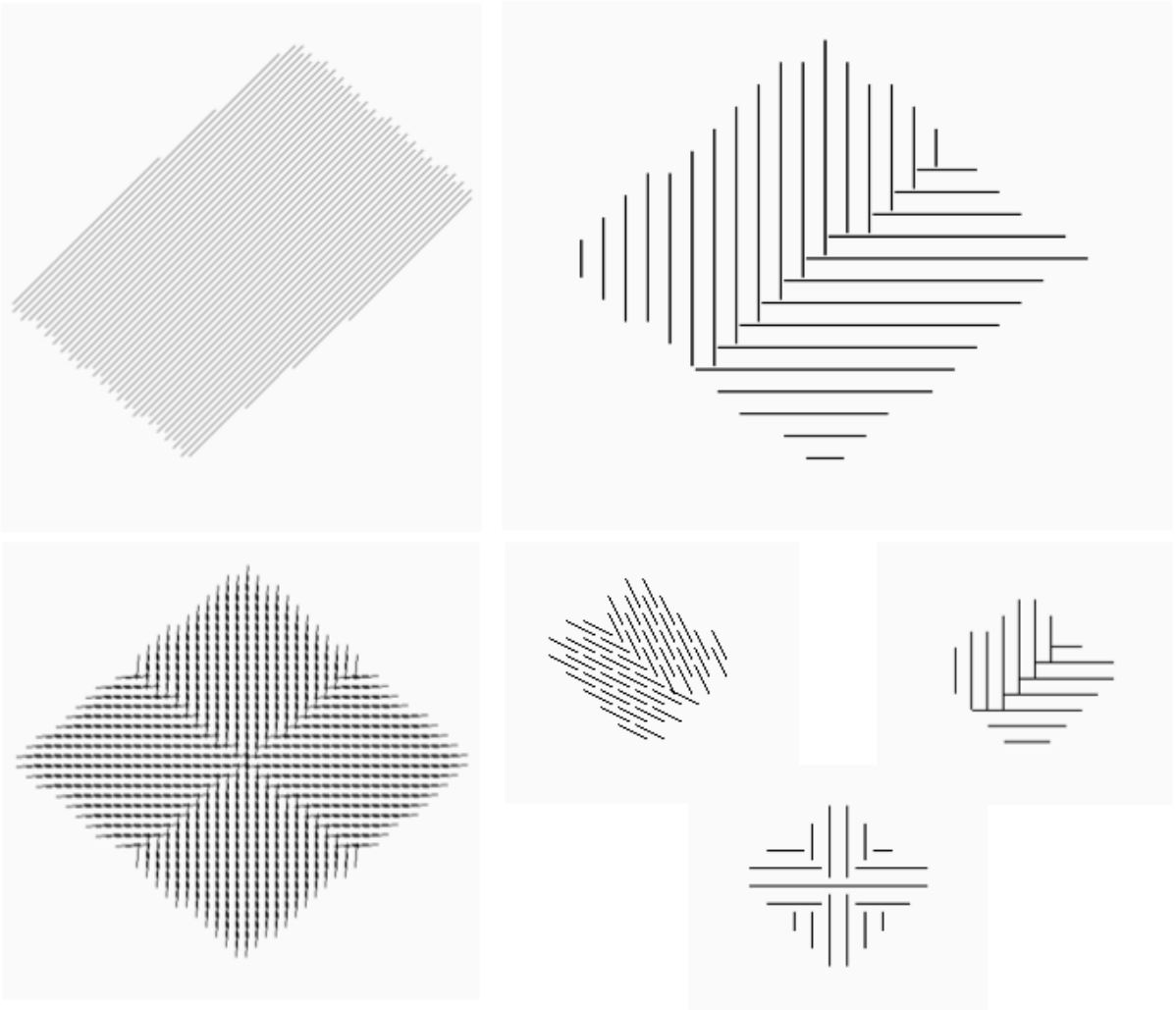


Figure 10.3: Examples of MP-multiscale coefficients backprojection.

Let us provide some essential notation : we assume that the entries of the phase map P lie in a manifold $\mathcal{M}$ (*e.g.* $\mathcal{M} \equiv S^1$). According to (Rahman et al. 2005), take $p_0, p_1 \in \mathcal{M}$ and define $Log_{p_0}(p_1)$ as the log-map of p_1 onto the tangent space $\mathcal{T}_{p_0}$ of $\mathcal{M}$ at p_0. The back-projection is obtained using the inverse of the log-map Exp_{p_0}. [2]
For instance, if we choose $\mathcal{M} \equiv S^1$ then $p_0 = \exp(i\theta_0)$ and $p_1 = \exp(i\theta_1)$. The Exp_{p_0} and Log_{p_0} maps are then defined as follows :

$$\forall p_1 \in S^, \ Log_{p_0}(p_1) = \theta_1 - \theta_0 \quad (10.5)$$

$$\forall s \in \mathbb{R} \ Exp_{p_0}(s) = exp(i(\theta_0 + s)) \quad (10.6)$$

[2] In differential geometry, the Exp map and Log map are generalizations of the usual exponential and logarithm function. Here the manifold $\mathcal{M}$ is a Riemannian manifold. In that case, the Exp map at point p_0, $Exp_{p_0}(s)$ is the map which takes a vector s of the tangent space of $\mathcal{M}$ at p_0 and provides the point p_1 by travelling along the geodesic starting at p_0 in the direction s.

The multiscale transform for manifold valued data introduced in (Rahman et al. 2005) is equivalent to a two-step interpolation-refinement scheme similar to the lifting scheme described in (Daubechies and Sweldens 1998). The wavelet coefficients and low pass approximation pixels are then computed as follows at each scale j and pixel k :

$$w^P_{j+1,k} = Log_{c^P_{j,2k+1}}\left(\mathcal{P}(c^P_{j,2k})\right) \tag{10.7}$$

$$c^P_{j+1,k} = Exp_{c^P_{j,2k}}(-\mathcal{U}(w^P_{j+1,k})) \tag{10.8}$$

The wavelet coefficient $w^P_{j+1,k}$ at pixel k and scale j is the projection of its prediction/interpolation $\mathcal{P}(c^P_{j,2k})$ onto the tangent space $\mathcal{T}_{c^P_{j,2k+1}}$ of $\mathcal{M}$ at $c^P_{j,2k+1}$. The low pass approximation $c^P_{j+1,k}$ at scale $j+1$ is computed by updating $c^P_{j,2k}$ from the wavelet coefficient $w^P_{j+1,k}$.

The main advantage of this scheme is its ability to capture local regularities while guaranteeing the low pass approximation to belong to the manifold $\mathcal{M}$. Indeed, the wavelet coefficient $w^P_{j+1,k}$ at pixel k and scale $j+1$ is computed as the Exp map at $c^P_{j,2k+1}$ of an approximation $\mathcal{P}(c^P_{j,2k})$ of $c^P_{j,2k}$.

Note also that even if the definitions of the Exp_{p_0} and Log_{p_0} maps involve the absolute phase $\theta(k)$ (*i.e.* $tan(\theta(k)) = U_k/Q_k$), at least they only require the computation of differences of phases values thus avoiding the explicit manipulation of an absolute phase. However the non-linearity of the proposed transform is a major drawback when considering denoising and restoration applications.

Illustration :

In the case of polarized data, the entries of the phase map P lie in $\mathcal{M} \equiv S^1$. In the following experiments, $\mathcal{P}$ and $\mathcal{U}$ are chosen such that :

$$w^P_{j+1} = Log_{c^P_{j,2k+1}}(c^P_{j,2k}) \tag{10.9}$$

$$c^P_{j+1,k} = Exp_{c^P_{j,2k}}\left(-\frac{w^P_{j+1}}{2}\right) \tag{10.10}$$

This multiscale transform is invertible and its inverse is computed as follows :

$$c^P_{j,2k} = Exp_{c^P_{j+1,k}}\left(\frac{w^P_{j+1}}{2}\right) \tag{10.11}$$

$$c^P_{j,2k+1} = Exp_{c^P_{j,2k}}\left(w^P_{j+1}\right) \tag{10.12}$$

The picture in Figure 10.3 features some examples of backprojections of MP-multiscale coefficients.

10.2.3 Undecimated MP-multiscale transform

For image restoration purposes, the use of undecimated multiscale transforms has been shown to provide better results than decimated transforms (Starck et al. 1998; Starck and Murtagh 2006). The aforementioned modulus/phase multiscale analysis can

be extended to an undecimated scheme consisting in : i) applying an undecimated wavelet transform to the modulus map, ii) analyzing the phase map P using an extension to the undecimated case of the multiscale transform described in (Rahman et al. 2005). In that case, Equations (10.7) and (10.8) are replaced with the following equations :

$$c^P_{j+1,k} = Exp_{c^P_{j,k}}(\mathcal{F}(c^P_{j,.})) \quad (10.13)$$

$$w^P_{j+1} = Log_{c^P_{j+1,k}}\left(c^P_{j,k}\right) \quad (10.14)$$

where $\mathcal{F}(c^P_{j,.}) = \sum_l h_l Log_{c_{j,k}}\left(c_{j,k-2^j l}\right)$ with $\sum_l h_l = 1$ and $h_l > 0$. The low pass approximation $c^P_{j+1,k}$ is then computed from a linear combination (linear filter) of a neighborhood $\{c_{j,k-2^j l}\}_l$ of $c_{j,k}$ weighted by the positive scalars $\{h_l\}_l$. Note that from scale j to scale $j+1$, the spatial size of the neighborhood increases by a factor 2 which would be equivalent to downsize by a factor 2 the band pass filter of the classical wavelet decomposition scheme.

10.2.4 Example

In the case of polarized data, the entries of the phase map P lie in $\mathcal{M} \equiv S^1$. In the following experiments, $\mathcal{F}$ is chosen such that :

$$c^P_{j+1,k} = Exp_{c^P_{j,k}}\left(\sum_l h_l Log_{c_{j,k}}\left(c^P_{j,k-2^j l}\right)\right) \quad (10.15)$$

$$w^P_{j+1,k} = Exp_{c^P_{j,k}}\left(c^P_{j+1,k}\right) \quad (10.16)$$

where :

$$h_l = \begin{cases} 0 & \text{if} \quad l < -2 \text{ or } l > 2 \\ 1/16 & \text{if} \quad l = -2 \text{ or } l = 2 \\ 1/4 & \text{if} \quad l = -1 \text{ or } l = 1 \\ 3/8 & \text{if} \quad l = 0 \end{cases} \quad (10.17)$$

This multiscale transform is invertible and its inverse is computed as follows :

$$c^P_{j,k} = Exp_{c^P_{j+1,k}}\left(-w^P_{j+1,k}\right) \quad (10.18)$$

Fig. 10.4 top shows a simulated polarized field of the synchrotron emission and its noisy version. We have applied the MP-multiscale transform and we remove the first three scales (i.e. we put all coefficients to zero) before reconstructing. The resulting image is shown on the bottom left of Fig. 10.4. The bottom right of Fig. 10.4 corresponds to the same experiment, but by removing the five first scales. We can see that the field is smoother and smoother, but respecting the large scale structure of the field. This transform will be very well suited to CMB studies where the phase is analyzed independently of the modulus, such as in (Dineen et al. 2005; Naselsky et al. 2005).

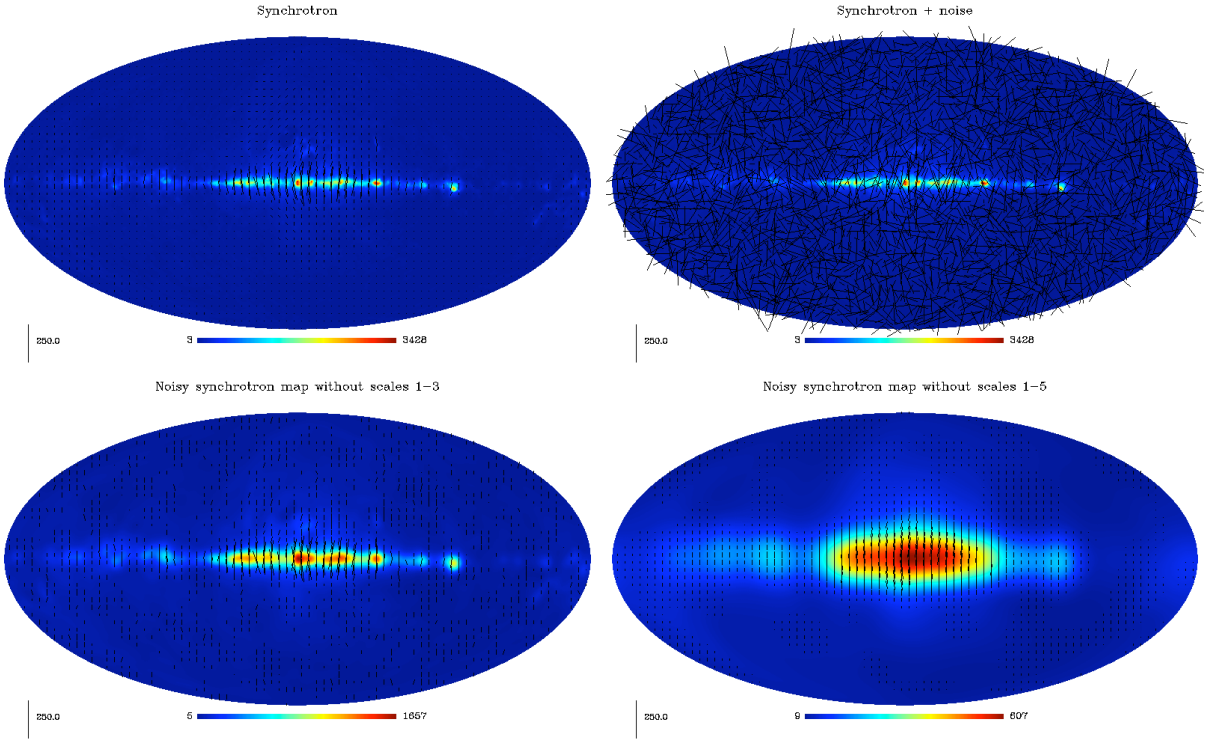


Figure 10.4: Polarized field smoothing - *top left :* simulated synchroton emission. *top right :* same field corrupted by additive noise. *bottom left :* MP-multiscale reconstruction after setting to zero all coefficients from the three first scales. *bottom right :* MP-multiscale reconstruction after setting to zero all coefficients from the five first scales.

10.3 Polarized Wavelet Transform using Spherical Harmonics

10.3.1 Isotropic Undecimated Wavelet Transform on the Sphere (UWTS)

The undecimated isotropic transform on the sphere described in (Starck et al. 2006) is similar in many respects to the usual "à trous" isotropic wavelet transform. It is obtained using a zonal scaling function $\phi_{l_c}(\vartheta, \varphi)$ which depends only on colatitude ϑ and is invariant with respect to a change in longitude φ. It follows that the spherical harmonic coefficients $\hat{\phi}_{l_c}(l, m)$ of ϕ_{l_c} vanish when $m \neq 0$ which makes it simple to compute those spherical harmonic coefficients $\hat{c}_0(l, m)$ of $c_0 = \phi_{l_c} * f$ where $*$ stands for convolution :

$$\hat{c}_0(l, m) = \widehat{\phi_{l_c} * f}(l, m) = \sqrt{\frac{4\pi}{2l+1}} \hat{\phi}_{l_c}(l, 0) \hat{f}(l, m) \tag{10.19}$$

A possible scaling function (Starck et al. 1998), defined in the spherical harmonics representation, is $\phi_{l_c}(l, m) = \frac{2}{3} B_3(\frac{2l}{l_c})$ where B_3 is the cubic B-spline compactly supported over $[-2, 2]$. Denoting $\phi_{2^{-j}l_c}$ a rescaled version of ϕ_{l_c} with cut-off frequency $2^{-j} l_c$, a multi-resolution decomposition of f on a dyadic scale is obtained recursively :

$$\begin{aligned} c_0 &= \phi_{l_c} * f \\ c_j &= \phi_{2^{-j}l_c} * f = c_{j-1} * h_{j-1} \end{aligned} \tag{10.20}$$

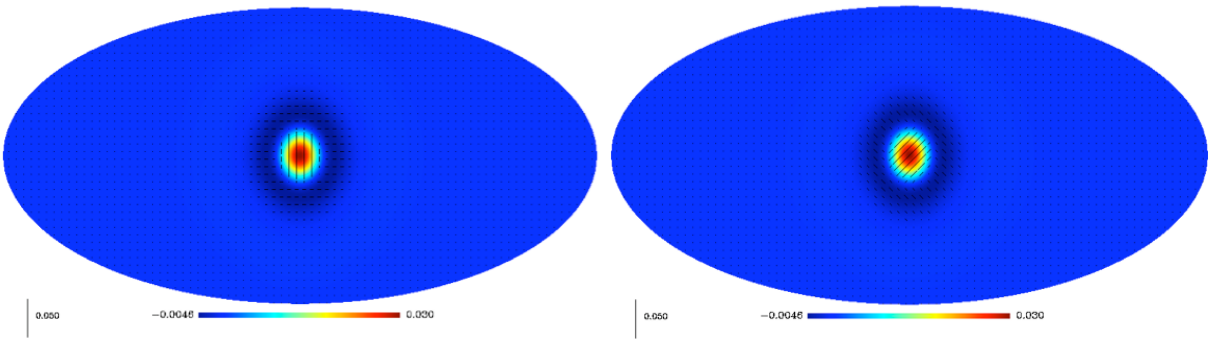


Figure 10.5: Q-isotropic wavelet transform backprojection (left) and U-isotropic wavelet backprojection (right).

where the zonal low pass filters h_j are defined by

$$\begin{aligned} \hat{H}_j(l,m) &= \sqrt{\frac{4\pi}{2l+1}}\hat{h}_j(l,m) \\ &= \begin{cases} \frac{\hat{\phi}_{\frac{l_c}{2^{j+1}}}(l,m)}{\hat{\phi}_{\frac{l_c}{2^j}}(l,m)} & \text{if } l < \frac{l_c}{2^{j+1}} \quad \text{and} \quad m = 0 \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{10.21}$$

The cut-off frequency is reduced by a factor of 2 at each step so that in applications where this is useful such as compression, the number of samples could be reduced adequately. Using a pixelization scheme such as Healpix (Górski et al. 2005), this can easily be done by dividing by 2 the Healpix *nside* parameter when computing the inverse spherical harmonics transform. As in the "à trous" algorithm, the wavelet coefficients can be defined as the difference between two consecutive resolutions, $w_{j+1}(\vartheta,\varphi) = c_j(\vartheta,\varphi) - c_{j+1}(\vartheta,\varphi)$. This defines a zonal wavelet function ψ_{l_c} as

$$\hat{\psi}_{\frac{l_c}{2^j}}(l,m) = \hat{\phi}_{\frac{l_c}{2^{j-1}}}(l,m) - \hat{\phi}_{\frac{l_c}{2^j}}(l,m) \tag{10.22}$$

With this particular choice of wavelet function, the decomposition is readily inverted by summing the coefficient maps on all wavelet scales

$$f(\vartheta,\varphi) = c_J(\vartheta,\varphi) + \sum_{j=1}^{J} w_j(\vartheta,\varphi) \tag{10.23}$$

where we have made the simplifying assumption that f is equal to c_0. Obviously, other wavelet functions ψ could be used just as well, such as the needlet function (Marinucci et al. 2008).

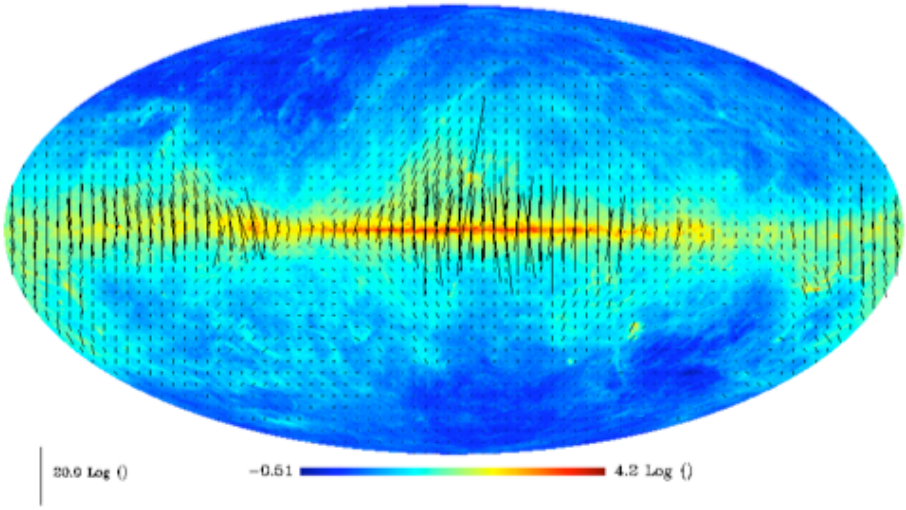


Figure 10.6: Simulated observations on the sphere of the polarized galactic dust emission.

10.3.2 Extension to Polarized Data

By applying the above scalar isotropic wavelet transform to each component T, Q, U of a polarized map on the sphere, we have (Starck et al. 2009b) :

$$\begin{aligned} T(\vartheta,\varphi) &= c_J^T(\vartheta,\varphi) + \textstyle\sum_{j=1}^{J} w_j^T(\vartheta,\varphi) \\ Q(\vartheta,\varphi) &= c_J^Q(\vartheta,\varphi) + \textstyle\sum_{j=1}^{J} w_j^Q(\vartheta,\varphi) \\ U(\vartheta,\varphi) &= c_J^U(\vartheta,\varphi) + \textstyle\sum_{j=1}^{J} w_j^U(\vartheta,\varphi) \end{aligned} \tag{10.24}$$

where c_J^X stands for the low resolution approximation to component X and w_j^X is the map of wavelet coefficients of that component on scale j. This leads to the following decomposition :

$$(Q \pm iU)[k] = (c_J^Q \pm c_{J,p}^U)[k] + \sum_{j=1}^{J} (w_j^Q \pm w_j^U)[k] \tag{10.25}$$

Fig.10.5 shows the backprojection of a Q-wavelet coefficient (left) and a U-wavelet coefficient (right). Fig. 10.7 shows the undecimated isotropic polarized wavelet transform of the dust image shown on Fig. 10.6 using six scales, *i.e.* five wavelet scales and the coarse approximation.

10.4 Polarized Curvelet Transform

The 2D ridgelet transform (Candès and Donoho 1999b) was developed in an attempt to overcome some limitations inherent in former multiscale methods e.g. the 2D wavelet, when handling smooth images with edges *i.e.* singularities along smooth curves. Ridgelets are translation invariant ridge functions with a wavelet profile in the normal direction. Although ridgelets provide sparse representations of smooth images with straight edges, they fail to efficiently handle edges along curved lines. This is the framework for curvelets

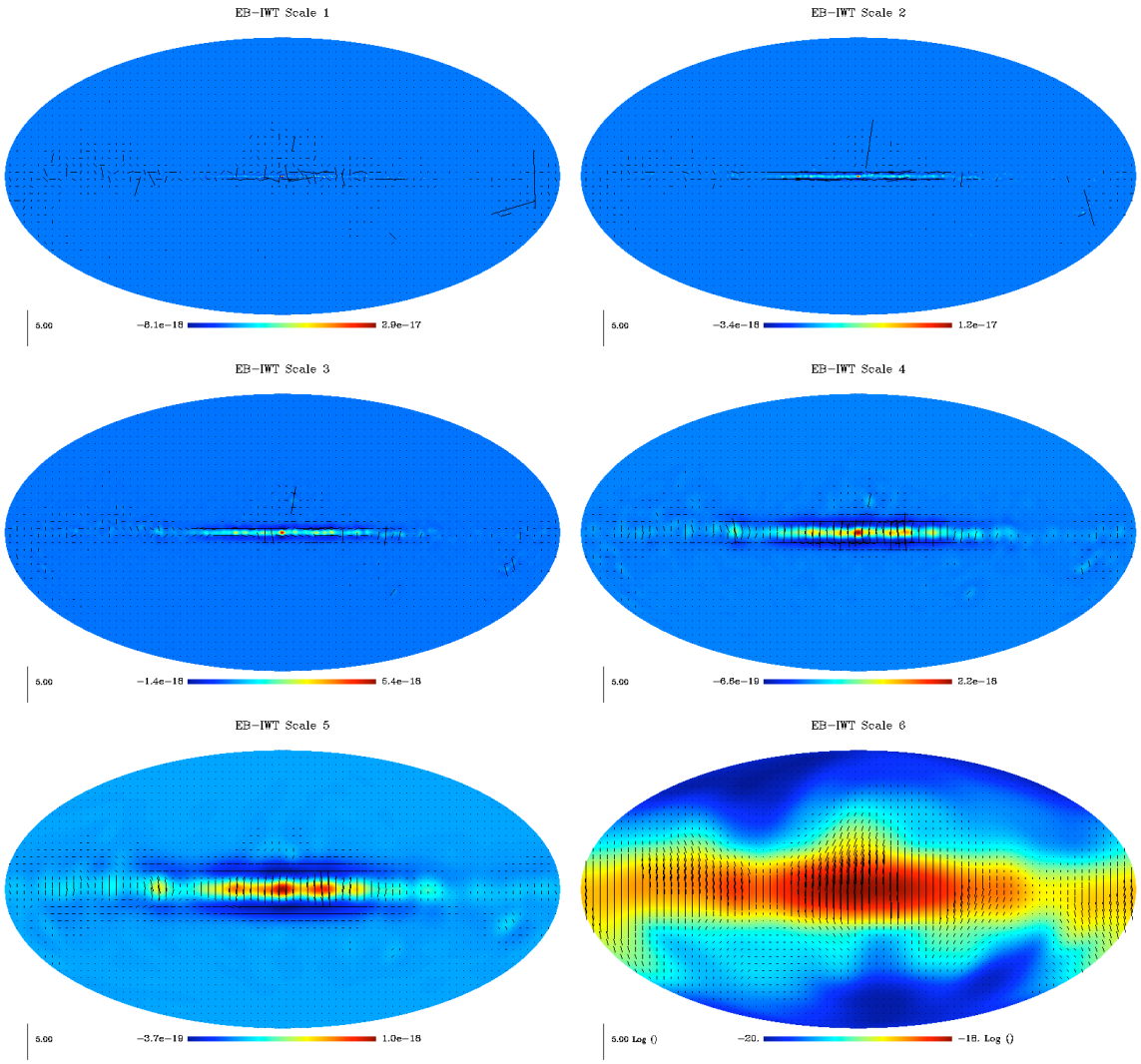


Figure 10.7: QU-Undecimated Wavelet Transform of the simulated polarized map of galactic dust emission shown in figure (10.6).

which were given a first mathematical description in (Candès and Donoho 1999a). Basically, the curvelet dictionary is a multiscale pyramid of localized directional functions with anisotropic support obeying a specific parabolic scaling such that at scale 2^{-j}, its length is $2^{-j/2}$ and its width is 2^{-j}. This is motivated by the parabolic scaling property of smooth curves. Other properties of the curvelet transform as well as decisive optimality results in approximation theory are reported in (Candès and Donoho 1999a). Notably, curvelets provide optimally sparse representations of manifolds which are smooth away from edge singularities along smooth curves. Several digital curvelet transforms (Donoho and Duncan 2000; Starck et al. 2002a; Candès et al. 2006) have been proposed which attempt to preserve the essential properties of the continuous curvelet transform and many papers (Starck et al. 2004b; Herrmann et al. 2008; Starck et al. 2004b) report on their successful application in image processing experiments. The so-called first generation discrete curvelet described in (Donoho and Duncan 2000; Starck et al. 2002a) consists in

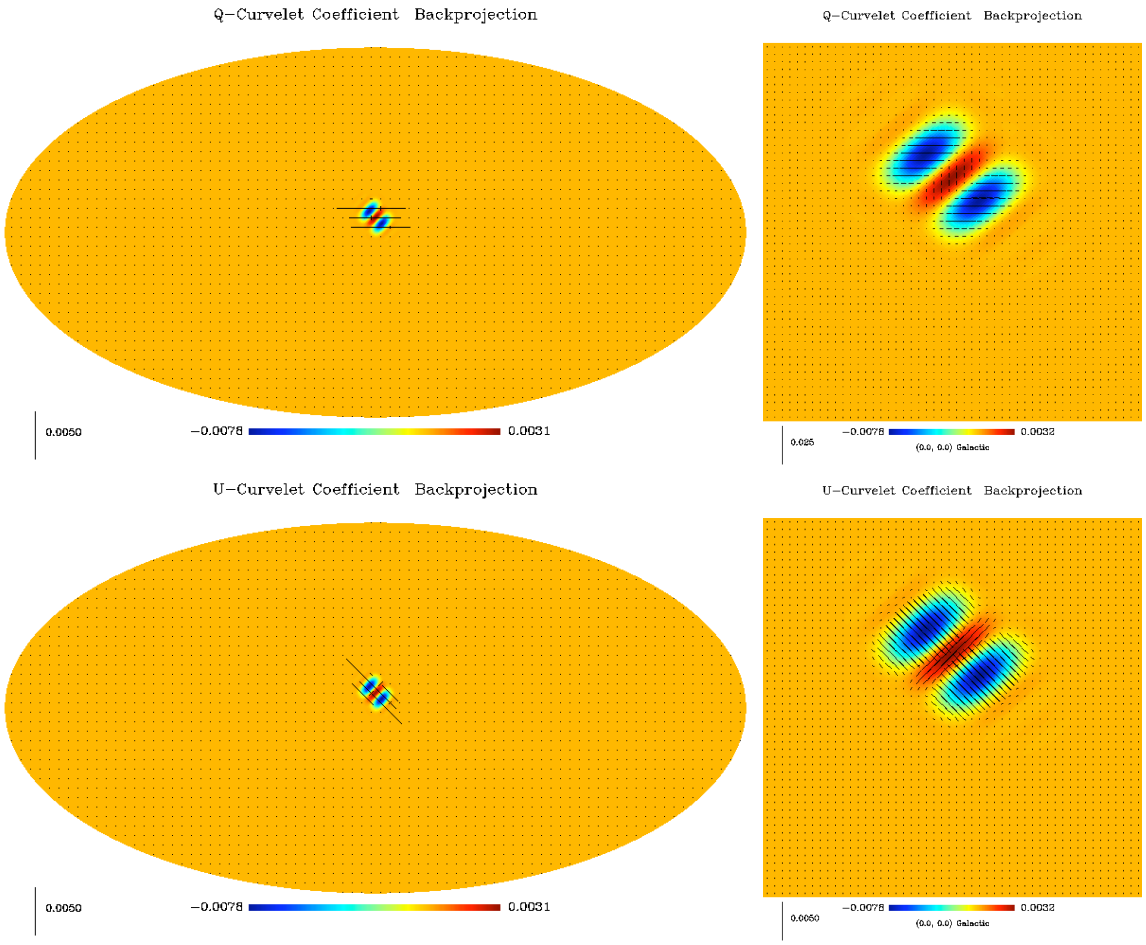


Figure 10.8: Top, Q-curvelet backprojection (left) and zoom (right). Bottom, U-curvelet backprojection (left) and zoom.

applying the ridgelet transform to sub-images of a wavelet decomposition of the original image. By construction, the sub-images are well localized in space and frequency and the subsequent ridgelet transform provides the necessary directional sensitivity. This latter implementation in combination with the good geometric properties of the Healpix pixelization scheme, inspired the digital curvelet transform on the sphere (Starck et al. 2006). The digital curvelet transform on the sphere is clearly invertible in the sense that each step of the overall transform is itself invertible. The curvelet transform on the sphere has a redundancy factor of $16J+1$ when J scales are used, which may be a problem for handling huge data sets such as from the future Planck-Surveyor experiment. This can be reduced by substituting the pyramidal wavelet transform to the undecimated wavelet transform in the above algorithm. More details on the wavelet, ridgelet, curvelet algorithms on the sphere can be found in (Starck et al. 2006). As for the isotropic wavelet on the sphere, a straightforward extension to polarized data will consist in applying successively the curvelet transform on the sphere to the three components T, Q and U. Figure 10.8 shows the backprojection of a Q-curvelet coefficient and U-curvelet coefficient. Clearly, the shapes of these polarized curvelet functions are very different from the polarized wavelet functions.

10.5 Polarized E/B Wavelet and E/B Curvelet

10.5.1 Introduction

We have seen that the generalization of the Fourier representation for polarized data on the sphere is the spin-2 spherical harmonics basis denoted ${}_{\pm 2}Y_{\ell m}$:

$$Q \pm iU = \sum_{\ell,m} {}_{\pm 2}a_{\ell m} {}_{\pm 2}Y_{\ell m} \tag{10.26}$$

At this point, it is convenient (Zaldarriaga 1998) to introduce the two quantities denoted E and B which are defined on the sphere by

$$\begin{aligned} E = &\ \textstyle\sum_{\ell,m} a^E_{\ell m} Y_{\ell m} = \sum_{\ell,m} -\frac{1}{2}({}_2a_{\ell m} + {}_{-2}a_{\ell m}) Y_{\ell m} \\ B = &\ \textstyle\sum_{\ell,m} a^B_{\ell m} Y_{\ell m} = \sum_{\ell,m} i\frac{1}{2}({}_2a_{\ell m} - {}_{-2}a_{\ell m}) Y_{\ell m} \end{aligned} \tag{10.27}$$

where $Y_{\ell m}$ stands for the usual spin 0 spherical harmonics basis functions. The quantities E and B are derived by applying the spin lowering operator twice to $Q + iU$ and the spin raising operator twice to $Q - iU$ so that E and B are real scalar fields on the sphere, invariant through rotations of the local reference frame. The normalization of $a^E_{\ell m}$ and $a^B_{\ell m}$ chosen in the latter definition is purely conventional but it appears to be rather popular (Zaldarriaga and Seljak 1997; Bunn et al. 2003). Still, we could multiply $a^E_{\ell m}$ and $a^B_{\ell m}$ by some A_ℓ and we would have just as good a representation of the initial polarization maps. Through a change of parity E will remain invariant whereas the sign of pseudo-scalar B will change. The E and B modes defined here are not so different from the gradient (i.e. curl free) and curl (i.e. divergence free) components encountered in the analysis of vector fields. Finally, the spatial anisotropies of the Gaussian CMB temperature and polarization fields are completely characterized in this new linear representation by the power spectra and cross spectra of T, E and B. Thanks to the different parities of T and E on one side and B on the other, the sufficient statistics reduce to only four spectra namely $C^{EE}_\ell, C^{TE}_\ell, C^{TT}_\ell, C^{BB}_\ell$. For a given cosmological model, it is possible to give a theoretical prediction of these spectra. Aiming at inverting the model and inferring the cosmological parameters, an important goal of CMB temperature and polarization data analysis is then to estimate the latter power spectra, based on sampled, noisy sometimes incomplete T, Q and U spherical maps.

10.5.2 E/B Isotropic Wavelet

Following the above idea of representing CMB polarization maps by means of E and B modes, we propose a formal extension of the previous undecimated isotropic wavelet transform that will allow us to handle linear polarization data maps T, Q and U on the sphere. Practically, the maps we consider are pixelized using for instance the Healpix pixelization scheme. In fact, we are not concerned at this point with the recovery of E and B modes from pixelized or incomplete data maps which itself is not a trivial task. The extension of the wavelet transform on the sphere we describe here makes use of the E and B representation of polarized maps described above in a formal way. Given polarization data maps T, Q and U, Starck et al. (2009b) proposed a wavelet transform algorithm consisting of the following steps :

1. Apply the spin ± 2 spherical harmonics transform to $Q + iU$ and $Q - iU$. Practically, the Healpix software package provides an implementation of this transform for maps that use this pixelization scheme. Otherwise, a fast implementation was recently proposed by (Wiaux et al. 2007).
2. Combine the decomposition coefficients ${}_2a_{\ell m}$ and ${}_{-2}a_{\ell m}$ from the first step into $a^E_{\ell m}$ and $a^B_{\ell m}$ and build formal E and B maps associated with Q and U by applying the usual inverse spherical harmonics transform, as in equation (10.27). For numerical and algorithmic purposes, it may be efficient to stay with the spherical harmonics representation of E and B.
3. Apply the undecimated isotropic transform on the sphere described above to map T and to the E, B representation of the polarization maps.

The wavelet coefficient maps w^T_j, w^E_j, w^B_j and the low resolution approximation maps c^T_J, c^E_J, c^B_J are obtained by applying the isotropic undecimated wavelet transform described in section 10.3.1 to the T, E, B representation of the polarized data. Figure 10.9 shows the result of applying the proposed transform to the polarized CMB data map Ka [3] from the WMAP experiment. The top two images show the initial Q and U maps while the subsequent maps are the low pass and wavelet coefficients'maps in a four scale decomposition. The scaling function we used is a cubic box spline as proposed in section 10.3.1. The wavelet coefficients were obtained as the difference between two successive low pass approximations of the multiresolution decomposition of the E and B maps. The proper choice for the scaling and wavelet functions will depend on the application and the existence of constraints to be enforced.

Reconstruction

Obviously, the transform described above is invertible and the inverse transform is readily obtained by applying the inverse of each of the three steps in reverse order. If, as in the example decomposition above, we take the wavelet function to be the difference between two successive low pass approximations, the third step is inverted by simply summing the last low pass approximation with the maps of wavelet coefficients from all scales as in equation (10.23) :

$$\begin{aligned} T &= c^T_J + \sum_{j=1}^{J} w^T_j \\ E &= c^E_J + \sum_{j=1}^{J} w^E_j \\ B &= c^B_J + \sum_{j=1}^{J} w^B_j \end{aligned} \tag{10.28}$$

[3] available at http://lambda.gsfc.nasa.gov/product/map/current/

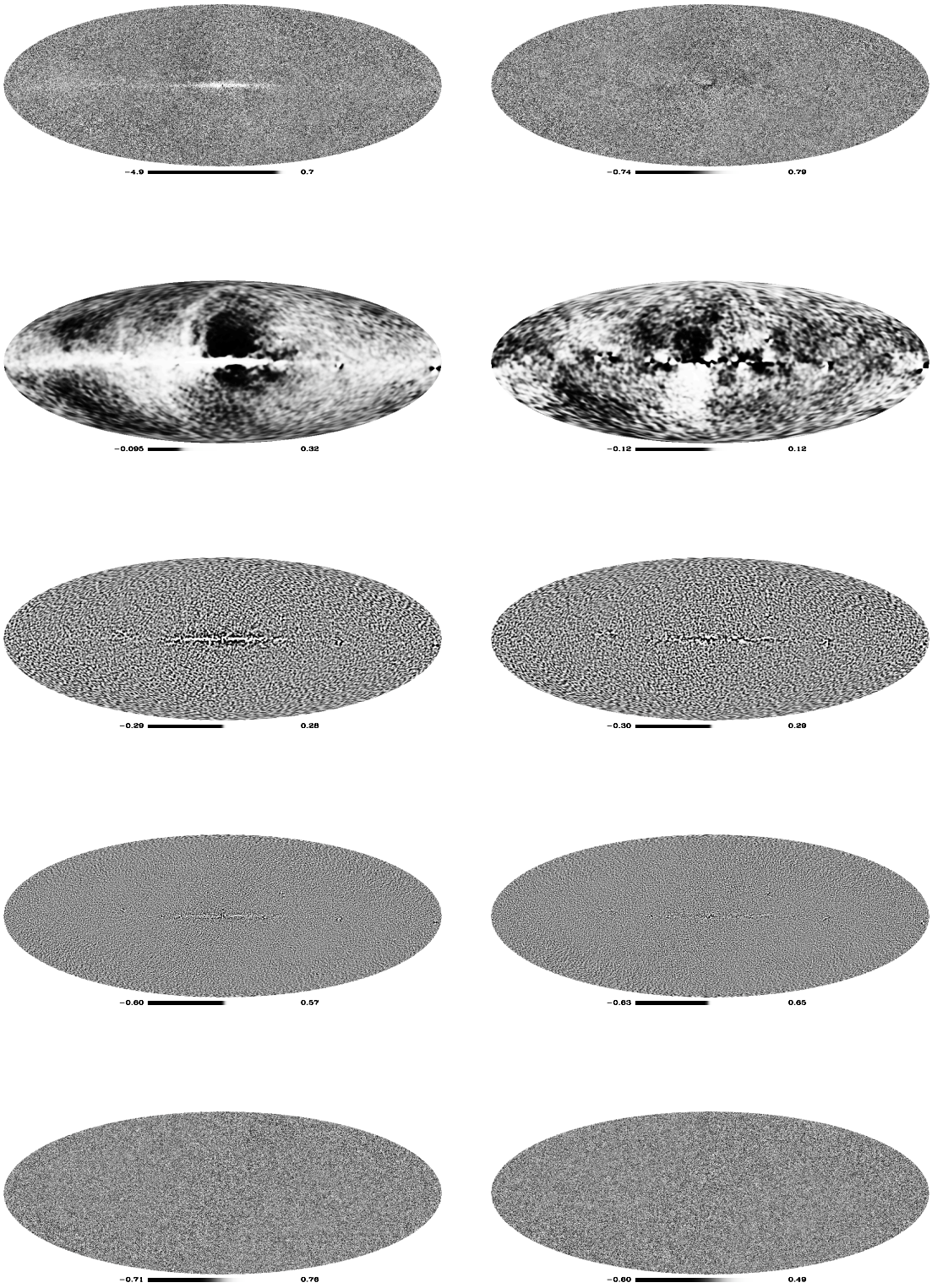


Figure 10.9: **top :** Q and U CMB polarization data maps from channel Ka of the WMAP experiment. **left :** low pass and wavelet coefficients in three scales of the formal E mode. **right :** low pass and wavelet coefficients in three scales of the formal B mode.

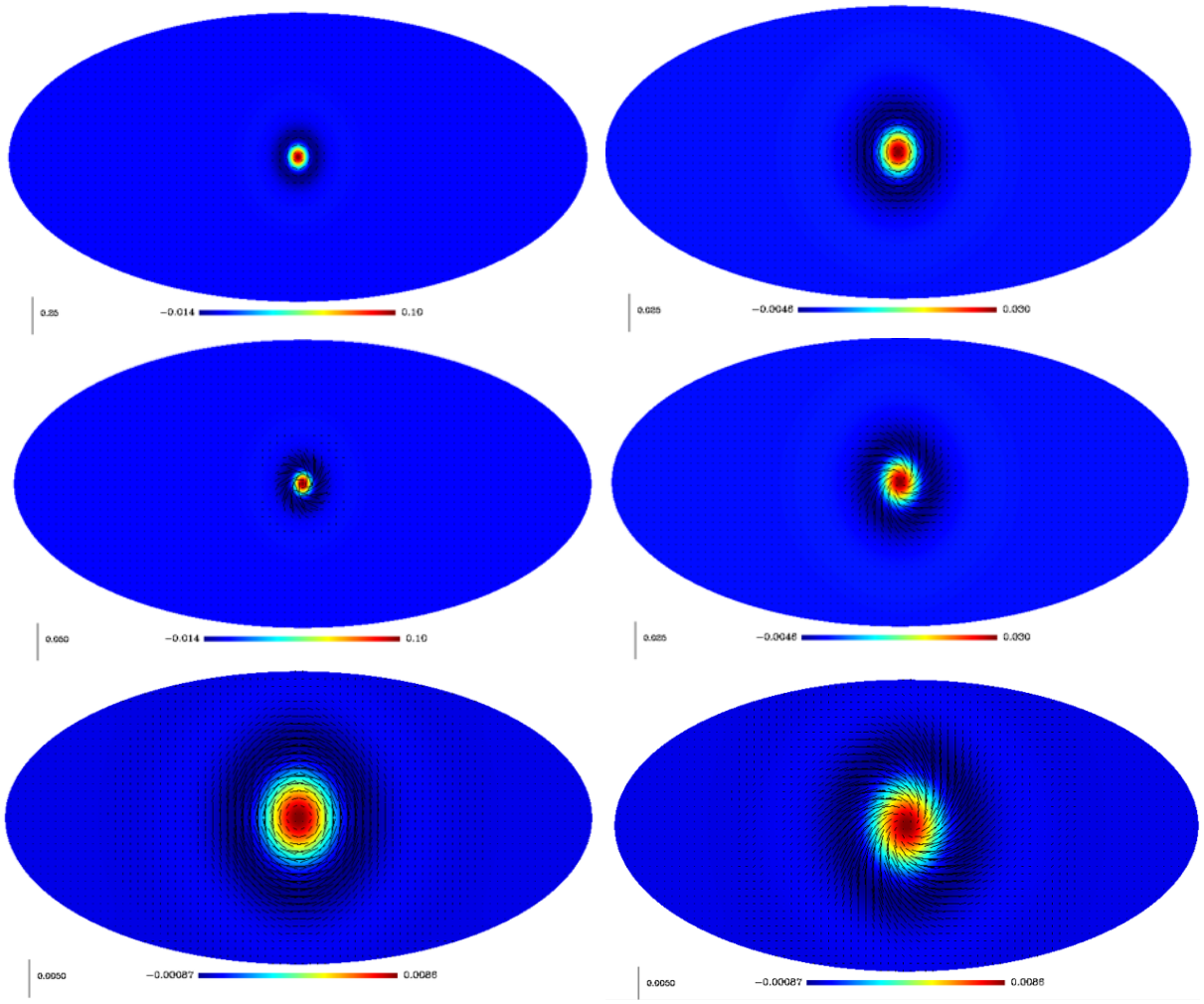


Figure 10.10: E-isotropic wavelet transform backprojection (left) and B-isotropic wavelet backprojection (right).

where c_J^X stands for the low resolution approximation to component X and w_j^X is the map of wavelet coefficients of that component on scale j. Finally, noting that :

$$\begin{aligned} Q &= -\frac{1}{2}\sum_{\ell,m} a^E_{\ell m}({}_2Y_{\ell m} + {}_{-2}Y_{\ell m}) + i a^B_{\ell m}({}_2Y_{\ell m} - {}_{-2}Y_{\ell m}) \\ &= \sum_{\ell,m} a^E_{\ell m} Z^+_{\ell m} + i a^B_{\ell m} Z^-_{\ell m} \qquad (10.29) \\ U &= -\frac{1}{2}\sum_{\ell,m} a^B_{\ell m}({}_2Y_{\ell m} + {}_{-2}Y_{\ell m}) - i a^E_{\ell m}({}_2Y_{\ell m} - {}_{-2}Y_{\ell m}) \\ &= \sum_{\ell,m} a^B_{\ell m} Z^+_{\ell m} - i a^E_{\ell m} Z^-_{\ell m} \qquad (10.30) \end{aligned}$$

the initial representation of the polarized data in terms of T, Q and U maps is reconstructed from its wavelet coefficients using the following equations :

$$\begin{aligned} T = & \quad c_J^T + \sum_{j=1}^J w_j^T \\ Q = & \quad c_J^{E,+} + i c_J^{B,-} + \sum_{j=1}^J \left\{ w_j^{E,+} + i w_j^{B,-} \right\} \\ U = & \quad c_J^{B,+} - i c_J^{E,-} + \sum_{j=1}^J \left\{ w_j^{B,+} - i w_j^{E,-} \right\} \end{aligned} \tag{10.31}$$

where

$$c_J^{X,+} = c_J^X \sum_{\ell,m} Y_{\ell m}^\dagger Z_{\ell m}^+ \quad \text{and} \quad c_J^{X,-} = c_J^X \sum_{\ell,m} Y_{\ell m}^\dagger Z_{\ell m}^- \tag{10.32}$$

with $W^\dagger$ denoting the transpose conjugate of W so that $\tilde{W}W^\dagger$ is the scalar dot product of $\tilde{W}$ and W while $W^\dagger\tilde{W}$ is an operator (or matrix) acting on its left hand side as a projection along W and reconstruction along $\tilde{W}$. In practice, the Healpix software package provides us with an implementation of the forward and inverse spin 0 and spin 2 spherical harmonics transforms which we need to implement the proposed inverse transform given by equations (10.31) and (10.32). Clearly, as mentioned earlier in section 10.3.1, we could have chosen some other wavelet function than merely the difference between two consecutive scaling functions, and the transformation would still be nearly as simple to invert. Figure 10.10 shows, on the left, backprojections of E-wavelet coefficients, and, on the right, backprojections of B-wavelet coefficients on the right hand side at different scales.

E-B Curvelet

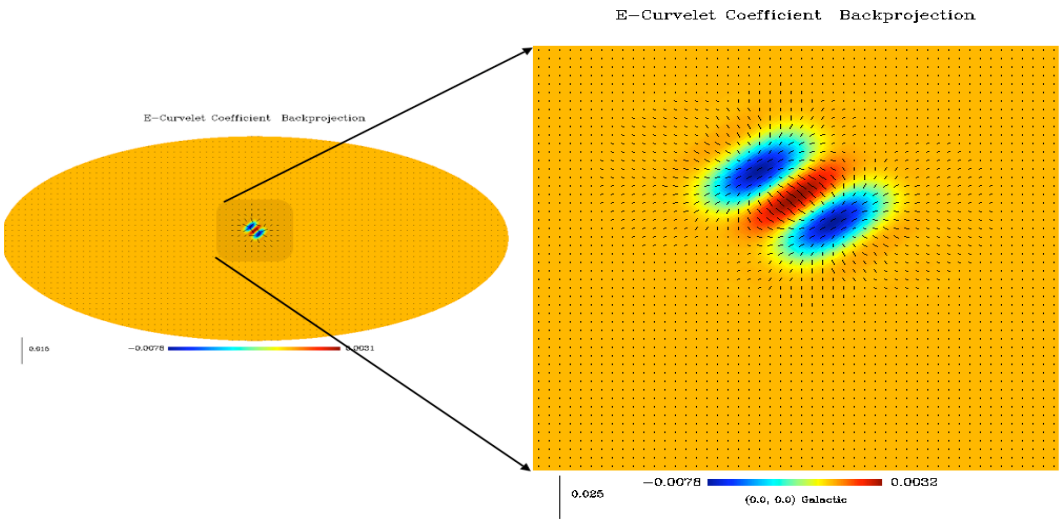


Figure 10.11: E-curvelet coefficient backprojection.

Similarly to the EB-wavelet constructions, we can easily construct an EB-curvelet transform by first computing the E and B components using the spin ± 2 spherical harmonics transform, and then applying a curvelet transform on the sphere separately on

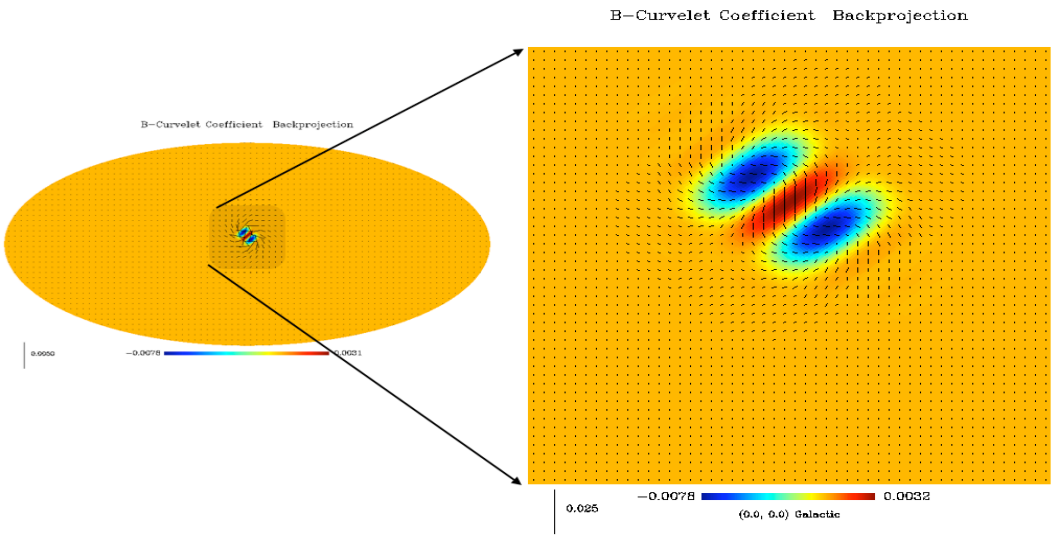


Figure 10.12: B-curvelet coefficient backprojection.

each of these two components Starck et al. (2009b). Figure 10.11 shows the backprojection of an E-curvelet coefficient and Figure 10.12 shows the backprojectionof a B-curvelet coefficient.

10.6 Experiments : Application to denoising

Thanks to the invertibility of the different proposed transforms for polarization maps on the sphere, it is possible to use these transformations for restoration applications. The denoising algorithm we use here consists in three consecutive steps :

1. Apply the chosen polarized multiscale transform.
2. Set to zero those coefficients whose absolute values are below a given threshold. We have used a threshold equal to five times the noise standard deviation.
3. Reconstruct the denoised field using the inverse transform.

More sophisticated thresholding strategies exist (Starck and Murtagh 2006) which can be used just as well on coefficients of polarized wavelets and curvelets.

Figure 10.13 illustrates the results of a simple denoising experiment. Noise was added to a simulated dust image (see Figure 10.13 top left and right), and the noisy QU-field was filtered by thresholding either the EB-isotropic wavelet coefficients of the polarized dust map (Figure 10.13 bottom left) or the EB-isotropic curvelet coefficients (Figure 10.13 bottom right). Both decompositions produce nice visual results.

In order to be more quantitative, we used two test images (synchrotron and dust) with different noise levels. The noise levels were taken equal to $0.1, 0.2, 0.5, 1.$ and 2 times the

Table 10.1: Error (in dB) for both the Q and U components between the original dust image and respectively its noisy version and the results obtained by hard thresholding representations of the noisy data in different dictionaries.

Q Component					
Noisy image	20.01	13.98	6.03	-0.007	-6.02
QU-UWT	34.91	32.41	28.88	26.41	23.68
EB-UWT	33.52	30.63	27.83	25.86	23.91
QU-CUR	34.14	31.12	28.56	26.58	24.56
EB-CUR	33.36	30.60	28.17	26.30	23.99
OWT	34.07	30.77	27.08	24.57	21.33
Mod-Phase	30.51	26.42	20.97	16.25	11.05
U Component					
Noisy image	19.99	13.97	6.01	-0.001	-6.01
QU-UWT	40.88	39.22	36.77	35.50	31.90
EB-UWT	38.80	36.71	35.41	34.67	32.23
QU-CUR	39.54	37.85	36.68	35.83	32.38
EB-CUR	39.64	37.72	35.33	33.51	29.30
OWT	39.18	37.29	33.30	29.06	23.56
Mod-Phase	30.76	26.81	21.31	16.83	11.36

Table 10.2: Error (in dB) for both the Q and U components between the original synchrotron image and respectively its noisy version and the results obtained by the hard thresholding using different dictionaries.

Q Component					
Noisy image	19.99	13.98	6.02	-0.005	-6.02
QU-UWT	33.40	31.27	27.78	24.71	21.16
EB-UWT	33.03	30.53	26.79	23.86	20.76
QU-CUR	35.04	32.92	29.19	26.19	22.37
EB-CUR	34.83	32.09	27.89	24.81	21.06
OWT	33.20	30.10	25.72	22.62	19.39
MP-UWT	26.76	23.05	17.85	13.69	8.97
U Component					
Noisy image	19.99	13.97	6.01	-0.007	-6.01
QU-UWT	32.90	31.25	29.04	27.51	25.27
EB-UWT	33.22	31.43	29.11	26.99	24.35
QU-CUR	33.79	31.75	29.28	27.71	25.46
EB-CUR	34.05	31.98	29.38	27.17	24.07
OWT	32.69	30.49	27.87	25.49	21.87
MP-UWT	26.76	22.76	17.85	13.86	9.41

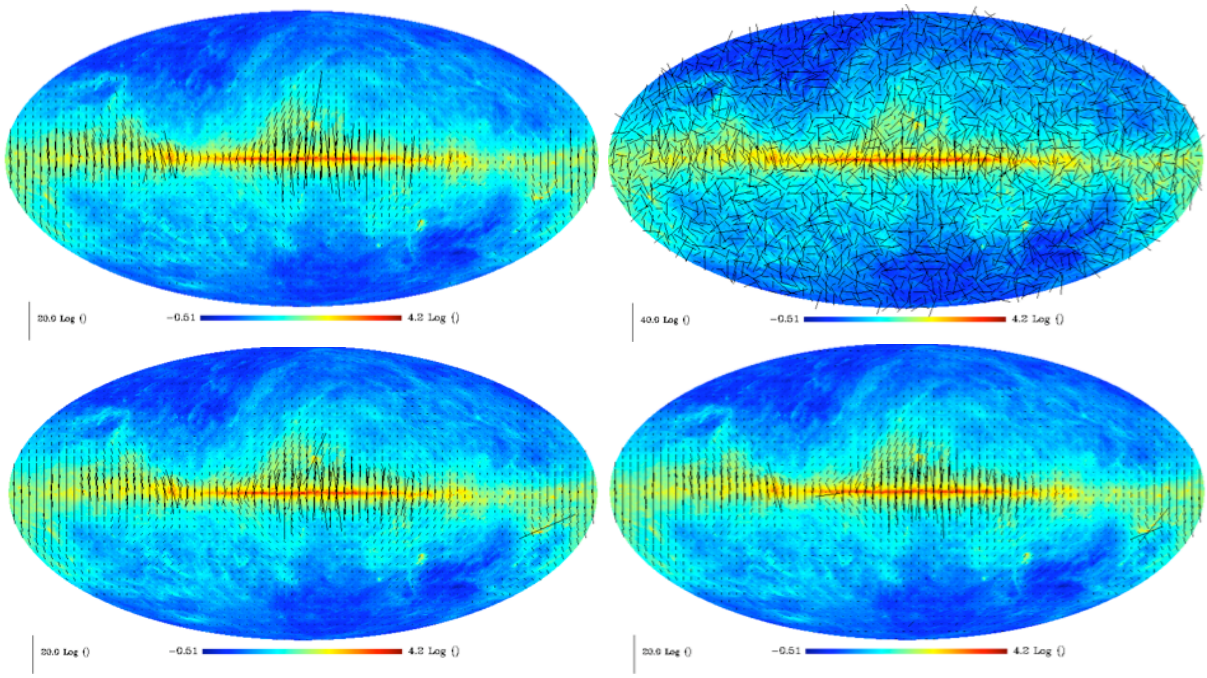


Figure 10.13: Top, simulated input polarized image (left) and noisy polarized field (right) Bottom, denoising of the polarized field using the EB-isotropic undecimated wavelets (left) and the EB-curvelets (right).

standard deviation of the noise-free image. On each noisy image, we applied six different transformations, thresholded the coefficients and reconstructed the filtered images. The transforms we used are the QU-wavelets (QU-UWT), the EB-wavelets (EB-UWT), the QU-curvelet (QU-CUR), the EB-curvelet (EB-CUR), the biorthogonal wavelet transform (OWT) and the modulus-phase undecimated multiscale transform (MP-UWT). For each filtered image Q (resp. U), we computed the Signal-to-Noise Ratio (SNR) in dB :

$$E = 10 \log_{10}(\sigma_I^2/\sigma_e^2) \tag{10.33}$$

where σ_I is the standard deviation of the noise free original image component Q (resp. U) and σ_e is the standard deviation of the error image *i.e.* the difference between the noise-free component Q (resp. U) and the filtered component Q (resp. U). These errors are given in Table (10.1) for the dust image and in Table (10.2) for the synchrotron image.

It is clear from the above results that the different transforms described here do not perform as well on this specific numerical experiment. For the dust, QU-wavelets are better when the noise level is not so high, while curvelets do better when the noise and signal levels are of the same order. This can be explained by the fact that structures on small scales are more or less isotropic, and therefore better represented by wavelets, while large scale structures are anisotropic and therefore better analyzed using curvelets. When increasing the noise level, structures on the smallest scales are no longer detected by either of the two dictionaries. Only large scale features are detectable, and curvelets do this job better. Dealing with the polarized synchrotron map, curvelets do better than wavelets in all cases experimented with here. Although the bi-orthogonal wavelet transform is clearly

not as good as the others in these experiments, it could nevertheless be very useful in situations where computation time is an important issue. Indeed, since it doesn't make use of the spherical harmonics transform and also because it is not redundant, it is a very fast transform. Finally the worse results were obtained using the Modulus-Phase multiscale transform. This can be explained by the fact that the Gaussian approximation we made for the noise in the wavelet transform of the modulus is not accurate enough. Furthermore, it is not clear what is the best way to correct the phase wavelet coefficients from the noise. A better understanding of the noise behavior after transformation is clearly required before the Mod-Phase multiscale transform can be used for restoration purposes. However, for other applications such as non-Gaussianity studies, the latter transform could prove an interesting tool to use as well.

Chapter 11

IDL Routines

11.1 Introduction

A set of routines has been developed in IDL. Starting IDL using the script program *isap.pro* allows the user to get the interactive sparse data analysis environment, and all routines described in the following can be called. An online help facility is also available by invoking the *isaph* program under IDL. See Chapter 1 for more details on the software installation.

11.2 Functions for polarized spherical maps

11.2.1 Reading a polarized spherical map from a file : mrsp_read

Read a polarized spherical map in Healpix format.

USAGE: map = mrsp_read(file, noverb=noverb)

where

- *file* : Input string, name of the file to be read. The pathname can be included in the string, by default 'file.fits' is equivalemt to './file.fits'
- *noverb* : scalar, prevent the printing on the screen of the format (RING or NESTED) of the read map and the number of pixels.
- *map* : Output 3D IDL array of Healpix map read. The map is setted to the NESTED format after reading.

Examples:

- map = mrsp_read('my_file_healpix_pola.fits', noverb=noverb)
  Read the map stored into the file 'my_file_healpix_pola.fits' and load it into map.

11.2.2 Writing a polarized spherical map into a file : mrsp_write

Write a polarized spherical map in Healpix format.

USAGE: mrsp_write, file, map, ring=ring

where

- *file* : Input string, name of the file to be writen. The pathname can be included in the string, by default 'file.fits' is equivalemt to './file.fits'
- *map* : Input 3D IDL array of Healpix map to be writen. The map is assumed to be in the NESTED format.
- *ring* : scalar, if set convert the Healpix map data to the RING format for the writing.

11.2.3 Conversion of a polarized spherical map from TQU scheme to TEB scheme : mrsp_tqu2teb

Convert a polarized map in Healpix nested format from TQU scheme to TEB scheme.

USAGE: mrsp_tqu2teb, map_tqu, map_teb

where

- *map_tqu* : Input 3D IDL array of healpix polarized map in TQU scheme.
- *map_teb* : Output 3D IDL array of healpix polarized map in TEB scheme.

11.2.4 Conversion of a polarized spherical map from TEB scheme to TQU scheme : mrsp_teb2tqu

Convert a polarized map in Healpix nested format from TEB scheme to TQU scheme.

USAGE: mrsp_teb2tqu, map_teb, map_tqu

where

- *map_teb* : Input 3D IDL array of healpix polarized map in TEB scheme.
- *map_tqu* : Output 3D IDL array of healpix polarized map in TQU scheme.

11.2.5 Resizing a polarized spherical map: mrsp_resize

Resize a polarized map in Healpix nested format.

USAGE: resize_map = mrsp_resize(map, nside=nside, ViaAlm=ViaAlm, teb=teb)

where

- *map* : Input 3D IDL array of healpix polarized map in TQU scheme to be transformed.

- *resize_map* : Output 3D IDL array of healpix polarized map in TQU scheme. Healpix input map and output resized map are in nested format.
- *nside* : int, the new nside parameter of the output healpix resized map.
- *ViaAlm* : scalar, if set use alm transform for the resizing, otherwise, use interpolation. Ignored if nside keyword value is lower than imag nside.
- *teb* : scalar, if set specifies that the input and output images are in TEB scheme.

Examples:

- map2 = mrsp_resize(map, nside = 256, /ViaAlm)
  resize an Healpix map.

11.3 General transform/reconstruction routines

11.3.1 Transformations of a polarized spherical map : mrsp_trans

Compute a transform (E-B mode decomposition, wavelet with spline, Meyer or needelet filters, curvelet, ...) on a polarized map on the sphere in the Healpix representation (nested data representation) in TQU scheme.

The transform can be:

1. A E/B decomposition using the spin_2 transform.
2. An orthogonal wavelet on each T,Q,U component.
3. A pyramidal isotropic wavelet on each T,Q,U component.
4. An undecimated wavelet transform on each component.
5. A decimated modulus-Phase wavelet transform.
6. A undecimated modulus-Phase wavelet transform.
7. A curvelet transform.

USAGE: mrsp_trans, Imag, Trans, NbrScale=NbrScale, lmax=lmax, MeyerWave=MeyerWave, ebdec=ebdec, Cur=Cur, uwt=uwt, owt=owt, mpdwt=mpdwt, mpuwt=mpuwt, PyrWT=PyrWT, DifInSH=DifInSH, Overlap=Overlap, FirstBlockSize=FirstBlockSize

where

- *Imag* : Input 3D IDL array of healpix polarized map in TQU scheme. Input image to be transformed.
- *Trans* : Output IDL structure with the following fields:

  - *NbrScale* : int, number of scales.
  - *nside* : int, Healpix nside parameter.
  - *lmax* : int, maximum l value in the Spherical Harmonic Space.
  - *npix* : long, number of pixels of the input image.
  - *MeyerWave* : int, 1 if the keyword MeyerWave used, otherwise 0.
  - *DifInSH* : int, 1 if the keyword DifInSH used, otherwise 0.
  - *pyrtrans* : int, 1 if a pyramidal decomposition has been applied, otherwise 0.
  - *ebdec* : int, 1 if an EB decomposiiton has been applied, otherwise 0.
  - *DEC1* : IDL structure, first component transformation (depends on the chosen transform).
  - *DEC2* : IDL structure, second component transformation (depends on the chosen transform).
  - *DEC3* : IDL structure, third component transformation (depends on the chosen transform).
  - *TransChoice* : string, code of the chosen transform.
  - *TabCodeTransform* : string array, array of transforms codes. TabCodeTransform = ['T_EBDEC', 'T_OWT', 'T_PyrWT', 'T_UWT', 'T_MPDWT', 'T_MPUWT', 'T_CUR']
  - *TransName* : string, transform's name.
  - *TransTypeName* : string array, array of transforms names. TransTypeName = ['EBDEC','Bi-Orthogonal WT', 'Pyramidal WT', 'Undecimated WT', 'Module-Phase Decimated Transform', 'Module-Phase Undecimated Transform', 'Curvelet']

- *NbrScale* : int, number of scales of the wavelet transforms.
- *ebdec* : scalar, if set an E/B decomposition is applied before the chosen multiscale decomposition. If no transform is selected, it will be the default transformation.
- *Cur* : scalar, if set perform a curvelet transform.
- *uwt* : scalar, if set perform an undecimated isotropic wavelet transform.
- *PyrWT* : scalar, if set perform a pyramidal isotropic wavelet transform.
- *owt* : scalar, if set perform a bi-orthogonal wavelet transform on each face.
- *mpdwt* : scalar, if set perform a decimated module-phase wavelet transform.
- *mpuwt* : scalar, if set perform a undecimated module-phase wavelet transform.
- *Overlap* : int, if equal to 1 if blocks are overlapping, only used with curvelet transform.
- *FirstBlockSize* : int, block size in the ridgelet transform at the finest scale (default is 16), only used with curvelet transform.
- *lmax* : int, maximum l value in the Spherical Harmonic Space (for isoptropic wavelet transform only).

- *DifInSH* : Input keyword parameter. If set, the wavelet coefficients are computed as the difference between two resolutions in the spherical harmonics representation. Otherwise, the wavelet coefficients are computed as the difference between two resolutions in direct space. Only used with keyword uwt or PyrWT.
- *MeyerWave* : If set, use Meyer wavelets and set the keyword DifInSH. Only used with keyword uwt or PyrWT.

Examples:

- mrsp_trans, Imag, WT, NbrScale=5, /uwt
  Compute the undecimated wavelet transform of the map Imag with five scales. The result is stored in WT.

11.3.2 Reconstructions of a polarized spherical map : mrsp_rec

Compute a inverse transform (wavelet, curvelet, ...) to get a polarized map on the sphere in the Healpix representation (nested data representation) in TQU scheme from its decomposition obtained by mrsp_trans.

The transform can be:

1. A E/B decomposition using the spin_2 transform.
2. An orthogonal wavelet on each T,Q,U component.
3. A pyramidal isotropic wavelet on each T,Q,U component.
4. An undecimated wavelet transform on each component.
5. A decimated modulus-Phase wavelet transform.
6. A undecimated modulus-Phase wavelet transform.
7. A curvelet transform.

USAGE: mrsp_rec, Trans, Rec

where

- *Trans* : Input IDL structure, see mrsp_trans for more details.
- *Rec* : Output 3D IDL array of healpix polarized map in TQU scheme. Reconstructed image.

Examples:

- mrsp_trans, Imag, WT, NbrScale=5, /uwt
  Compute the undecimated wavelet transform of the map Imag with five scales. The result is stored in WT.
- mrsp_rec, WT, RecIma
  Reconstruct the image.

11.4 Spin-2 spherical harmonic transform

11.4.1 ALM transform of a polarized spherical map : mrsp_almtrans

Computes the spherical harmonic transform of a polarized TQU map using the Healpix representation (nested data).

USAGE: mrsp_almtrans, Imag, Trans, lmax=lmax, tab=tab, complex=complex, norm=norm, fast=fast

where

- *Imag* : Input 3D IDL array of healpix polarized map in TQU scheme to be transformed.
- *Trans* : Output IDL structure with the following fields:
  - *ALM* : array of the ALM coefficients

    ALM = fltarray[*,2,3] list of the real part (ALM[*,0,*]) and imaginary part (ALM[*,1,*]) of the ALM. This is the default storage, ALM[*,*,0] is ALM T, ALM[*,*,1] is ALM E and ALM[*,*,2] is ALM B
    ALM = cfarr[*,3] list of the ALM in complex values format if the keyword complex is set. ALM[*,0] is ALM T, ALM[*,1] is ALM E and ALM[*,2] is ALM B
    ALM = fltarray[NbrMaxM, NbrMaxL, 2, 3] table of the real part (ALM[*,*,0,*]) and imaginary part (ALM[*,*,1,*]) of the ALM if the keyword tab is set. ALM[*,*,*,0] is ALM T, ALM[*,*,*,1] is ALM E and ALM[*,*,*,2] is ALM B
    ALM = cfarr[NbrMaxM, NbrMaxL, 3] table of the ALM in complex values format if the keywords complex and tab are both setted. ALM[*,*,0] is ALM T, ALM[*,*,1] is ALM E and ALM[*,*,2] is ALM B
    By default, NbrMaxM = NbrMaxL = lmax+1
  - *complex_alm* : int, 0 (default value) if ALM array contains real and imaginary part separated. 1 if ALM is a complex array.
  - *PixelType* : int, 0 for a Healpix input map (1 for GLESP but not used).
  - *tab* : int, 0 for default ALM representation as a list (i.e. 1D IDL array) and 1 for 2D representation as a table (i.e. l for the first dimension and m for the second).
  - *nside* : int, Healpix nside parameter.
  - *lmax* : int, maximum l value in the Spherical Harmonic Space.
  - *npix* : long, number of pixels of the input image.
  - *TabNbrM* : int array[NbrMaxL], max number of m value for a given l, only used if keyword tab is set otherwise, 0.
  - *index* : long array, indicies of the ALM coefficients, used only if keyword tab is not set.
  - *NormVal* : float, normalization value applied to the alm coefficients (only if keyword norm used).

  - *norm* : int, 0 if no normalization has been aplied, else 1.

- *lmax* : int, Number of spherical harmonics computed in the decomposition. For a Healpix map, default is 3*nside and should be between 2*nside and 4*nside.

- *tab* : scalar, if set, ALM coefficients in Trans.alm are stored in a 2D array: Trans.alm[m,l] where m = 0..Trans.TabNbrM[l]-1 and l = 0..lmax-1

- *complex* : scalar, if set Trans.alm will contain complex values instead of the real and imaginary parts.

- *norm* : scalar, if set, a normalization is performed to the alm coefficients.

Example:

- mrsp_almtrans, Imag, Output
  Compute the spherical harmonics transform of a polarized image, the result is stored in Output.

11.4.2 ALM inverse transform of a polarized spherical map : mrsp_almrec

Computes the inverse spherical harmonic transform of a polarized TQU map using using the Healpix representation (nested data).

USAGE: mrsp_almrec, Trans, imag, pixel_window=pixel_window

where

- *Trans* : Input IDL structure of ALM coefficients, see mrsp_almtrans above for details.

- *Imag* : Output 3D IDL array of healpix polarized map in TQU scheme. Image reconstructed in Healpix nested representation.

- *pixel_window* : scalar, if set the image is convolved by the healpix pixel window (only for Healpix map).

Example:

- mrsp_almtrans, PolaImag, Output
  Compute the spherical harmonics transform of a polarized image, the result is stored in Output.

- mrsp_almrec, Output, PolaRec
  Reconstruct the image.

11.4.3 Power spectrum and cross spectrum exctraction from polarized ALM : mrsp_alm2spec

Computes the power spectrums and cross spectrums of a polarized map from the polarized ALM coefficients.

USAGE: spec = mrsp_alm2spec(ALM, StdPS=StdPS)

where

- *ALM* : Input IDL structure of ALM polarized coefficients, see mrsp_almtrans above for details.
- *spec* : Output 2D IDL float array[ALM.lmax+1,6], the TT, EE, BB, TE, TB, EB spectrums. P[k,i] = Mean(SPECTRUM[*,l,i]) i=0...5.
- *StdPS* : Output 2D IDL float array[ALM.lmax+1,6]: estimated standard deviation of the spectrums coefficients.

Example:

- mrsp_almtrans, Imag, Output
  Compute the spherical harmonics transform of a polarized image, the result is stored in Output.
- spec = mrsp_alm2spec(Output, StdPS=StdPS)
  Compute the spectrums of the image and it's associated standard deviation.

11.4.4 Power spectrum and cross spectrum exctraction from a polarized image : mrsp_spec

Computes the power spectrums and cross spectrums of a polarized map, using the HEALPix representation (nested data representation by default). By default a normalisation is applied on the ALM coefficents.

USAGE: spec = mrsp_spec(Imag, nonorm=nonorm, teb=teb, NormVal=NormVal, StdPS=StdPS, lmax=lmax)

where

- *Imag* : Input 3D IDL array of healpix polarized map in TQU scheme. Input image whose power spectrum will be extracted.
- *spec* : Output 2D IDL float array[ALM.lmax+1,6], the TT, EE, BB, TE, TB, EB spectrums. P[k,i] = Mean(SPECTRUM[*,l,i]) i=0...5.
- *Lmax* : int, number of spherical harmonics computed in the decomposition and size of the computed spectrum (Lmax+1). Default is 3*nside and should be between 2*nside and 4*nside.
- *nonorm* : scalar, if set no normalisation is applied on the ALM computed.

- *StdPS* : Output 2D IDL float array[ALM.lmax+1,6]: estimated standard deviation of the spectrums coefficients.
- *NormVal* : float, normalization value applied to the alm coefficients.
- *teb* : scalar, if set specifies that the input map is in TEB scheme.

Example:

- P = mrsp_spec(Imag)
  Compute the spectrum of the polarized image.

11.5 Polarized Wavelets

11.5.1 Undecimated Isotropic Wavelet Transform of a polarized spherical map : mrsp_wttrans

Computes the undecimated isotropic wavelet transform of polarized maps on the sphere in TQU scheme, using the Healpix representation (nested data representation). The wavelet function is zonal and its spherical harmonics coefficients $a_{l,0}$ follow a cubic box-spline profile. If the keyword DifInSH is set, the wavelet coefficients are derived in the Spherical Harmonic Space, otherwise (default) they are derived in the direct space.

USAGE: mrsp_wttrans, Imag, Trans, NbrScale=NbrScale, lmax=lmax, DifInSH=DifInSH, MeyerWave=MeyerWave

where

- *Imag* : Input 3D IDL array of healpix polarized map in TQU scheme. Input image to be transformed.
- *Trans* : Output IDL structure with the following fields:
  - *NbrScale* : int, number of scales.
  - *nside* : int, Healpix nside parameter.
  - *lmax* : int, maximum l value in the Spherical Harmonic Space.
  - *npix* : long, number of pixels of the input image.
  - *MeyerWave* : int, 1 if the keyword MeyerWave used, otherwise 0
  - *DifInSH* : int, 1 if the keyword DifInSH used, otherwise 0
  - *Coef* : fltarr[npix,NbrScale,3] wavelet transform of the data. Coef[*,*,0] = wavelet transform on T, Coef[*,*,1] = wavelet transform on E, Coef[*,*,2] = wavelet transform on B

    Coef[*,0,*] = wavelet coefficients of the finest scale (highest frequencies).
    Coef[*,NbrScale-1,*] = coarsest scale (lowest frequencies).
- *NbrScale* : int, optional input parameter specifying the number of scales (default is 4).

- *Lmax* : int, optional input parameter specifying the maximum multipole number l in the spherical harmonics decomposition (default is $3 \times$ nside, should be between $2 \times$ nside and $4 \times$ nside).
- *DifInSH* : Input keyword parameter. If set, the wavelet coefficients are computed as the difference between two resolutions in the spherical harmonics representation. Otherwise, the wavelet coefficients are computed as the difference between two resolutions in direct space.
- *MeyerWave* : If set, use Meyer wavelets and set the keyword DifInSH.

Example:

- mrsp_wttrans, Imag, Output, NbrScale=5
  Compute the isotropic wavelet transform of the map Imag with five scales. The result is stored in Output.

11.5.2 Undecimated Isotropic Wavelet Reconstruction of a polarized spherical map : mrsp_wtrec

Reconstructs a polarized maps on the sphere in TQU scheme using the Healpix representation (nested data representation) from its wavelet coefficients obtained with the undecimated isotropic wavelet transform on the sphere, described right above.

USAGE: mrsp_wtrec, Trans, Rec, filter=filter

where

- *Trans*: Input IDL structures with the following fields:
  - *NbrScale* : int, number of scales.
  - *nside* : int, Healpix nside parameter.
  - *lmax* : int, maximum l value in the Spherical Harmonic Space.
  - *npix* : long, number of pixels of the input image.
  - *MeyerWave* : int, 1 if the keyword MeyerWave used, otherwise 0
  - *DifInSH* : int, 1 if the keyword DifInSH used, otherwise 0
  - *Coef* : fltarr[npix,NbrScale,3] wavelet transform of the data. Coef[*,*,0] = wavelet transform on T, Coef[*,*,1] = wavelet transform on E, Coef[*,*,2] = wavelet transform on B

    Coef[*,0,*] = wavelet coefficients of the finest scale (highest frequencies).
    Coef[*,NbrScale-1,*] = coarsest scale (lowest frequencies).
- *Rec* : Output 3D IDL array of healpix polarized map in TQU scheme. Reconstructed image from the wavelet coefficients.
- *filter* : Input keyword parameter. Use filters for the reconstructions. If this keyword is not set, the reconstructed image is obtained by a simple addition of all wavelet scales. Automaticaly applied if keyword MeyerWave or DifInSH were set at the wavelet decomposition.

Examples:

- mrsp_wttrans, Imag, Output, NbrScale=5
  Compute the isotropic wavelet transform of the map Imag with five scales. The result is stored in Output.
- mrsp_wtrec, Output, map
  Reconstruct the map.

11.5.3 Extract a scale from a polar decomposition : mrsp_wtget

Return a band of a transform for Healpix polarized map (wavelet, curvelet...) obtained by the command mrsp_trans.

USAGE: Scale = mrsp_wtget(Trans, Component, ScaleNumber, BandNumber=BandNumber, NormVal=NormVal)

where

- *Trans* : Input IDL structure, see mrsp_trans for more details.
- *ScaleNumber* : int, scale number of the band to be extracted. The scale number must be between 0 and Trans.NbrScale-1.
- *Component* : int, choice of the component, 0 is for T, 1 for E and 2 for B.
- *NormVal* : float, optional normalization value of the band (for isotropic wavelet transform).
- *BandNumber* : int, ridgelet band number (for curvelet transform).
- *Scale* : return value IDL array of the band extracted. See more details on the 1D versions of the functions. No band is extracted from Modulus-Phases transforms (return 0). For a E/B decomposition, it will be either the T map, the E map or the B map.

11.5.4 Put a scale into a polar decomposition : mrsp_wtput

Put a band into a transform for Healpix polarized map (wavelet, curvelet...) obtained by the command mrsp_trans.

USAGE: mrsp_wtput, Trans, Scale, Component, ScaleNumber, BandNumber=BandNumber

where

- *Trans* : Input IDL structure, see mrsp_trans for more details.
- *Scale* : Input IDL array of the band inserted. See more details on the 1D versions of the functions. No band is inserted into Modulus-Phases transforms. For a E/B decomposition, it will be either the T map, the E map or the B map.

- *ScaleNumber* : int, scale number of the band to be extracted. The scale number must be between 0 and Trans.NbrScale-1.
- *Component* : int, choice of the component, 0 is for T, 1 for E and 2 for B.
- *BandNumber* : int, ridgelet band number (for curvelet transform).

11.6 Denoising

11.6.1 Wavelet filtering of a polarized spherical map : mrsp_wtfilter

Wavelet denoising of a polarized image on the sphere using Healpix representation in TQU scheme (nested pixel representation). By default Gaussian noise is considered. If the keyword SigmaNoise is not set, then the noise standard deviation is automatically estimated. If the keyword MAD is set, then a correlated Gaussian noise is considered and the noise level at each scale is derived from the Median Absolution Deviation (MAD) method. If the keyword KillLastScale is set, the coarsest resolution is set to zero. The thresholded wavelet coefficients can be obtained using the keyword Trans. If the input keyword niter is set, then an iterative algorithm is applied and if the pos keyword is also set, then a positivity constraint is added.

USAGE: mrsp_wtfilter, Imag, Filter, NbrScale=NbrScale, NSigma=NSigma, SigmaNoise=SigmaNoise, KillLastScale=KillLastScale, pos=pos, mad=mad, Trans=Trans, niter=niter, FirstScale=FirstScale, Use_FdrAll=Use_FdrAll, soft=soft, fdr=fdr, lmax=lmax, FilterLast=FilterLast, mask=mask

where

- *Imag* : Input 3D IDL array of healpix polarized map in TQU scheme. Input image to be filtered.
- *Filter* : Output 3D IDL array of healpix polarized map in TQU scheme containing the filtered map.
- *NbrScale* : int, number of scales (default is 4).
- *NSigma* : float, level of thresholding (default is 3).
- *SigmaNoise* : float, noise standard deviation. Default is automatically estimated.
- *mad* : scalar, if set the noise level is derived at each scale using the MAD of the wavelet coefficient.
- *KillLastScale* : scalar, if set the last scale is set to zero.
- *niter* : int, number of iterations used in the reconstruction.
- *pos* : scalar, if set the solution is assumed to be positive.
- *FirstScale* : int, consider only scales larger than FirstScale. Default is 1 (i.e. all scales are used).

- *Soft* : scalar, if set use soft thresholding instead of hard thresholding.
- *fdr* : float between 0 (default) and 1 (max, if greater or equal to 1, set to 0.05), used to estimate a threshold level instead of a NSigma threshold, threshold is applied from scale j=FirstScale to the last.
- *Use_FdrAll* : same as fdr but applied to all scales.
- *FilterLast* : scalar, if set the last scale is filtered.
- *mask* : IDL array of healpix map, input mask applied.
- *lmax* : int, maximum l value in the Spherical Harmonic Space.
- *Trans* : IDL structure: Thresholded wavelet decomposition of the input image.

Example:

- mrsp_wtfilter, Imag, Filter, NbrScale=5, Nsigma=5
  Wavelet filtering with five scales and a 5 sigma threshold.

11.6.2 Thresholding in polarized wavelet or curvelet space : mrsp_threshold

Threshold the decomposition coefficients of a polarized healpix map. This routine works with several decompositions (see mrsp_trans).

USAGE: mrsp_threshold, Trans, NSigma=Nsigma, Mad=Mad, KillLastScale=KillLastScale

where where

- *Trans* : IDL structures obtained from the *mrsp_trans* routine.
- *Nsigma*: Level of thresholding (default is 3)
- *KillLastScale*: if set, the last scale is set to zero

Examples:

Compute the undecimated wavelet transform of a vector field I with five scales (needlet filters). The result is stored in WT, then wavelet coefficients are threshold at 2 sigma, and the filtered polarized map is reconstructed.

- mrsp_trans, I, WT, NbrScale=5, /UWT, /NeedletWave
- mrsp_threshold, WT, NSigma=2
- mrsp_rec, WT, RecPola

Compute the curvelet transform of a vector field I with five scales. The result is stored in C, then curvelet coefficients are threshold at 5 sigma, and the filtered polarized map is reconstructed.

- mrsp_trans, I, C, NbrScale=5, /Cur
- mrsp_threshold, C, NSigma=5
- mrsp_rec, C, RecPola

Part III

MRS-MSVST/V1.1 : Multi-Scale Variance Stabilizing Transform on the Sphere

Chapter 12

Introduction

Several techniques have been proposed in the literature to estimate Poisson intensity in 2D. A major class of methods adopt a multiscale bayesian framework specifically tailored for Poisson data (Nowak and Kolaczyk 2000), independently initiated by Timmermann and Nowak (1999) and Kolaczyk (1999). Lefkimmiaits et al. (2009) proposed an improved bayesian framework for analyzing Poisson processes, based on a multiscale representation of the Poisson process in which the ratios of the underlying Poisson intensities in adjacent scales are modeled as mixtures of conjugate parametric distributions. Another approach includes preprocessing the count data by a variance stabilizing transform (VST) such as the Anscombe (Anscombe 1948) and the Fisz (Fryźlewicz and Nason 2004b) transforms, applied respectively in the spatial (Donoho 1993) or in the wavelet domain (Fryźlewicz and Nason 2004a). The transform reforms the data so that the noise approximately becomes Gaussian with a constant variance. Standard techniques for independant identically distributed Gaussian noise are then used for denoising. Zhang et al. (2008) proposed a powerful method called Multi-Scale Variance Stabilizing Tranform (MS-VST). It consists in combining a VST with a multiscale transform (wavelets, ridgelets or curvelets), yielding asymptotically normally distributed coefficients with known variances. The choice of the multi-scale method depends on the morphology of the data. Wavelets represent more efficiently regular structures and isotropic singularities, whereas ridgelets are designed to represent global lines in an image, and curvelets represent efficiently curvilinear contours. Significant coefficients are then detected with binary hypothesis testing, and the final estimate is reconstructed with an iterative scheme. In Starck et al. (2009a), it was shown that sources can be detected in 3D FERMI LAT data (2D+time or 2D+energy) using a specific 3D extension of the MS-VST. Schmitt et al. (2010) proposed a method for Poisson intensity estimation on spherical data called Multi-Scale Variance Stabilizing Transform on the Sphere (MS-VSTS). This MS-VSTS (Multi-Scale Variance Stabilizing Transform on the Sphere) package offers a Poisson denoising method on the sphere, designed for Fermi photon counts maps. This method is based on the MS-VST (Zhang et al. 2008) and on multi-scale transforms on the sphere (Starck et al. 2007b; Abrial et al. 2007; Abrial et al. 2008). Chapter 13 introduces the MS-VSTS. Chapter 14 applies the MS-VSTS to spherical data restoration. Chapter 15 applies the MS-VSTS to inpainting. Chapter 16 applies the MS-VSTS to background extraction. An accurate description of the IDL routines that makeup this package is given in Chapter 18. An extension to multichannel denoising and deconvolution is given in Chapter 17 Conclusions are drawn in Chapter 19. All experiments were performed on HEALPix maps with $nside = 128$ (Górski et al. 2005),

which corresponds to a good pixelisation choice for data such as the GLAST/FERMI resolution. The performance of the method is not dependent on the nside parameter. For a given data set, if nside is small, it just means that we don't want to investigate the finest scales. If nside is large, the number of counts per pixel will be very small, and we may not have enough statistics to get any information at the finest resolution levels. But it will not have any bad effect on the solution. Indeed, the finest scales will be smoothed, since our algorithm will not detect any significant wavelet coefficients in the finest scales. Hence, starting with a fine pixelisation (i.e. large nside), our method will provide a kind of automatic binning, by thresholding wavelets coefficients at scales and at spatial positions where the number of counts is not sufficient.

Chapter 13

Multi-Scale Variance Stabilizing Transform on the Sphere (MS-VSTS)

13.1 Principle of VST

13.1.1 VST of a Poisson process

Given Poisson data $\mathbf{Y} := (Y_i)_i$, each sample $Y_i \sim \mathcal{P}(\lambda_i)$ has a variance $\mathrm{Var}[Y_i] = \lambda_i$. Thus, the variance of $\mathbf{Y}$ is signal-dependant. The aim of a VST $\mathbf{T}$ is to stabilize the data such that each coefficient of $\mathbf{T}(\mathbf{Y})$ has an (asymptotically) constant variance, say 1, irrespective of the value of λ_i. In addition, for the VST used in this study, $T(\mathbf{Y})$ is asymptotically normally distributed. Thus, the VST-transformed data are asymptotically stationary and gaussian.

The Anscombe transform (Anscombe 1948) is a widely used VST which has a simple square-root form

$$\mathbf{T}(Y) := 2\sqrt{Y + 3/8}. \tag{13.1}$$

We can show that $\mathbf{T}(Y)$ is asymptotically normal as the intensity increases.

$$\mathbf{T}(Y) - 2\sqrt{\lambda} \xrightarrow[\lambda \to +\infty]{\mathcal{D}} \mathcal{N}(0,1) \tag{13.2}$$

It can be shown that the Anscombe VST requires a high underlying intensity to well stabilize the data (typically for $\lambda \geqslant 10$) (Zhang et al. 2008).

13.1.2 VST of a filtered Poisson process

Let $Z_j := \sum_i h[i] Y_{j-i}$ be the filtered process obtained by convolving $(Y_i)_i$ with a discrete filter h. We will use Z to denote any of the Z_j's. Let us define $\tau_k := \sum_i (h[i])^k$ for $k = 1, 2, \cdots$. In addition, we adopt a local homogeneity assumption stating that $\lambda_{j-i} = \lambda$ for all i within the support of h.

We define the square-root transform T as follows:

$$T(Z) := b \cdot \mathrm{sign}(Z + c)|Z + c|^{1/2}, \tag{13.3}$$

where b is a normalizing factor.
(Square root as VST) If $\tau_1 \neq 0$, $\|h\|_2, \|h\|_3 < \infty$, then we have :

$$\begin{aligned} &\mathrm{sign}(Z+c)\sqrt{|Z+c|} - \mathrm{sign}(\tau_1)\sqrt{|\tau_1|\lambda} \\ &\qquad \xrightarrow[\lambda \to +\infty]{\mathcal{D}} \mathcal{N}\Big(0, \frac{\tau_2}{4|\tau_1|}\Big). \end{aligned} \tag{13.4}$$

This proves that T is a VST for a filtered Poisson process (with a nonzero-mean filter) in that $T(Y)$ is asymptotically normally distributed with a stabilized variance as λ becomes large (see Zhang et al. (2008) for a proof).

13.2 MS-VSTS

The MS-VSTS (Schmitt et al. 2010) consists in combining the square-root VST with a multi-scale transform on the sphere.

13.2.1 MS-VSTS + IUWT

This section describes the MS-VSTS + IUWT, which is a combination of a square-root VST with the IUWT. The recursive scheme is:

$$\begin{aligned} &\text{IUWT} \left\{ \begin{array}{rcl} a_j &=& h_{j-1} * a_{j-1} \\ d_j &=& a_{j-1} - a_j \end{array} \right. \\ \Longrightarrow &\begin{array}{l}\text{MS-VSTS} \\ \text{+ IUWT}\end{array} \left\{ \begin{array}{rcl} a_j &=& h_{j-1} * a_{j-1} \\ d_j &=& T_{j-1}(a_{j-1}) - T_j(a_j) \end{array} \right. . \end{aligned} \tag{13.5}$$

In (13.5), the filtering on a_{j-1} can be rewritten as a filtering on $a_0 := \mathbf{Y}$, i.e., $a_j = h^{(j)} * a_0$, where $h^{(j)} = h_{j-1} * \cdots * h_1 * h_0$ for $j \geqslant 1$ and $h^{(0)} = \delta$, where δ is the Dirac pulse ($\delta = 1$ on a single pixel and 0 everywhere else). T_j is the VST operator at scale j:

$$T_j(a_j) = b^{(j)} \mathrm{sign}(a_j + c^{(j)}) \sqrt{|a_j + c^{(j)}|}. \tag{13.6}$$

Let us define $\tau_k^{(j)} := \sum_i (h^{(j)}[i])^k$. In Zhang et al. (2008), it has ben shown that, to have an optimal convergence rate for the VST, the constant $c^{(j)}$ associated to $h^{(j)}$ should be set to:

$$c^{(j)} := \frac{7\tau_2^{(j)}}{8\tau_1^{(j)}} - \frac{\tau_3^{(j)}}{2\tau_2^{(j)}}. \tag{13.7}$$

The MS-VSTS+IUWT procedure is directly invertible as we have:

$$a_0(\theta,\varphi) = T_0^{-1}\left[T_J(a_J) + \sum_{j=1}^{J} d_j\right](\theta,\varphi). \tag{13.8}$$

Setting $b^{(j)} := \mathrm{sgn}(\tau_1^{(j)})/\sqrt{|\tau_1^{(j)}|}$, if λ is constant within the support of the filter. $h^{(j)}$, then

we have (Zhang et al. 2008):

$$d_j(\theta,\varphi) \xrightarrow[\lambda \to +\infty]{\mathcal{D}} \mathcal{N}\left(0, \frac{\tau_2^{(j-1)}}{4\tau_1^{(j-1)^2}} + \frac{\tau_2^{(j)}}{4\tau_1^{(j)^2}} - \frac{\langle h^{(j-1)}, h^{(j)}\rangle}{2\tau_1^{(j-1)}\tau_1^{(j)}}\right), \tag{13.9}$$

where $\langle ., . \rangle$ denotes inner product.

It means that the detail coefficients issued from locally homogeneous parts of the signal follow asymptotically a central normal distribution with an intensity-independant variance which relies solely on the filter h and the current scale for a given filter h. Consequently, the stabilized variances and the constants $b^{(j)}$,$c^{(j)}$,$\tau_k^{(j)}$ can all be pre-computed. Let us define $\sigma_{(j)}^2$ the stabilized variance at scale j for a locally homogeneous part of the signal:

$$\sigma_{(j)}^2 = \frac{\tau_2^{(j-1)}}{4\tau_1^{(j-1)^2}} + \frac{\tau_2^{(j)}}{4\tau_1^{(j)^2}} - \frac{\langle h^{(j-1)}, h^{(j)}\rangle}{2\tau_1^{(j-1)}\tau_1^{(j)}}. \tag{13.10}$$

To compute the $\sigma_{(j)}$, $b^{(j)}$,$c^{(j)}$,$\tau_k^{(j)}$, we only have to know the filters $h^{(j)}$. We compute these filters thanks to the formula $a_j = h^{(j)} * a_0$, by applying the IUWT to a Dirac pulse $a_0 = \delta$. Then, the $h^{(j)}$ are the scaling coefficients of the IUWT. The $\sigma_{(j)}$ have been precomputed for a 6-scaled IUWT (Table 13.1).

Table 13.1: Precomputed values of the variances σ_j of the wavelet coefficients.

Wavelet scale j	Value of σ_j
1	0.484704
2	0.0552595
3	0.0236458
4	0.0114056
5	0.00567026

We have simulated Poisson images of different constant intensities λ, computed the IUWT with MS-VSTS on each image and observed the variation of the normalized value of $\sigma_{(j)}$ $((\sigma_{(\mathbf{j})})_{\text{simulated}}/(\sigma_{(j)})_{\text{theoretical}})$ as a function of λ for each scale j (Fig. 13.1). We see that the wavelet coefficients are stabilized when $\lambda \gtrsim 0.1$ except for the first wavelet scale, which is mostly constituted of noise. On Fig. 13.2, we compare the result of MS-VSTS with Anscombe + wavelet shrinkage, on sources of varying intensities. We see that MS-VSTS works well on sources of very low intensities, whereas Anscombe doesn't work when the intensity is too low.

13.2.2 MS-VSTS + Curvelets

As the first step of the algorithm is an IUWT, we can stabilize each resolution level as in Equation (13.5). We then apply the local ridgelet transform on each stabilized wavelet band.

It is not as straightforward as with the IUWT to derive the asymptotic noise variance in the stabilized curvelet domain. In our experiments, we derived them using simulated

Poisson data of stationary intensity level λ. After having checked that the standard deviation in the curvelet bands becomes stabilized as the intensity level increases (which means that the stabilization is working properly), we stored the standard deviation $\sigma_{j,l}$ for each wavelet scale j and each ridgelet band l (Table 13.2).

Table 13.2: Asymptotic values of the variances $\sigma_{j,k}$ of the curvelet coefficients.

j	$l=1$	$l=2$	$l=3$	$l=4$
1	1.74550	0.348175		
2	0.230621	0.248233	0.196981	
3	0.0548140	0.0989918	0.219056	
4	0.0212912	0.0417454	0.0875663	0.20375
5	0.00989616	0.0158273	0.0352021	0.163248

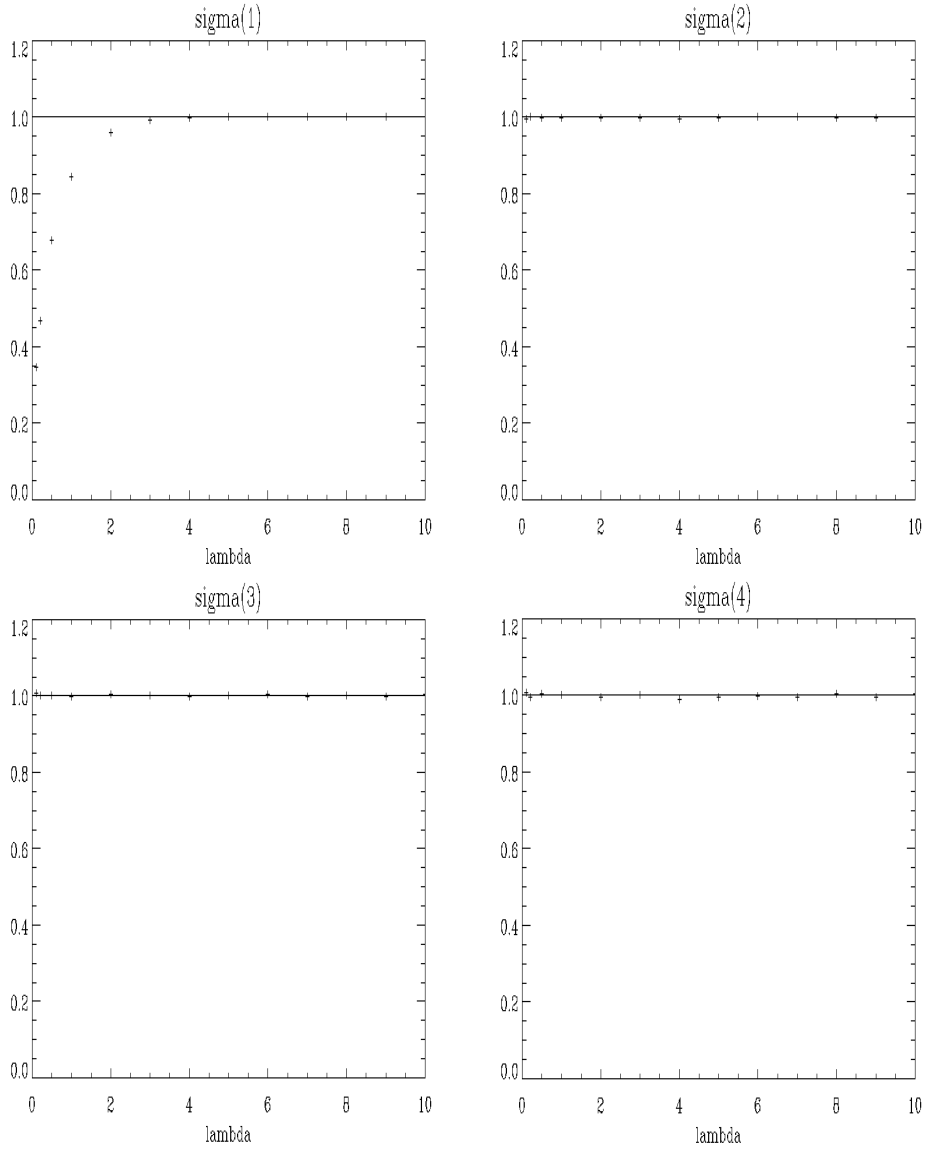


Figure 13.1: Normalized value $((\sigma_{(\mathbf{j})})_{\text{simulated}}/(\sigma_{(j)})_{\text{theoretical}})$ of the stabilized variances at each scale j as a function of λ.

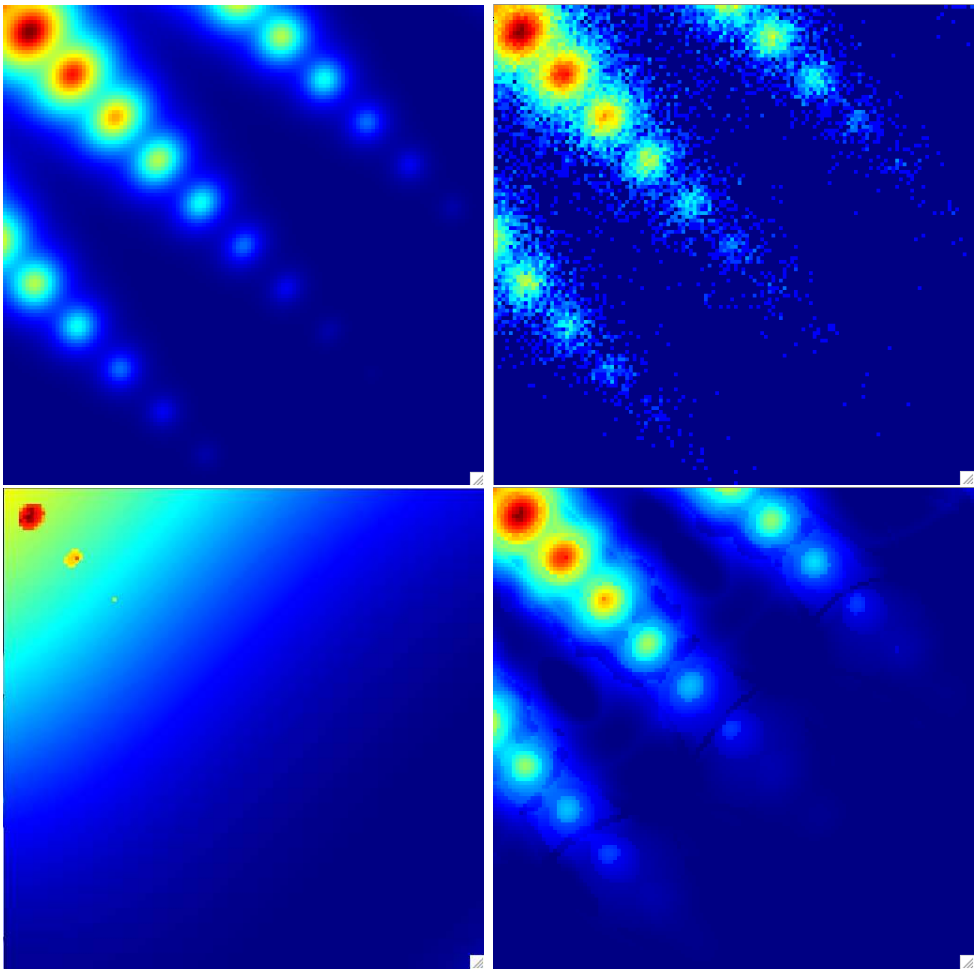


Figure 13.2: Comparison of MS-VSTS with Anscombe + wavelet shrinkage on a single HEALPix face. Top Left : Sources of varying intensity. Top Right : Sources of varying intensity with Poisson noise. Bottom Left : Poisson sources of varying intensity reconstructed with Anscombe + wavelet shrinkage. Bottom Right : Poisson sources of varying intensity reconstructed with MS-VSTS.

Chapter 14

Poisson denoising

14.1 MS-VST + IUWT

Under the hypothesis of homogeneous Poisson intensity, the stabilized wavelet coefficients d_j behave like centered Gaussian variables of standard deviation $\sigma_{(j)}$. We can detect significant coefficients with binary hypothesis testing as in Gaussian denoising.

Under the null hypothesis $\mathcal{H}_0$ of homogeneous Poisson intensity, the distribution of the stabilized wavelet coefficient $d_j[k]$ at scale j and location index k can be written as:

$$p(d_j[k]) = \frac{1}{\sqrt{2\pi}\sigma_j} \exp(-d_j[k]^2/2\sigma_j^2). \tag{14.1}$$

The rejection of the hypothesis $\mathcal{H}_0$ depends on the double-sided p-value:

$$p_j[k] = 2\frac{1}{\sqrt{2\pi}\sigma_j} \int_{|d_j[k]|}^{+\infty} \exp(-x^2/2\sigma_j^2)dx. \tag{14.2}$$

Consequently, to accept or reject $\mathcal{H}_0$, we compare each $|d_j[k]|$ with a critical threshold $\kappa\sigma_j$, $\kappa = 3, 4$ or 5 corresponding respectively to significance levels. This amounts to deciding that:

- if $|d_j[k]| \geqslant \kappa\sigma_j$, $d_j[k]$ is significant.
- if $|d_j[k]| < \kappa\sigma_j$, $d_j[k]$ is not significant.

Then we have to invert the MS-VSTS scheme to reconstruct the estimate. However, although the direct inversion is possible (Eq. (13.8)), it can not guarantee a positive intensity estimate, while the Poisson intensity is always nonnegative. A positivity projection can be applied, but important structures could be lost in the estimate. To tackle this problem, we reformulate the reconstruction as a convex optimisation problem and solve it iteratively with an algorithm based on Hybrid Steepest Descent (HSD) (Yamada 2001).

We define the multiresolution support $\mathcal{M}$, which is determined by the set of detected significant coefficients after hypothesis testing:

$$\mathcal{M} := \{(j,k)|\text{if } d_j[k] \text{ is declared significant}\}. \tag{14.3}$$

We formulate the reconstruction problem as a convex constrained minimization problem:

$$\begin{array}{c} \mathrm{Arg}\min_{\mathbf{X}} \|\mathbf{\Phi}^T\mathbf{X}\|_1, \text{s.t.} \\ \left\{ \begin{array}{c} \mathbf{X} \geqslant 0, \\ \forall (j,k) \in \mathcal{M}, (\mathbf{\Phi}^T\mathbf{X})_j[k] = (\mathbf{\Phi}^T\mathbf{Y})_j[k], \end{array} \right. \end{array} \tag{14.4}$$

where $\mathbf{\Phi}$ denotes the IUWT synthesis operator.

This problem is solved with the following iterative scheme: the image is initialised by $\mathbf{X}^{(0)} = 0$, and the iteration scheme is, for $n = 0$ to $N_{\max} - 1$:

$$\tilde{\mathbf{X}} = P_+[\mathbf{X}^{(n)} + \mathbf{\Phi} P_{\mathcal{M}} \mathbf{\Phi}^T(\mathbf{Y} - \mathbf{X}^{(n)})] \tag{14.5}$$

$$\mathbf{X}^{(n+1)} = \mathbf{\Phi}\mathrm{ST}_{\lambda_n}[\mathbf{\Phi}^T\tilde{\mathbf{X}}] \tag{14.6}$$

where P_+ denotes the projection on the positive orthant, $P_{\mathcal{M}}$ denotes the projection on the multiresolution support $\mathcal{M}$:

$$P_{\mathcal{M}} d_j[k] = \left\{ \begin{array}{cc} d_j[k] & \text{if } (j,k) \in \mathcal{M}, \\ 0 & \text{otherwise} \end{array} \right. . \tag{14.7}$$

and ST_{λ_n} the soft-thresholding with threshold λ_n:

$$\mathrm{ST}_{\lambda_n}[d] = \left\{ \begin{array}{cc} \mathrm{sign}(d)(|d| - \lambda_n) & \text{if } |d| \geqslant \lambda_n, \\ 0 & \text{otherwise} \end{array} \right. . \tag{14.8}$$

We chose a decreasing threshold $\lambda_n = \frac{N_{\max} - n}{N_{\max} - 1}, n = 1, 2, \cdots, N_{\max}$.

The final estimate of the Poisson intensity is: $\hat{\mathbf{\Lambda}} = \mathbf{X}^{(N_{\max})}$. Algorithm 12 summarizes the main steps of the MS-VSTS + IUWT denoising algorithm.

Algorithm 12 MS-VSTS + IUWT Denoising

Require: data $a_0 := \mathbf{Y}$, number of iterations $N_{\max}$, threshold κ

Detection

for $j = 1$ to J **do**
  Compute a_j and d_j using (13.5).
  Hard threshold $|d_j[k]|$ with threshold $\kappa\sigma_j$ and update $\mathcal{M}$.
end for

Estimation

Initialize $\mathbf{X}^{(0)} = 0$, $\lambda_0 = 1$.
for $n = 0$ to $N_{\max} - 1$ **do**
  $\tilde{\mathbf{X}} = P_+[\mathbf{X}^{(n)} + \mathbf{\Phi} P_{\mathcal{M}} \mathbf{\Phi}^T(\mathbf{Y} - \mathbf{X}^{(n)})]$.
  $\mathbf{X}^{(n+1)} = \mathbf{\Phi}\mathrm{ST}_{\lambda_n}[\mathbf{\Phi}^T\tilde{\mathbf{X}}]$.
  $\lambda_{n+1} = \frac{N_{\max} - (n+1)}{N_{\max} - 1}$.
end for
Get the estimate $\hat{\mathbf{\Lambda}} = \mathbf{X}^{(N_{\max})}$.

14.2 Multi-resolution support adaptation

When two sources are too close, the less intense source may not be detected because of the negative wavelet coefficients of the brightest source. To avoid such a drawback, we may update the multi-resolution support at each iteration. The idea is to withdraw the detected sources and to make a detection on the remaining residual, so as to detect the sources which may have been missed at the first detection.

At each iteration n, we compute the MS-VSTS of $\mathbf{X}^{(n)}$. We denote $d_j^{(n)}[k]$ the stabilised coefficients of $\mathbf{X}^{(n)}$. We make a hard thresholding on $(d_j[k] - d_j^{(n)}[k])$ with the same thresholds as in the detection step. Significant coefficients are added to the multiresolution support $\mathcal{M}$.

Algorithm 13 MS-VSTS + IUWT Denoising + Multiresolution Support Adaptation

Require: data $a_0 := \mathbf{Y}$, number of iterations $N_{\max}$, threshold κ

Detection

for $j = 1$ to J **do**
  Compute a_j and d_j using (13.5).
  Hard threshold $|d_j[k]|$ with threshold $\kappa\sigma_j$ and update $\mathcal{M}$.
end for

Estimation

Initialize $\mathbf{X}^{(0)} = 0$, $\lambda_0 = 1$.
for $n = 0$ to $N_{\max} - 1$ **do**
  $\tilde{\mathbf{X}} = P_+[\mathbf{X}^{(n)} + \mathbf{\Phi} P_{\mathcal{M}} \mathbf{\Phi}^T(\mathbf{Y} - \mathbf{X}^{(n)})]$.
  $\mathbf{X}^{(n+1)} = \mathbf{\Phi}\mathrm{ST}_{\lambda_n}[\mathbf{\Phi}^T\tilde{\mathbf{X}}]$.
  Compute the MS-VSTS on $\mathbf{X}^{(n)}$ to get the stabilised coeffcients $d_j^{(n)}$.
  Hard threshold $|d_j[k] - d_j^{(n)}[k]|$ and update $\mathcal{M}$.
  $\lambda_{n+1} = \frac{N_{\max}-(n+1)}{N_{\max}-1}$.
end for
Get the estimate $\hat{\mathbf{\Lambda}} = \mathbf{X}^{(N_{\max})}$.

The main steps of the algorithm are summarized in Algorithm 13. In practice, we use Algorithm 13 instead of Algorithm 12 in our experiments.

14.3 MS-VST + Curvelets

Insignificant coefficients are zeroed by using the same hypothesis testing framework as in the wavelet scale. At each wavelet scale j and ridgelet band k, we make a hard thresholding on curvelet coefficients with threshold $\kappa\sigma_{j,k}$, $\kappa = 3, 4$ or 5. Finally, a direct reconstruction can be performed by first inverting the local ridgelet transforms and then inverting the MS-VST + IUWT (Equation (13.8)). An iterative reconstruction may also be performed.

Algorithm 14 summarizes the main steps of the MS-VSTS + Curvelets denoising algorithm.

Algorithm 14 MS-VSTS + Curvelets Denoising

1: Apply the MS-VST + IUWT with J scales to get the stabilized wavelet subbands d_j.
2: Set $B_1 = B_{\min}$.
3: **for** $j = 1$ to J **do**
4: Partition the subband d_j with blocks of side-length B_j and apply the digital ridgelet transform to each block to obtain the stabilized curvelets coefficients.
5: **if** j modulo $2 = 1$ **then**
6: $B_{j+1} = 2B_j$
7: **else**
8: $B_{j+1} = B_j$
9: **end if**
10: HTs on the stabilized curvelet coefficients.
11: **end for**
12: Invert the ridgelet transform in each block before inverting the MS-VST + IUWT.

14.4 Experiments

The method was tested on simulated Fermi data. The simulated data are the sum of a Milky Way diffuse background model and 1000 gamma ray point sources. We based our Galactic diffuse emission model intensity on the model gll_iem_v02 obtained at the Fermi Science Support Center (Myers 2009) . This model results from a fit of the LAT photons with various gas templates as well as inverse Compton in several energy bands. We used a realistic point-spread function for the sources, based on Monte Carlo simulations of the LAT and accelerator tests, that scale approximately as $0.8(E/1GeV)^{-0.8}$ degrees. The position of the 205 brightest sources were taken from the Fermi 3-month source list (Abdo et al. 2009). The position of the 795 remaining sources follow the LAT 1-year Point Source Catalog (Myers 2010) sources distribution: each simulated source was randomly sorted in a box of Δl=5^o and Δb=1^o around a LAT 1-year catalog source. We simulated each source assuming a power-law dependence with its spectral index given by the 3-month source list and the first year catalog. We used an exposure of $3.10^{10} s.cm^2$ corresponding approximatively to one year of Fermi all-sky survey around 1 GeV. The simulated counts map shown here correspond to photons energy from 150 MeV to 20 GeV.

Fig. 14.1 compares the result of denoising with MS-VST + IUWT (Algorithm 12), MS-VST + curvelets (Algorithm 14) and Anscombe VST + wavelet shrinkage on a simulated Fermi map. Fig. 14.2 shows one HEALPix face of the results. As expected from theory, the Anscombe method produces poor results to denoise Fermi data, because the underlyning intensity is too weak. Both wavelet and curvelet denoising on the sphere perform much better. For this application, wavelets are slightly better than curvelets ($SNR_{wavelets} = 65.8dB$, $SNR_{curvelets} = 37.3dB$, $SNR(dB) = 20\log(\sigma_{signal}/\sigma_{noise})$). As this image contains many point sources, thisresult is expected. Indeed wavelet are better than curvelets to represent isotropic objects.

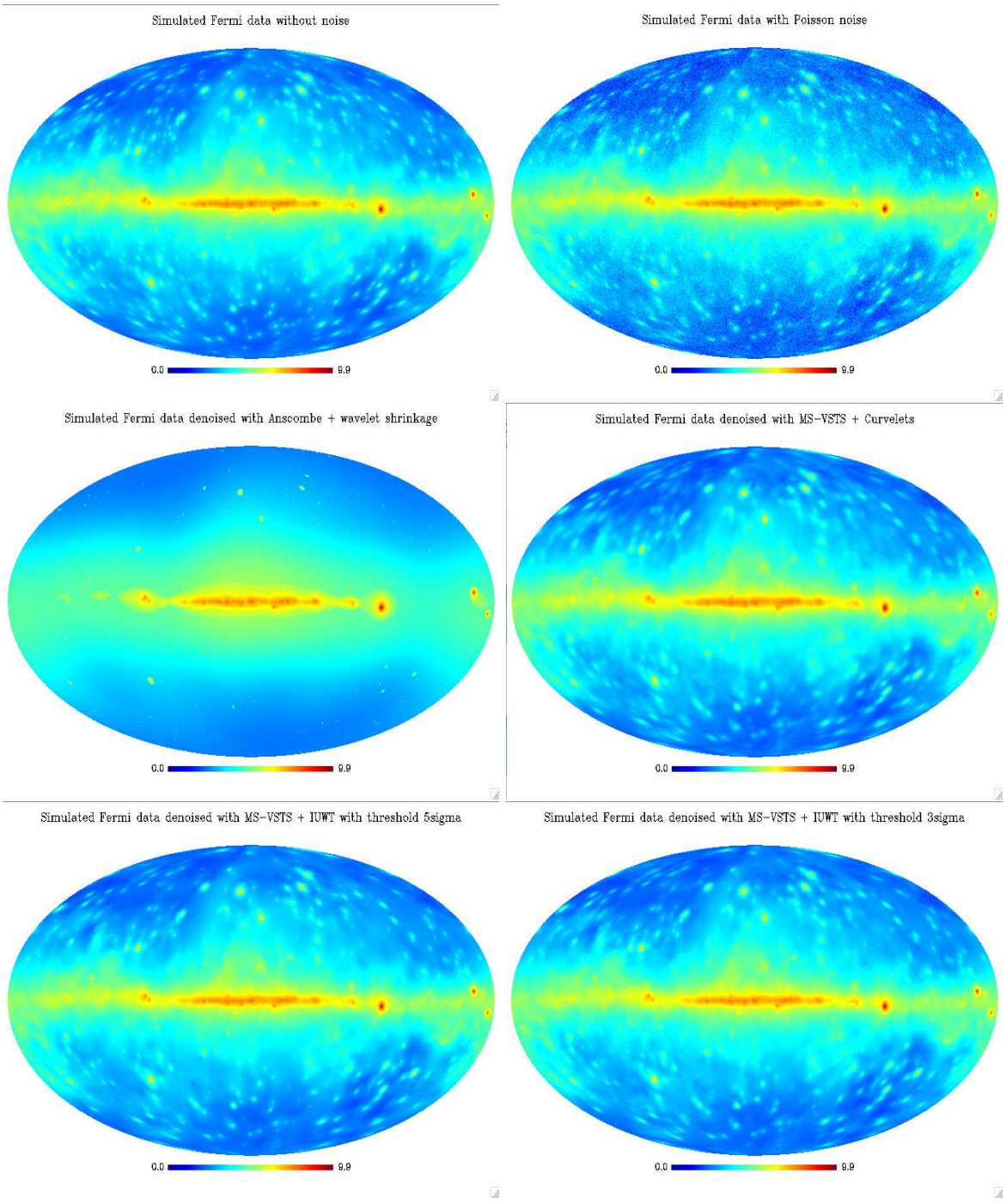


Figure 14.1: Top Left: Fermi simulated map without noise. Top Right: Fermi simulated map with Poisson noise. Middle Left: Fermi simulated map denoised with Anscombe VST + wavelet shrinkage. Middle Right: Fermi simulated map denoised with MS-VSTS + curvelets (Algorithm 14). Bottom Left: Fermi simulated map denoised with MS-VSTS + IUWT (Algorithm 12) with threshold $5\sigma_j$. Bottom Right: Fermi simulated map denoised with MS-VSTS + IUWT (Algorithm 12) with threshold $3\sigma_j$. Pictures are in logarithmic scale.

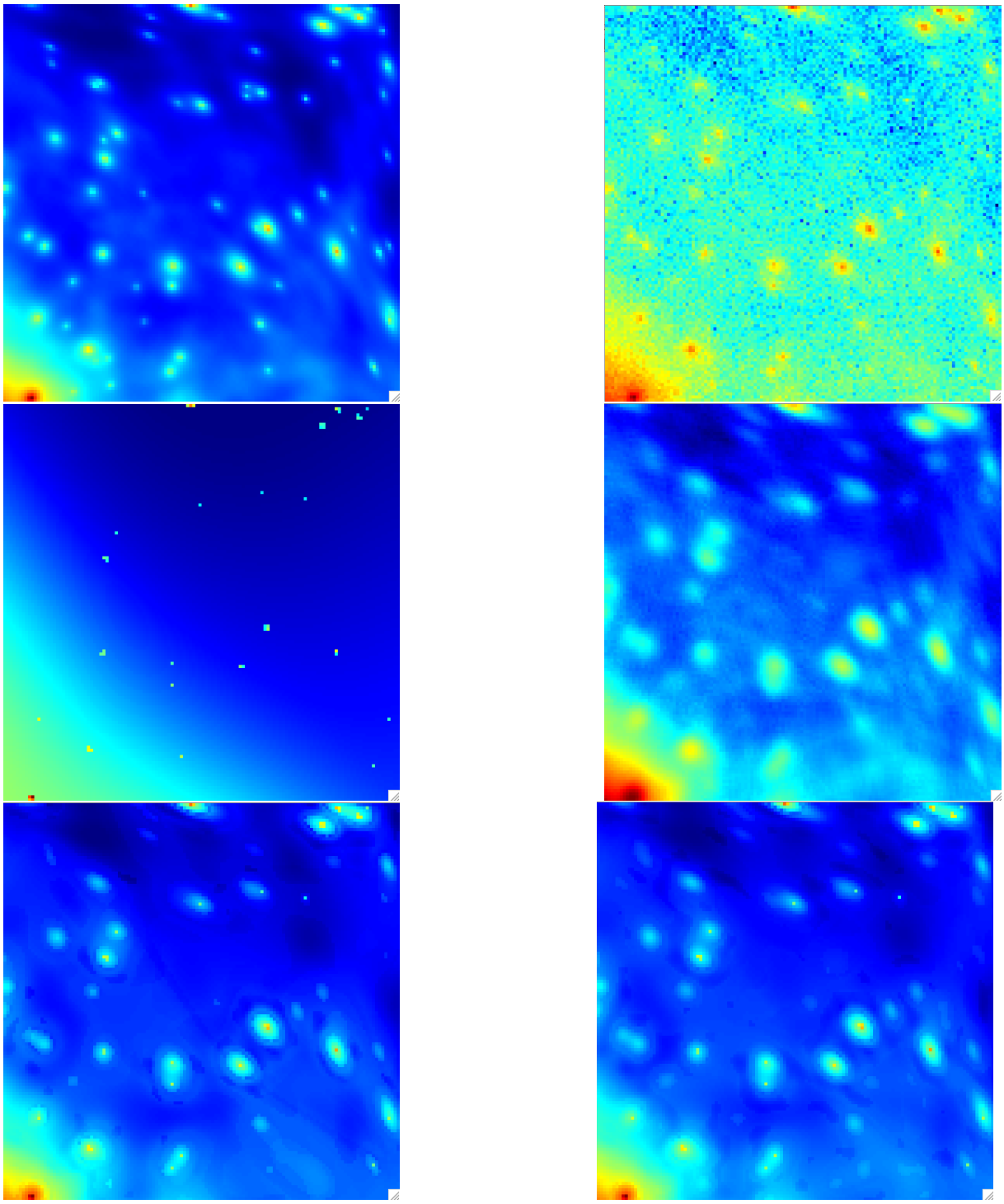


Figure 14.2: View of a single HEALPix face from the results of Figure 14.1. Top Left: Fermi simulated map without noise. Top Right: Fermi simulated map with Poisson noise. Middle Left: Fermi simulated map denoised with Anscombe VST + wavelet shrinkage. Middle Right: Fermi simulated map denoised with MS-VSTS + curvelets (Algorithm 14). Bottom Left: Fermi simulated map denoised with MS-VSTS + IUWT (Algorithm 12) with threshold $5\sigma_j$. Bottom Right: Fermi simulated map denoised with MS-VSTS + IUWT (Algorithm 12) with threshold $3\sigma_j$. Pictures are in logarithmic scale.

Chapter 15

Milky Way diffuse background study: denoising and inpainting

In order to extract a diffuse emission, we want to remove the point sources from the data. As our HSD algorithm is very close to the MCA algorithm (Starck et al. 2004b), an idea is to mask the most intense sources and to modify our algorithm in order to interpolate through the gaps exactly as in the MCA-Inpainting algorithm (Abrial et al. 2007). This modified algorithm can be called MS-VSTS-Inpainting algorithm.

The problem can be reformulated as a convex constrained minimization problem:

$$\begin{array}{c} \text{Arg}\min_{\mathbf{X}} \|\mathbf{\Phi}^T\mathbf{X}\|_1, \text{s.t.} \\ \left\{ \begin{array}{c} \mathbf{X} \geqslant 0, \\ \forall (j,k) \in \mathcal{M}, (\mathbf{\Phi}^T \Pi \mathbf{X})_j[k] = (\mathbf{\Phi}^T\mathbf{Y})_j[k], \end{array} \right. \end{array} \tag{15.1}$$

where Π is a binary mask (1 on valid data and 0 on invalid data).

The iterative scheme can be adapted to cope with a binary mask, which gives:

$$\tilde{\mathbf{X}} = P_+[\mathbf{X}^{(n)} + \mathbf{\Phi} P_{\mathcal{M}} \mathbf{\Phi}^T \Pi(\mathbf{Y} - \mathbf{X}^{(n)})], \tag{15.2}$$

$$\mathbf{X}^{(n+1)} = \mathbf{\Phi}\text{ST}_{\lambda_n}[\mathbf{\Phi}\tilde{\mathbf{X}}]. \tag{15.3}$$

The thresholding strategy has to be adapted. Indeed, for the impainting task we need to have a very large initial threshold in order to have a very smooth image in the beginning and to refine the details progressively. We chose an exponentially decreasing threshold:

$$\lambda_n = \lambda_{\max}(2^{(\frac{N_{\max}-n}{N_{\max}-1})} - 1), n = 1, 2, \cdots, N_{\max}, \tag{15.4}$$

where $\lambda_{\max} = \max(\mathbf{\Phi}^T\mathbf{X})$.

Experiment

We applied this method on simulated Fermi data where we masked the most luminous sources.

The results are on Figure 15.1. The MS-VST + IUWT + Inpainting method (Algorithm 15) interpolates the missing data very well. Indeed, the missing part can not be seen anymore in the inpainted map, which shows that the diffuse emission component has been correctly reconstructed.

Algorithm 15 MS-VST + IUWT Denoising + Inpainting

Require: data $a_0 := \mathbf{Y}$, mask Π, number of iterations $N_{\max}$, threshold κ.

Detection

for $j = 1$ to J **do**
  Compute a_j and d_j using (13.5).
  Hard threshold $|d_j[k]|$ with threshold $\kappa\sigma_j$ and update $\mathcal{M}$.
end for

Estimation

Initialize $\mathbf{X}^{(0)} = 0$, $\lambda_0 = \lambda_{\max}$.
for $n = 0$ to $N_{\max} - 1$ **do**
  $\tilde{\mathbf{X}} = P_+[\mathbf{X}^{(n)} + \mathbf{\Phi} P_{\mathcal{M}} \mathbf{\Phi}^T \Pi(\mathbf{Y} - \mathbf{X}^{(n)})]$.
  $\mathbf{X}^{(n+1)} = \mathbf{\Phi}^{\mathrm{ST}}_{\lambda_n}[\mathbf{\Phi}^T \tilde{\mathbf{X}}]$.
  $\lambda_{n+1} = \lambda_{\max}(2^{(\frac{N_{\max}-(n+1)}{N_{\max}-1})} - 1)$
end for
Get the estimate $\hat{\mathbf{\Lambda}} = \mathbf{X}^{(N_{\max})}$.

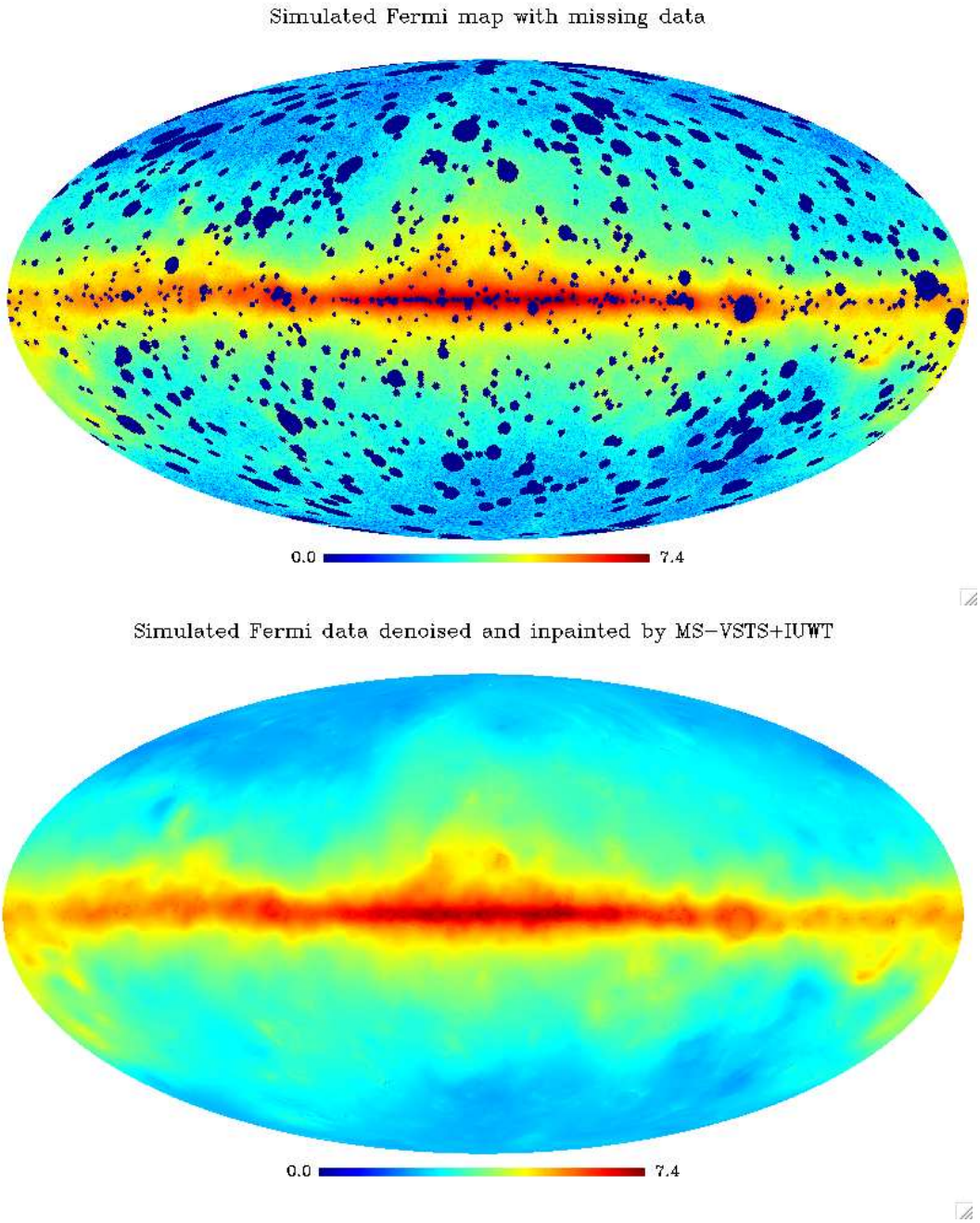


Figure 15.1: MS-VSTS - Inpainting. Top: Fermi simulated map with Poisson noise and the most luminous sources masked. Bottomt: Fermi simulated map denoised and inpainted with wavelets (Algorithm 15). Pictures are in logarithmic scale.

Chapter 16

Source detection: denoising and background modeling

16.1 Method

In some cases such as for Fermi data, the diffuse emission from the Milky Way makes a relatively intense background. We have to extract this background in order to detect point sources. This diffuse interstellar emission may be modeled, and we want to use such a background model and incorporate a background removal in our denoising algorithm.

We note $\mathbf{Y}$ the data, $\mathbf{B}$ the background we want to remove, and $d_j^{(b)}[k]$ the MS-VSTS coefficients of $\mathbf{B}$ at scale j and position k. We determine the multi-resolution support by comparing $|d_j[k] - d_j^{(b)}[k]|$ with $\kappa\sigma_j$.

We formulate the reconstruction problem as a convex constrained minimization problem:

$$\begin{array}{c} \underset{\mathbf{X}}{\text{Arg min}} \|\mathbf{\Phi}^T\mathbf{X}\|_1, \text{s.t.} \\ \left\{ \begin{array}{c} \mathbf{X} \geqslant 0, \\ \forall (j,k) \in \mathcal{M}, (\mathbf{\Phi}^T\mathbf{X})_j[k] = (\mathbf{\Phi}^T(\mathbf{Y}-\mathbf{B}))_j[k], \end{array} \right. \end{array} \tag{16.1}$$

Then, the reconstruction algorithm scheme becomes:

$$\tilde{\mathbf{X}} = P_+[\mathbf{X}^{(n)} + \mathbf{\Phi} P_{\mathcal{M}} \mathbf{\Phi}^T(\mathbf{Y} - \mathbf{B} - \mathbf{X}^{(n)})], \tag{16.2}$$

$$\mathbf{X}^{(n+1)} = \mathbf{\Phi}\text{ST}_{\lambda_n}[\mathbf{\Phi}^T\tilde{\mathbf{X}}]. \tag{16.3}$$

The algorithm is illustrated by the theoretical study in Figure 16.1. We denoise Poisson data while separating a single source, which is a Gaussian of standard deviation equal to 0.01, from a background, which is a sum of two Gaussians of standard deviation equal to 0.1 and 0.01 respectively.

Like Algorithm 12, Algorithm 16 can be adapted to make multiresolution support adaptation.

16.2 Experiment

Algorithm 16 MS-VSTS + IUWT Denoising + Background extraction

Require: data $a_0 := \mathbf{Y}$, background B, number of iterations $N_{\max}$, threshold κ.

Detection

for $j = 1$ to J **do**
  Compute a_j and d_j using (13.5).
  Hard threshold $(d_j[k] - d_j^{(b)}[k])$ with threshold $\kappa\sigma_j$ and update $\mathcal{M}$.
end for

Estimation

Initialize $\mathbf{X}^{(0)} = 0$, $\lambda_0 = 1$.
for $n = 0$ to $N_{\max} - 1$ **do**
  $\tilde{\mathbf{X}} = P_+[\mathbf{X}^{(n)} + \mathbf{\Phi} P_{\mathcal{M}} \mathbf{\Phi}^T (\mathbf{Y} - \mathbf{B} - \mathbf{X}^{(n)})]$.
  $\mathbf{X}^{(n+1)} = \mathbf{\Phi} \mathrm{ST}_{\lambda_n}[\mathbf{\Phi}^T \tilde{\mathbf{X}}]$.
  $\lambda_{n+1} = \frac{N_{\max} - (n+1)}{N_{\max} - 1}$.
end for
Get the estimate $\hat{\mathbf{\Lambda}} = \mathbf{X}^{(N_{\max})}$.

Table 16.1: Percent of true and false detection and signal-noise ratio versus the standard deviation of the Gaussian noise on the background model.

Model error std dev	% of true detect	% of false detect	SNR (dB)
0	59.3%	7.1%	23.8
10	57.0%	11.0%	23.2
20	53.2%	18.9%	22.6
30	49.1%	43.5%	21.7
40	42.3%	44.3%	21.0
50	34.9%	39.0%	20.3
60	30.3%	37.5%	19.5
70	25.0%	34.6%	18.9
80	23.0%	28.5%	18.7
90	23.6%	27.1%	18.3

We applied Algorithms 16 on simulated Fermi data. To test the efficiency of our method, we detect the sources with the SExtractor routine (Bertin and Arnouts 1996), and compare the detected sources with the theoretical sources catalog to get the number of true and false detections. Results are shown on Figures 16.2 and 16.3. The SExtractor method was applied on the first wavelet scale of the reconstructed map, with a detection threshold equal to 1. It has been chosen to optimise the number of true detections. SExtractor makes 593 true detections and 71 false detections on the Fermi simulated map restored with Algorithm 13 among the 1000 sources of the simulation. On noisy data, many fluctuations due to Poisson noise are detected as sources by SExtractor, which leads to a big number of false detections (more than 2000 in the case of Fermi data).

16.2.1 Sensitivity to model errors

As it is difficult to model the background precisely, it is important to study the sensitivity of the method to model errors. We add a stationary Gaussian noise to the background model, we compute the MS-VSTS + IUWT with threshold $3\sigma_j$ on the simulated Fermi

Poisson data with extraction of the noisy background, and we study the percent of true and false detections with respect to the total number of sources of the simulation and the signal-noise ratio ($\mathrm{SNR}(dB) = 20\log(\sigma_{signal}/\sigma_{noise})$) versus the standard deviation of the Gaussian perturbation. Table 16.1 shows that, when the standard deviation of the noise on the background model becomes of the same range as the mean of the Poisson intensity distribution ($\lambda_{\mathrm{mean}} = 68.764$), the number of false detections increases, the number of true detections decreases and the signal noise ratio decreases. While the perturbation is not too strong (standard deviation < 10), the effect of the model error remains low.

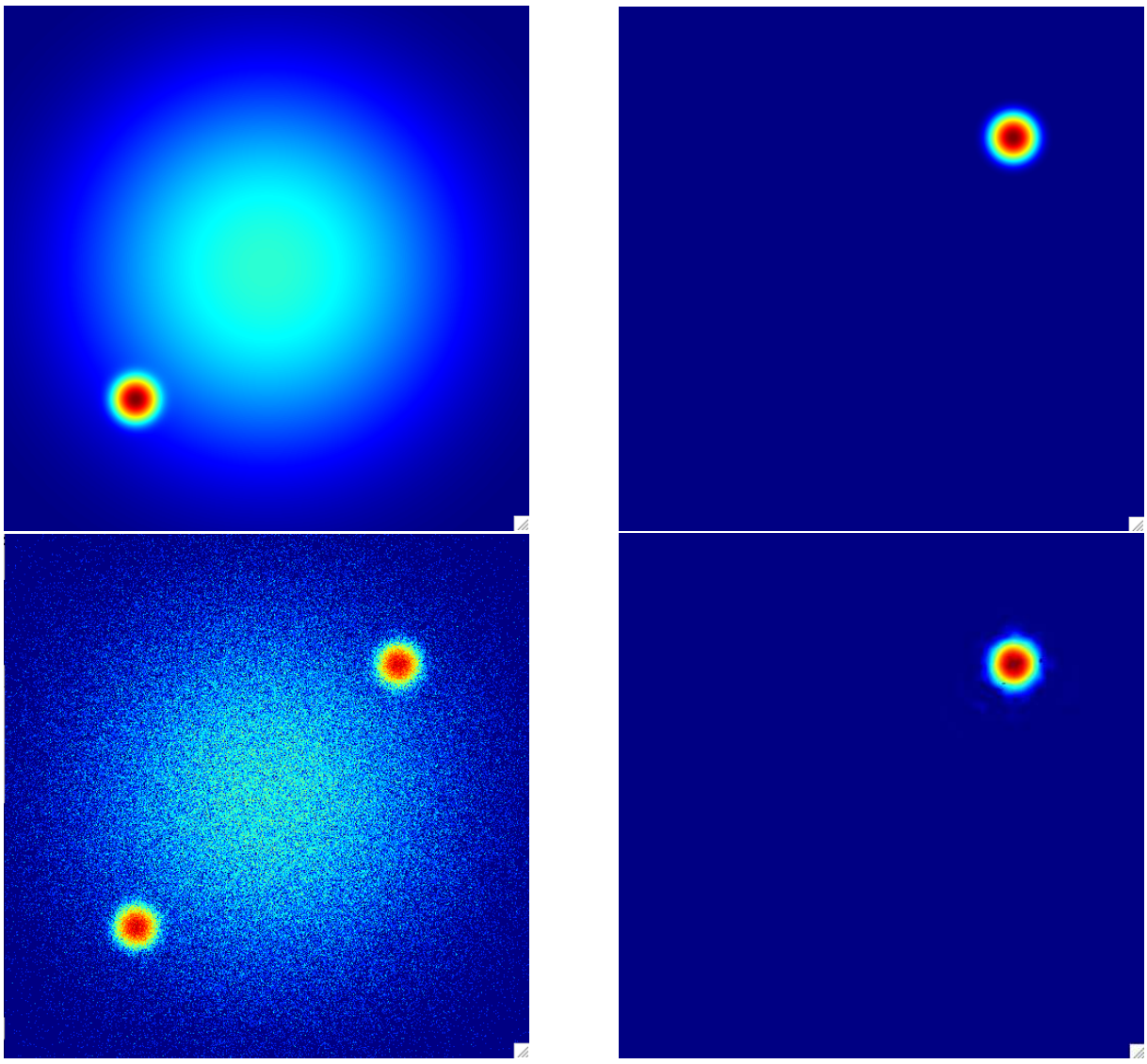


Figure 16.1: Theoretical testing for MS-VSTS + IUWT denoising + background removal algorithm (Algorithm 16). View on a single HEALPix face. Top Left: Simulated background : sum of two Gaussians of standard deviation equal to 0.1 and 0.01 respectively. Top Right: Simulated source: Gaussian of standard deviation equal to 0.01. Bottom Left: Simulated poisson data. Bottom Right: Image denoised with MS-VSTS + IUWT and background removal.

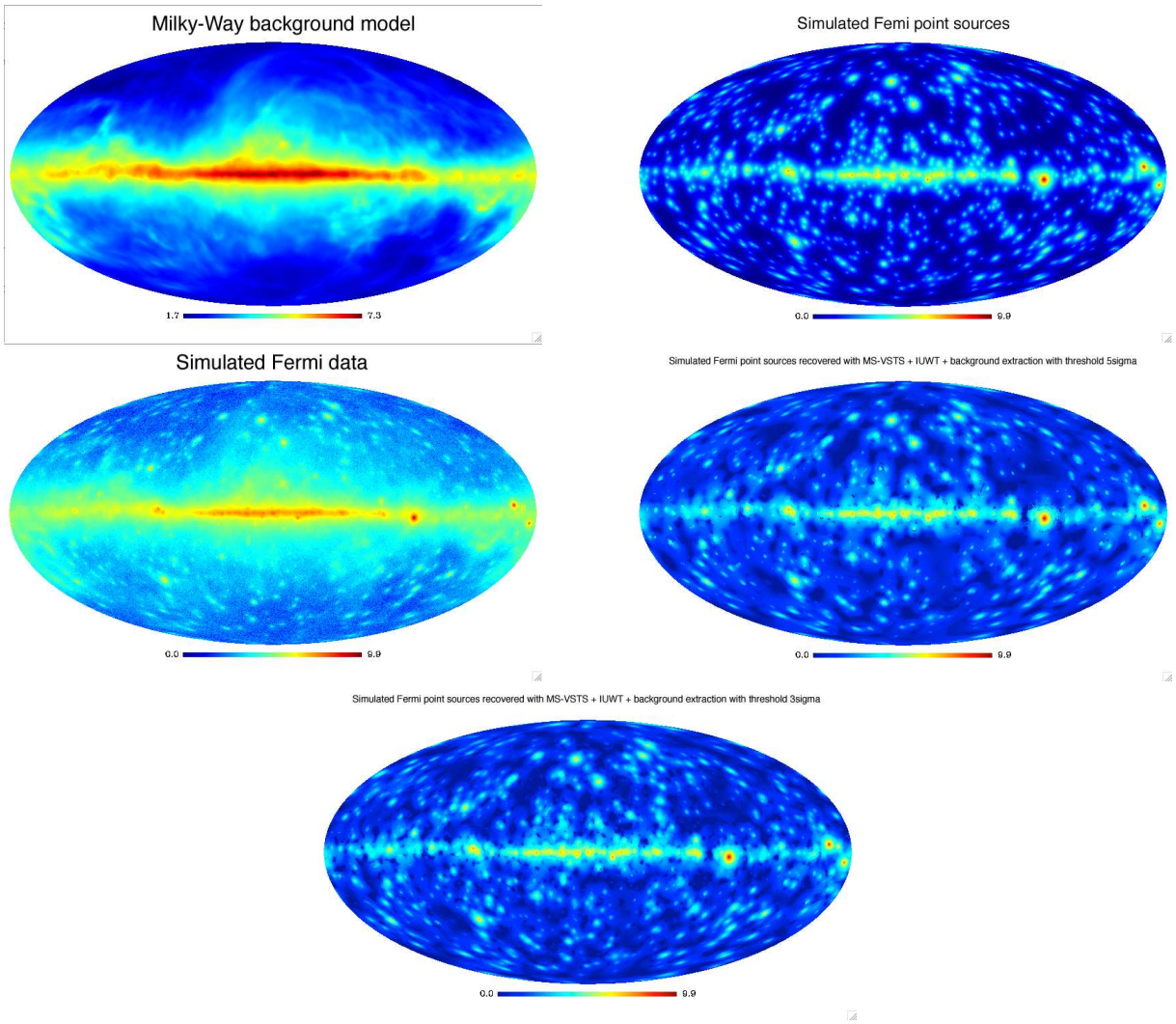


Figure 16.2: Top Left: Simulated background model. Top Right: Simulated Gamma Ray sources. Middle Left: Simulated Fermi data with Poisson noise. Middle Right: Reconstructed Gamma Ray Sources with MS-VSTS + IUWT + background removal (Algorithm 16) with threshold $5\sigma_j$. Bottom: Reconstructed Gamma Ray Sources with MS-VSTS + IUWT + background removal (Algorithm 16) with threshold $3\sigma_j$. Pictures are in logarithmic scale.

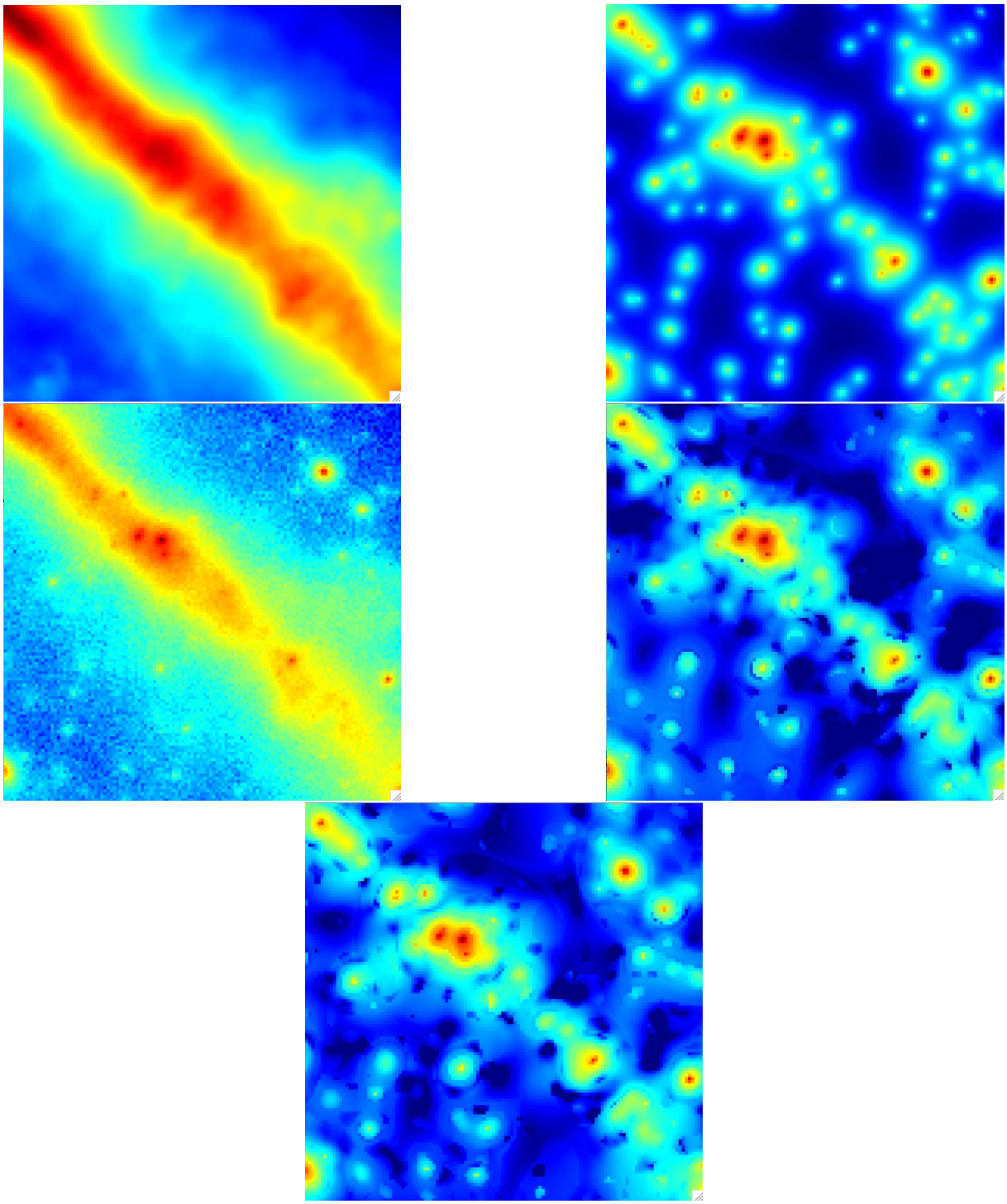


Figure 16.3: View of a single HEALPix face from the results of Figure 16.2.Top Left: Simulated background model. Top Right: Simulated Gamma Ray sources. Middle Left: Simulated Fermi data with Poisson noise. Middle Right: Reconstructed Gamma Ray Sources with MS-VSTS + IUWT + background removal (Algorithm 16) with threshold $5\sigma_j$. Bottom: Reconstructed Gamma Ray Sources with MS-VSTS + IUWT + background removal (Algorithm 16) with threshold $3\sigma_j$. Pictures are in logarithmic scale.

Chapter 17

Multichannel Denoising and Deconvolution on the Sphere

17.1 Introduction

This chapter describes how to remove Poisson noise while deconvolving the effect of the PSF, and presents an multichannel extension of the restoration process. Indeed, some instruments such as FERMI-LAT produce 2D-1D data where the two first dimensions are spatial (longitude and latitude) and the third dimension is either the time or the energy. We present here a multichannel sparse representation for Poisson data combining the MS-VSTS with a spherical 2D-1D wavelet-based deconvolution algorithm. This multiscale representation is used in a wavelet-regularized Richardson-Lucy deconvolution algorithm to remove the effect of the PSF. This method has the advantage to take into account the strong energy dependence of PSF, and the recovering of the spectral information of point sources.

Section 2 proposes a multichannel representation of spherical Poisson data based on the MS-VSTS and a 2D-1D spherical wavelet transform. Section 3 applies the multichannel MS-VSTS to Poisson denoising. Section 4 applies the multichannel MS-VSTS to Poisson deconvolution.

Experiments are based on simulated Fermi HEALPix cubes ($nside = 256$) with 14 energy bands between 50 MeV and 50 GeV.

17.2 Sparse Representation for Multichannel Spherical Data with Poisson Noise

17.2.1 Fast Undecimated 2D-1D Wavelet Decomposition/Reconstruction on the Sphere

We propose a denoising method for 2D - 1D data on the sphere, where the two first dimensions are spatial (longitude and latitude) and the third dimension is either the time or the energy. We need to analyze the data with a non-isotropic wavelet, where the time or energy scale is not connected to the spatial scale. We use a spherical extension of the Fast Undecimated 2D-1D Wavelet Transform proposed in Starck et al. (2009a).

For a given data set $D(k_\theta, k_\varphi, k_t)$ (k_θ and k_φ are spatial index and k_z a time (or energy)

index), the 2D-1D decomposition consists in applying first a IUWT on the sphere for each frame k_t. Using the spherical IUWT, we have the reconstruction formula:

$$D[k_\theta, k_\varphi, k_t] = a_{J_1}[k_\theta, k_\varphi] + \sum_{j_1=1}^{J1} w_{j_1}[k_\theta, k_\varphi, k_t], \forall k_t \tag{17.1}$$

where J_1 is the number of spatial scales. To have lighter notations, we replace the two spatial indexes by a single index k_r which corresponds to the pixel index:

$$D[k_r, k_t] = a_{J_1}[k_r] + \sum_{j_1=1}^{J1} w_{j_1}[k_r, k_t], \forall k_t \tag{17.2}$$

Then, for each spatial location k_r and for each 2D wavelet scale j_1, we apply a 1D wavelet transform along t on the spatial wavelet coefficients $w_{j_1}[k_r, k_t]$ such that

$$w_{j_1}[k_r, k_t] = w_{j_1,J_2}[k_r, k_t] + \sum_{j_2=1}^{J_2} w_{j_1,j_2}[k_r, k_t], \forall (k_r, k_t) \tag{17.3}$$

where j_2 is the number of scales along t.The same processing is also applied on the coarse spatial scale $a_{J_1}[k_r, k_t]$ and we have

$$a_{J_1}[k_r, k_t] = a_{J_1,J_2}[k_r, k_t] + \sum_{j_2=1}^{J_2} w_{J_1,j_2}[k_r, k_t], \forall (k_r, k_t) \tag{17.4}$$

Hence, we have a 2D-1D spherical undecimated wavelet representation of the input data D:

$$D[k_r, k_t] = a_{J_1,J_2}[k_r, k_t] + \sum_{j_1=1}^{J_1} w_{j_1,J_2}[k_r, k_t] + \sum_{j_2=1}^{J_2} w_{J_1,j_2}[k_r, k_t] + \sum_{j_1=1}^{J_1} \sum_{j_2=1}^{J_2} w_{j_1,j_2}[k_r, k_t] \tag{17.5}$$

From this expression, we distinguish four kinds of coefficients:

- Detail-Detail coefficients ($j_1 \leqslant J_1$ and $j_2 \leqslant J_2$):

$$w_{j_1,j_2}[k_r, k_t] = (\delta - \bar{h}_{1D}) \star (\bar{h}_{1D}^{(j_2-1)} \star a_{j_1-1}[k_r, \cdot] - h_{1D}^{(j_2-1)} \star a_{j_1}[k_r, \cdot]) \tag{17.6}$$

- Approximation-Detail coefficients ($j_1 = J_1$ and $j_2 \leqslant J_2$):

$$w_{J_1,j_2}[k_r, k_t] = h_{1D}^{(j_2-1)} \star a_{J_1}[k_r, \cdot] - h_{1D}^{(j_2)} \star a_{J_1}[k_r, \cdot] \tag{17.7}$$

- Detail-Approximation coefficients ($j_1 \leqslant J_1$ and $j_2 = J_2$):

$$w_{j_1,J_2}[k_r, k_t] = h_{1D}^{(J_2)} \star a_{j_1-1}[k_r, \cdot] - h_{1D}^{(J_2)} \star a_{j_1}[k_r, \cdot] \tag{17.8}$$

- Approximation-Approximation coefficients ($j_1 = J_1$ and $j_2 = J_2$):

$$a_{J_1,J_2}[k_r, k_t] = h_{1D}^{(J_2)} \star a_{J_1}[k_r, \cdot] \tag{17.9}$$

17.2.2 Poisson Noise

The Multi-Scale Variance Stabilizing Transform on the Sphere (MS-VSTS)

Schmitt et al. (2010) proposed a multiscale decomposition on the sphere adapted for spherical data with Poisson noise, called Multi-Scale Variance Stabilizing Transform on the Sphere (MS-VSTS) (see previous chapters). This method consists in mixing a multiscale transform (for instance the Isotropic Undecimated Wavelet Transform on the Sphere) with a variance stabilizing transform (VST), in order to have "Gaussianized" multiscale coefficients.

The recursive scheme of the MS-VSTS with IUWT is: This section describes the MS-VSTS + IUWT, which is a combination of a square-root VST with the IUWT. The recursive scheme is:

$$\begin{array}{l} \text{IUWT} \left\{ \begin{array}{rcl} a_j & = & h_{j-1} * a_{j-1} \\ d_j & = & a_{j-1} - a_j \end{array} \right. \\ \Longrightarrow \begin{array}{l} \text{MS-VSTS} \\ \text{+ IUWT} \end{array} \left\{ \begin{array}{rcl} a_j & = & h_{j-1} * a_{j-1} \\ d_j & = & T_{j-1}(a_{j-1}) - T_j(a_j) \end{array} \right. . \end{array} \tag{17.10}$$

where T_j is the VST operator at scale j:

$$T_j(a_j) = b^{(j)} \text{sign}(a_j + c^{(j)}) \sqrt{|a_j + c^{(j)}|}. \tag{17.11}$$

It has been shown that the detail coefficients d_j issued from locally homogeneous parts of the signal follow asymptotically a central normal distribution with an intensity-independant variance which relies solely on the filter h and the current scale for a given filter h. Consequently, the stabilized variances and the constants $b^{(j)}$ and $c^{(j)}$ can all be pre-computed Schmitt et al. (2010).

Multichannel MS-VSTS

We propose a multichannel extension of the MS-VSTS method. We plug the VST into the spherical 2D-1D undecimated wavelet transform. Again, we distinguish four kinds of coefficients that take the following forms:

- Detail-Detail coefficients ($j_1 \leqslant J_1$ and $j_2 \leqslant J_2$):

$$w_{j_1,j_2}[k_r, k_t] = (\delta - \bar{h}_{1D}) \star (T_{j_1-1,j_2-1}[\bar{h}_{1D}^{(j_2-1)} \star a_{j_1-1}[k_r, \cdot]] - T_{j_1,j_2-1}[h_{1D}^{(j_2-1)} \star a_{j_1}[k_r, \cdot]]) \tag{17.12}$$

- Approximation-Detail coefficients ($j_1 = J_1$ and $j_2 \leqslant J_2$):

$$w_{J_1,j_2}[k_r, k_t] = T_{J_1,j_2-1}[h_{1D}^{(j_2-1)} \star a_{J_1}[k_r, \cdot]] - T_{J_1,j_2}[h_{1D}^{(j_2)} \star a_{J_1}[k_r, \cdot]] \tag{17.13}$$

- Detail-Approximation coefficients ($j_1 \leqslant J_1$ and $j_2 = J_2$):

$$w_{j_1,J_2}[k_r, k_t] = T_{j_1-1,J_2}[h_{1D}^{(J_2)} \star a_{j_1-1}[k_r, \cdot]] - T_{j_1,J_2}[h_{1D}^{(J_2)} \star a_{j_1}[k_r, \cdot]] \tag{17.14}$$

- Approximation-Approximation coefficients ($j_1 = J_1$ and $j_2 = J_2$):

$$a_{J_1,J_2}[k_r, k_t] = h_{1D}^{(J_2)} \star a_{J_1}[k_r, \cdot] \tag{17.15}$$

Hence, all 2D-1D wavelet coefficients w_{j_1,j_2} are now stabilized, and the noise on all these wavelet coefficients is Gaussian with known scale-dependent variance that depends solely on h.

17.3 Application to Multichannel Denoising

17.3.1 Gaussian Case

As the spherical 2D-1D undecimated wavelet transform described before is fully linear, a Gaussian noise remains Gaussian after transformation. Therefore, all thresholding strategies which have been developed for wavelet Gaussian denoising are still valid with the spherical 2D-1D wavelet transform. Denoting TH the thresholding operator, the denoised cube in the case of additive white Gaussian noise is obtained by:

$$\tilde{D}[k_r, k_t] = a_{J_1,J_2}[k_r, k_t] + \sum_{j_1=1}^{J_1} \mathrm{TH}(w_{j_1,J_2}[k_r, k_t]) + \sum_{j_2=1}^{J_2} \mathrm{TH}(w_{J_1,j_2}[k_r, k_t]) + \sum_{j_1=1}^{J_1} \sum_{j_2=1}^{J_2} \mathrm{TH}(w_{j_1,j_2}[k_r, k_t]) \tag{17.16}$$

A typical choice of TH is the hard thresholding operator, i.e.:

$$\mathrm{TH}(x) = \begin{cases} 0 & if |x| < \tau \\ x & if |x| \geqslant \tau \end{cases} \tag{17.17}$$

The threshold τ is generally chosen between 3 and 5 times the noise standard deviation.

17.3.2 Poisson Case

We perform a MS-VSTS transform on the data. As the noise on the stabilized coefficients is Gaussian, and without loss of generality, we let its standard deviation equal to 1, we consider that a wavelet coefficient $w_{j_1,j_2}[k_r, k_t]$ is significant, i.e., not due to noise, if its absolute value is larger than a critical threshold τ, where τ is typically between 3 and 5. Denoising is however not straightforward because there is no explicit reconstruction formula available because of the form of the stabilization equations above. Formally, the stabilizing operators T_{j_1,j_2} and the convolution operators along the spatial and temporal dimensions do not commute, even though the filter bank satisfies the exact reconstruction formula. To circumvent this difficulty, we propose to solve this reconstruction problem by using an iterative reconstruction scheme.

17.3.3 Iterative Reconstruction

We define a multiresolution support which is obtained by detecting at each scale the significant coefficients. The multiresolution support for $j_1 \leqslant J_1$ and $j_2 \leqslant J_2$ is defined as:

$$\mathcal{M}_{j_1,j_2}[k_r, k_t] = \begin{cases} 1 & \text{if } w_{j_1,j_2}[k_r, k_t] \text{ is significant} \\ 0 & \text{otherwise} \end{cases} \tag{17.18}$$

We denote $\mathcal{W}$ the spherical 2D-1D undecimated wavelet transform described above, and $\mathcal{R}$ the inverse wavelet transform. We want our solution X to preserve the significant structures of the original data by reproducing exactly the same coefficients as the wavelet coefficients of the input data Y, but only at scales and positions where significant signal has been detected. At other scales and positions, we want the smoothest solution with the lowest budget in terms of wavelet coefficients. Furthermore, as Poisson intensity functions are positive by nature, a positivity constraint is imposed on the solution. It is clear that there are many solutions satisfying the positivity and multiresolution support consistency requirements, e.g. Y itself. Thus, our reconstruction problem based solely on these constraints is an ill-posed inverse problem that must be regularized. Typically, the solution in which we are interested must be sparse by involving the lowest budget of wavelet coefficients. Therefore our reconstruction is formulated as a constrained sparsity-promoting minimization problem that can be written as follows

$$\min_{\mathbf{X}} \|\mathcal{W}\mathbf{X}\|_1 \text{ subject to } \begin{cases} \mathcal{M}\mathcal{W}\mathbf{X} = \mathcal{M}\mathcal{W}\mathbf{Y} \\ \mathbf{X} \geqslant 0 \end{cases} \tag{17.19}$$

where $\|\cdot\|$ is the l_1-norm playing the role of regularization and is well known to promote sparsity (Donoho 2004). This problem can be solved efficiently using the hybrid steepest descent algorithm (Yamada 2001; Zhang et al. 2008), and requires about 10 iterations in practice. Transposed into our context, its main steps can be summarized as follows:

Require: Input noisy data $\mathbf{Y}$, a low-pass filter h, multiresolution support $\mathcal{M}$ from the detection step, number of iterations $N_{\max}$

1: Initialize $\mathbf{X}^{(0)} = \mathcal{M}\mathcal{W}\mathbf{Y} = \mathcal{M}w_Y$,
2: **for** $n = 1$ to $N_{\max}$ **do**
3: $\quad \tilde{d} = \mathcal{M}w_Y + (1 - \mathcal{M})\mathcal{W}X^{(n-1)}$,
4: $\quad \mathbf{X}^{(n)} = P_+(\mathcal{R}\mathrm{ST}_{\beta_n}[\tilde{d}])$,
5: $\quad$ Update the step $\beta_n = (N_{\max} - n)/(N_{\max} - 1)$
6: **end for**

where P_+ is the projector onto the positive orthant, i.e. $P_+(x) = max(x, 0)$, ST_{β_n} is the soft-thresholding operator with threshold β_n, i.e. $\mathrm{ST}_{\beta_n}[x] = x - \beta_n \mathrm{sign}(x)$ if $|x| \geqslant \beta_n$, and 0 otherwise.

The final spherical MS-VSTS 2D-1D wavelet denoising algorithm is the following:

Require: Input noisy data $\mathbf{Y}$, a low-pass filter h, threshold level τ

1: Spherical 2D-1D MS-VST: Apply the spherical 2D-1D-MS-VST to the data using (17.12)-(17.15).
2: Detection: Detect the significant wavelet coefficients that are above τ, and compute the multiresolution support $\mathcal{M}$.
3: Reconstruction: Reconstruct the denoised data using the algorithm above.

17.3.4 Experiments

The algorithm has been applied on our simulated Fermi data set, with 7 energy bands between 50 MeV and 1.58 GeV. Figures 17.1 and 17.2 shows the result of the algorithm

on 2 energy bands. The multichannel MS-VSTS provides a performant denoising on each energy band and enables us to get the spectral information for each spatial position (Figure 17.3).

17.4 Multichannel Sparse Deconvolution for Spherical Poisson Data

17.4.1 Introduction

Many problems in signal and image processing can be cast as inverting the linear system:

$$y = \mathbf{H}x + \epsilon \tag{17.20}$$

where $x \in \mathbb{R}^N$ is the data to recover, $y \in \mathbb{R}^m$ is the image of noisy observations, and ϵ is an additive noise with bounded variance. The unknown error can be either a stochastic measurement noise induced by the sensor, or a deterministic perturbation due for example to an imperfect signal model. $\mathbf{H} : \mathbb{R}^N \rightarrow \mathbb{R}^m$ is a bounded linear operator which is typically ill-behaved since it models an acquisition process that encounters loss of information. Eq. (17.20) is usually an ill-posed problem. This means that there is no unique and stable solution.

Our objective is to remove the effect of the instrument's PSF. In our case, $\mathbf{H}$ is the convolution by a blurring kernel (PSF) and y lacks the high frequency content of x and the problem above is a deconvolution problem.

In order to regularize such an inversion problem and reduce the space of candidate solutions, one has to add some prior knowledge on the typical structure of the original image x. This prior information account for the smoothness of the solution and can range from the uniform smoothness assumption to more complex knowledge of the geometrical structures of x.

In our LAT realistic simulations, the point spread function width depends a lot on the energy, from 6.9!‘ at 50 MeV to better than 0.1!‘ at 10 GeV and above. Figure 17.4 shows the normalized profiles of the PSF for different energy bands.

17.4.2 Monochannel Deconvolution

For Poisson image deconvolution, several methods were proposed. Richardson-Lucy is certainly the most famous in astrophysics. In this paper, we propose a new regularized Richardon-Lucy algorithm for spherical Poisson data.

We denote $\mathbf{Y}$ the observed data, $\mathbf{H}$ the convolution kernel, and $\mathcal{N}(\mathbf{X})$ the additive noise. The deconvolution consists in estimating $\mathbf{X}$ so that:

$$\mathbf{Y} = \mathbf{H} \star \mathbf{X} + \mathcal{N}(\mathbf{X}) \tag{17.21}$$

where $\star$ denotes the convolution product.

An iterative deconvolution scheme is given by Richardson-Lucy method. We start with $n = 0$ and $\mathbf{X}^{(0)} = 1$ and we iterate:

$$\mathbf{X}^{(n+1)} = P_+\Big[\mathbf{X}^{(n)}\Big(\mathbf{H}^T * \frac{\mathbf{Y}}{\mathbf{H} * \mathbf{X}^{(n)}}\Big)\Big] \tag{17.22}$$

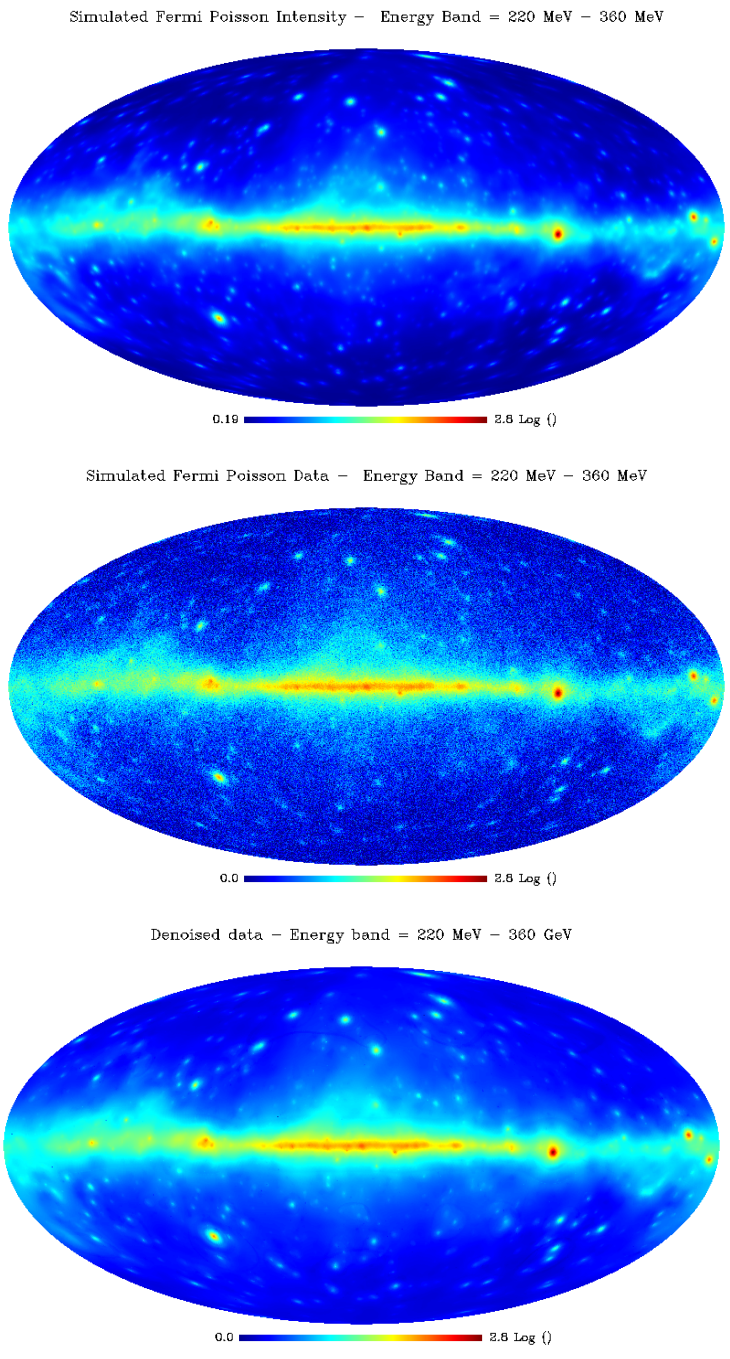


Figure 17.1: Result of the multichannel Poisson denoising algorithm on simulated Fermi data on energy band 220 MeV - 360 MeV (Top: Simulated intensity skymap. Middle: Simulated noisy skymap. Bottom: denoised skymap). Pictures are in logarithmic scale.

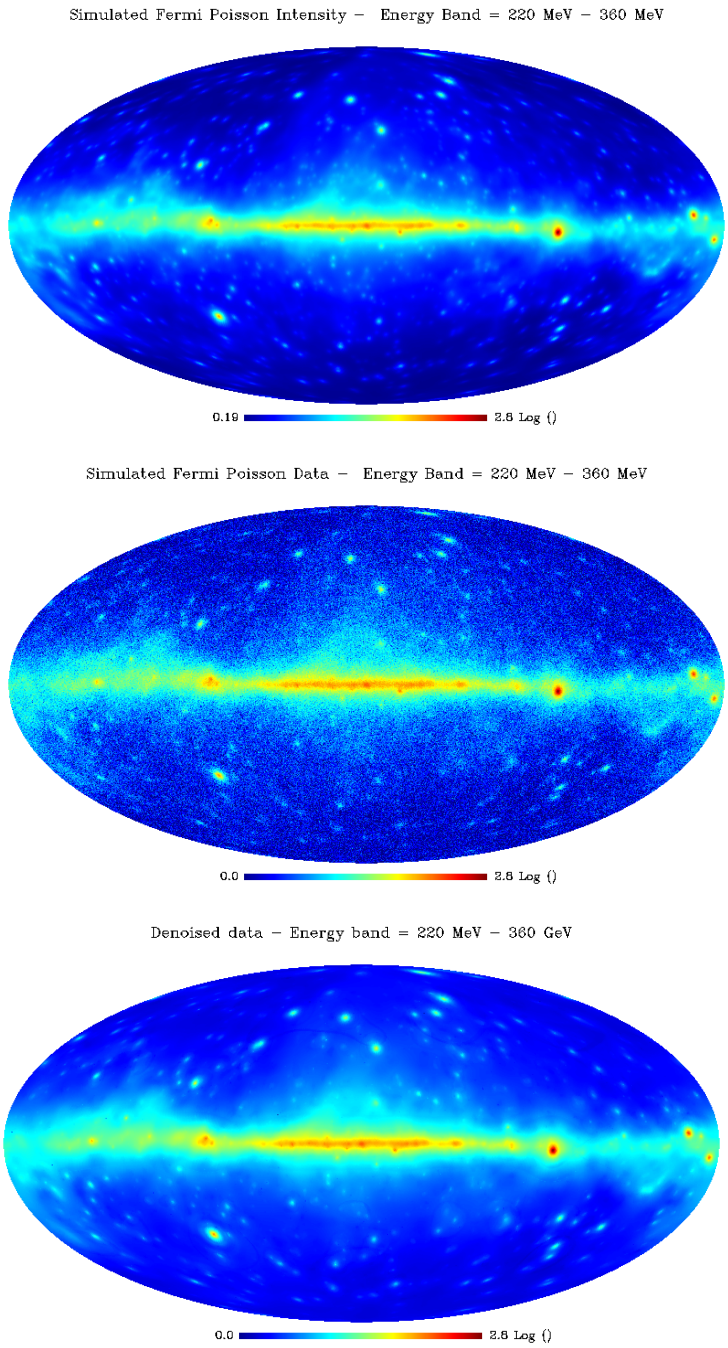


Figure 17.2: Result of the multichannel Poisson denoising algorithm on simulated Fermi data on energy band 589 MeV - 965 MeV (Top: Simulated intensity skymap. Middle: Simulated noisy skymap. Bottom: denoised skymap). Pictures are in logarithmic scale.

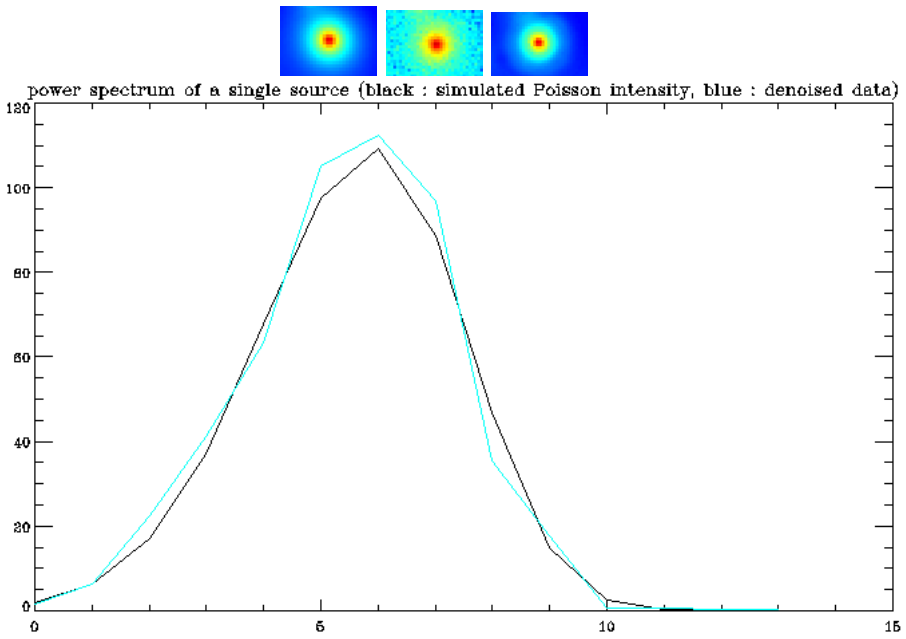


Figure 17.3: Power spectrum of a single gamma ray point source recovered using the multichannel MS-VSTS denoising algorithm. Top: Single gamma ray point source on simulated Fermi data integrated along the energy axis (Left: Poisson Intensity. Middle: Noisy data. Right: Denoised data.) Bottom: Power spectrum of the center of the denoised point source: intensity as a function of the energy band (Black: Simulated Poisson intensity map. Blue: Denoised map.) (14 energy bands between 50 MeV and 50 GeV)

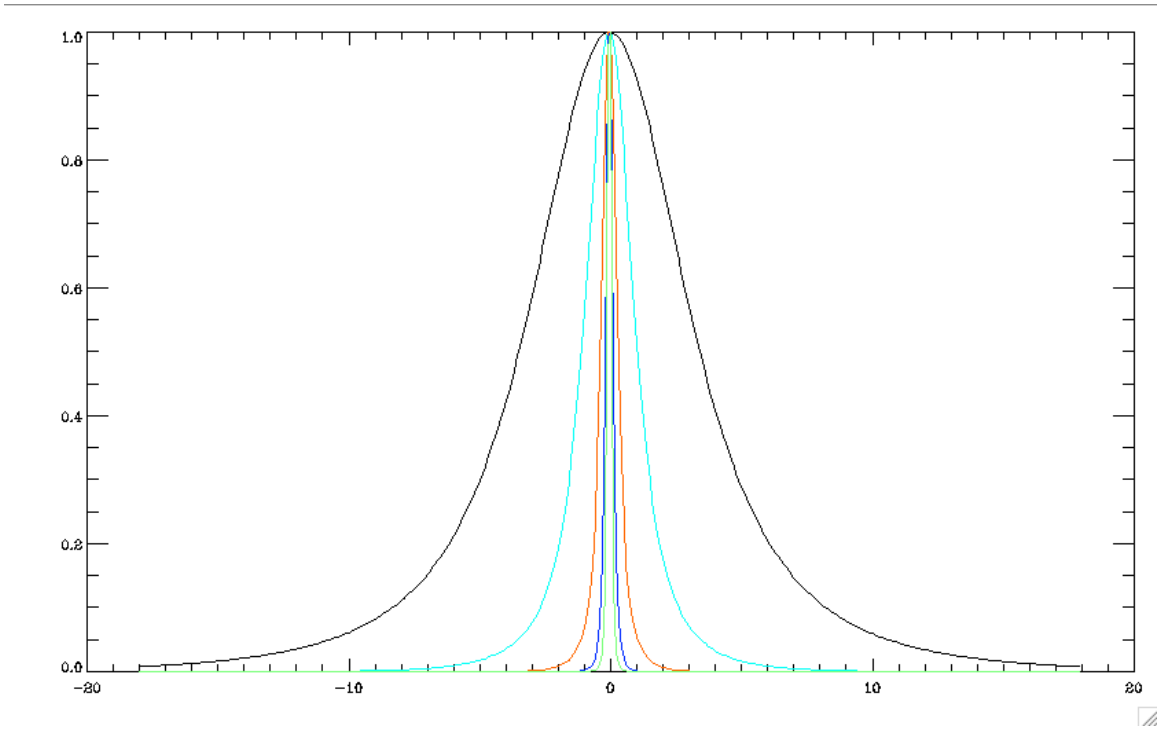


Figure 17.4: Normalized profile of the point spread function for different energy bands as a function of the angle in degree. Black: 50 MeV - 82 MeV. Light Blue: 220 MeV - 360 MeV. Orange: 960 MeV - 1.6 GeV. Dark Blue: 4.2 GeV - 6.9 GeV. Green: 19 GeV - 31 GeV.

where $\mathbf{H}^T$ is the transpose of $\mathbf{H}$, and P_+ a positivity projection.

We define $\mathbf{R}^{(n)}$ the residual at iteration n:

$$\mathbf{R}^{(n)} = \mathbf{Y} - (\mathbf{H} * \mathbf{X}^{(n)}) \tag{17.23}$$

By using the Isotropic Undecimated Wavelet Transform on the sphere (IUWT), $\mathbf{R}^{(n)}$ can be defined as the sum of its J wavelet scales and the last smooth array, for a pixel k:

$$\mathbf{R}^{(n)}[k] = a_J[k] + \sum_{j=1}^{J} d_j[k] \tag{17.24}$$

where a_J denotes the last smoothed array, and d_j denotes a wavelet scale.

The wavelet coefficients provide a mechanism to extract only the significant structures from the residual at each iteration. Normally, a large part of these residuals are statistically non-significant. The significant residual is then, for a pixel k:

$$\bar{\mathbf{R}}^{(n)}[k] = a_J[k] + \sum_{j=1}^{J} \mathcal{M}(j,k) d_j[k] \tag{17.25}$$

where $\mathcal{M}(j,k)$ is the multiresolution support. The regularized iterative scheme becomes:

$$\mathbf{X}^{(n+1)} = P_+\Big[\mathbf{X}^{(n)}\Big(\mathbf{H}^T * \frac{\mathbf{H} * \mathbf{X}^{(n)} + \bar{\mathbf{R}}^{(n)}}{\mathbf{H} * \mathbf{X}^{(n)}}\Big)\Big] \tag{17.26}$$

17.4.3 Multichannel Deconvolution

In this problem, the data can be viewed as a matrix $\mathbf{Y} = (Y_t)_t$, where, for each channel t, $Y_t = (y_{r,t})_r$ is the vector corresponding to the spherical data at channel t, where r is the index corresponding to the pixel. Each channel is convolved by a known blurring kernel $\mathbf{H}_t$: $Y_t = \mathbf{H}_t \star X_t + \mathcal{N}(X_t)$.

We compute the spherical 2D-1D MS-VSTS transform, and the multiresolution support $\mathcal{M}_{j_1,j_2}[k_r,k_t]$ is obtained by hypothesis testings on coefficients.

We denote $\mathcal{H}$ the convolution by channel operator: $\mathcal{H}\mathbf{X}$ means that each channel X_t is convolved by the convolution kernel $\mathbf{H}_t$. $\mathcal{H}^T\mathbf{X}$ means that each channel X_t is convolved by the transposed convolution kernel $\mathbf{H}_t^T$. The regularized iterative scheme is then:

$$\mathbf{X}^{(n+1)} = P_+\Big[\mathbf{X}^{(n)}\Big(\mathcal{H}^T \frac{\mathcal{H}\mathbf{X}^{(n)} + \bar{\mathbf{R}}^{(n)}}{\mathcal{H}\mathbf{X}^{(n)}}\Big)\Big] \tag{17.27}$$

with $\bar{\mathbf{R}}^{(n)}[k_r,k_t]$ the significant residual:

$$\begin{aligned}\bar{\mathbf{R}}^{(n)}[k_r,k_t] = a_{J_1,J_2}[k_r,k_t] + \sum_{j_1=1}^{J_1} \mathcal{M}_{j_1,J_2}[k_r,k_t] w_{j_1,J_2}[k_r,k_t] \\ + \sum_{j_2=1}^{J_2} \mathcal{M}_{J_1,j_2}[k_r,k_t] w_{J_1,j_2}[k_r,k_t] + \sum_{j_1=1}^{J_1}\sum_{j_2=1}^{J_2} \mathcal{M}_{j_1,j_2}[k_r,k_t] w_{j_1,j_2}[k_r,k_t]\end{aligned} \tag{17.28}$$

where $\mathcal{M}_{j_1,j_2}$ is the multiresolution support.

17.4.4 Experiments

The algorithm is applied on the 7 energy bands of our simulated Fermi data set ($50MeV$ to $1.58GeV$). Figures 17.5 to 17.8 show the result of the deconvolution on 4 energy bands. Figure 17.9 shows the effect of the MS-VSTS deconvolution algorithm on a single point source. The deconvolution gives a better spatial localisation for point sources. Figure 17.10 shows the effect of the MS-VSTS deconvolution algorithm on the galactic plan, where the effect of the deconvolution is particularly spectacular. Our MS-VSTS multichannel deconvolution algorithm manages to remove a large part of the PSF effect.

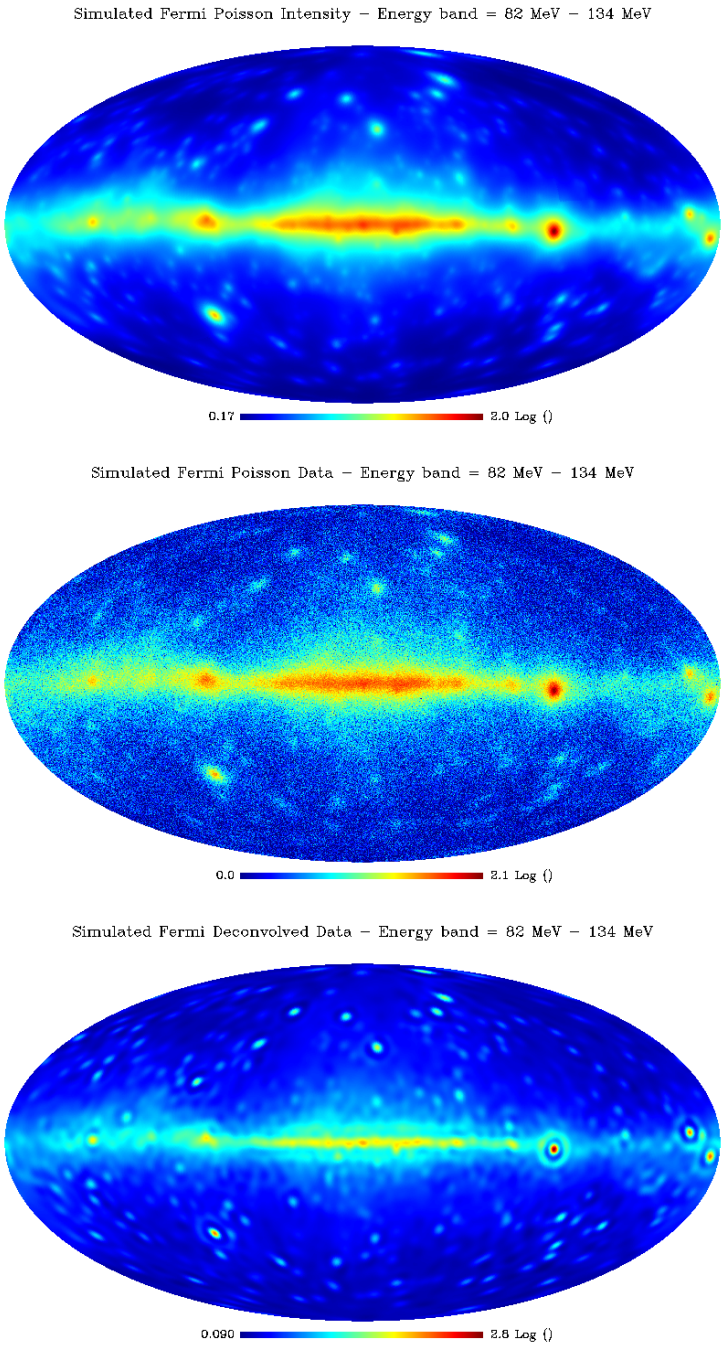


Figure 17.5: Result of the multichannel deconvolution algorithm on different energy bands (Top: Simulated Intensity skymap. Middle: noisy skymap. Bottom: deconvolved skymap). Energy band : 82 MeV - 134 MeV. Pictures are in logarithmic scale.

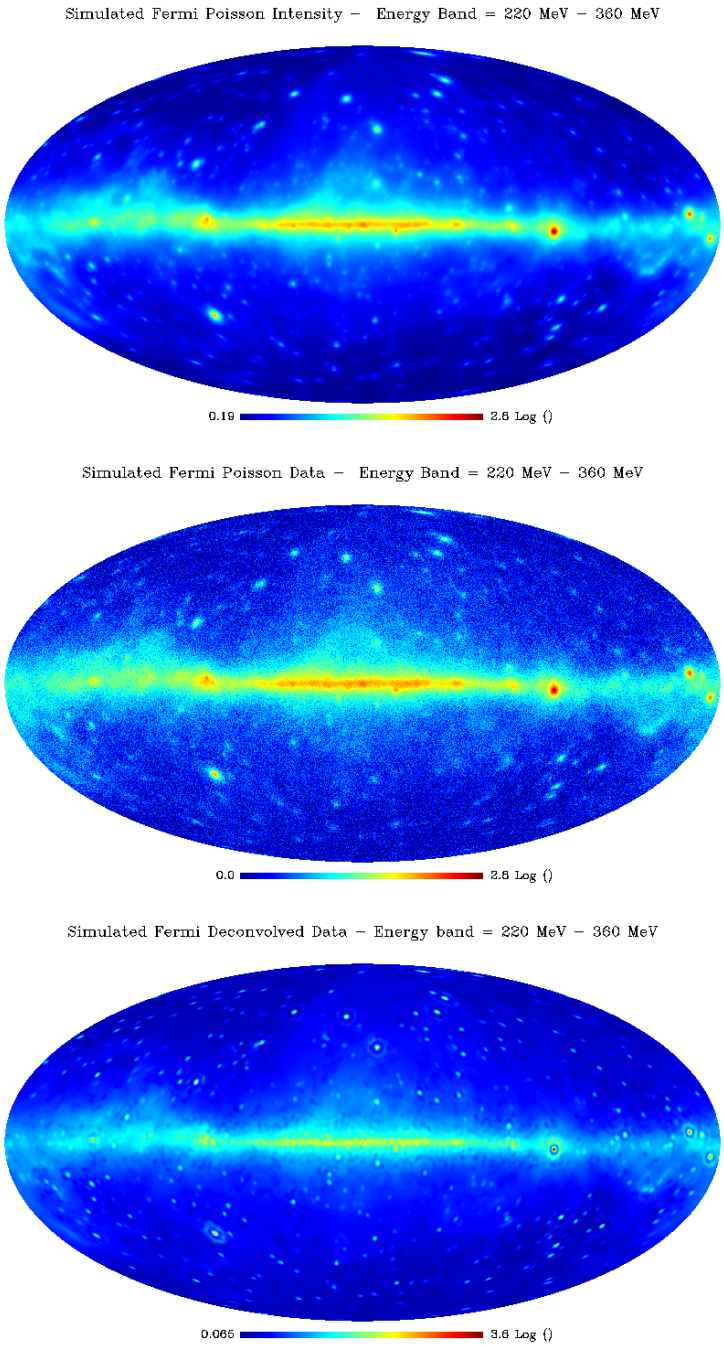


Figure 17.6: Result of the multichannel deconvolution algorithm on different energy bands (Top: Simulated Intensity skymap. Middle: noisy skymap. Bottom: deconvolved skymap). Energy band : 220 MeV - 360 MeV. Pictures are in logarithmic scale.

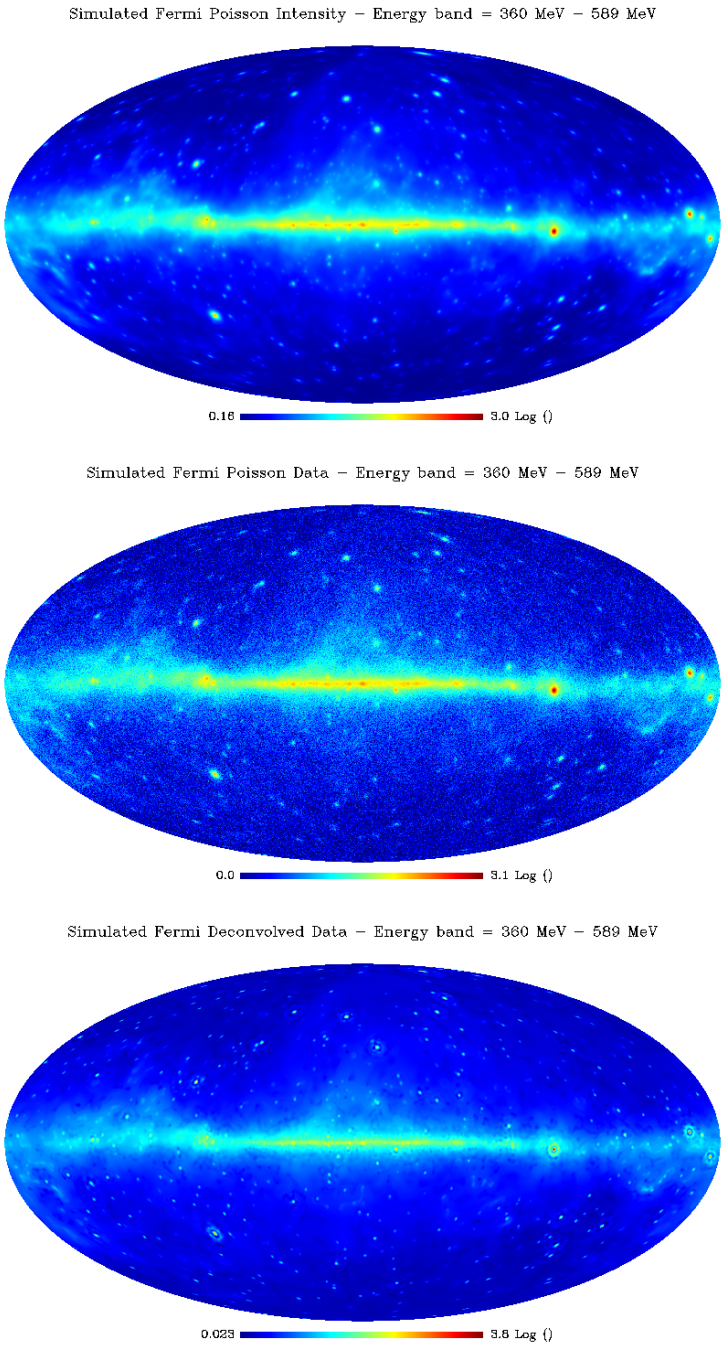


Figure 17.7: Result of the multichannel deconvolution algorithm on different energy bands (Top: Simulated Intensity skymap. Middle: noisy skymap. Bottom: deconvolved skymap). Energy band : 360 MeV - 589 MeV. Pictures are in logarithmic scale.

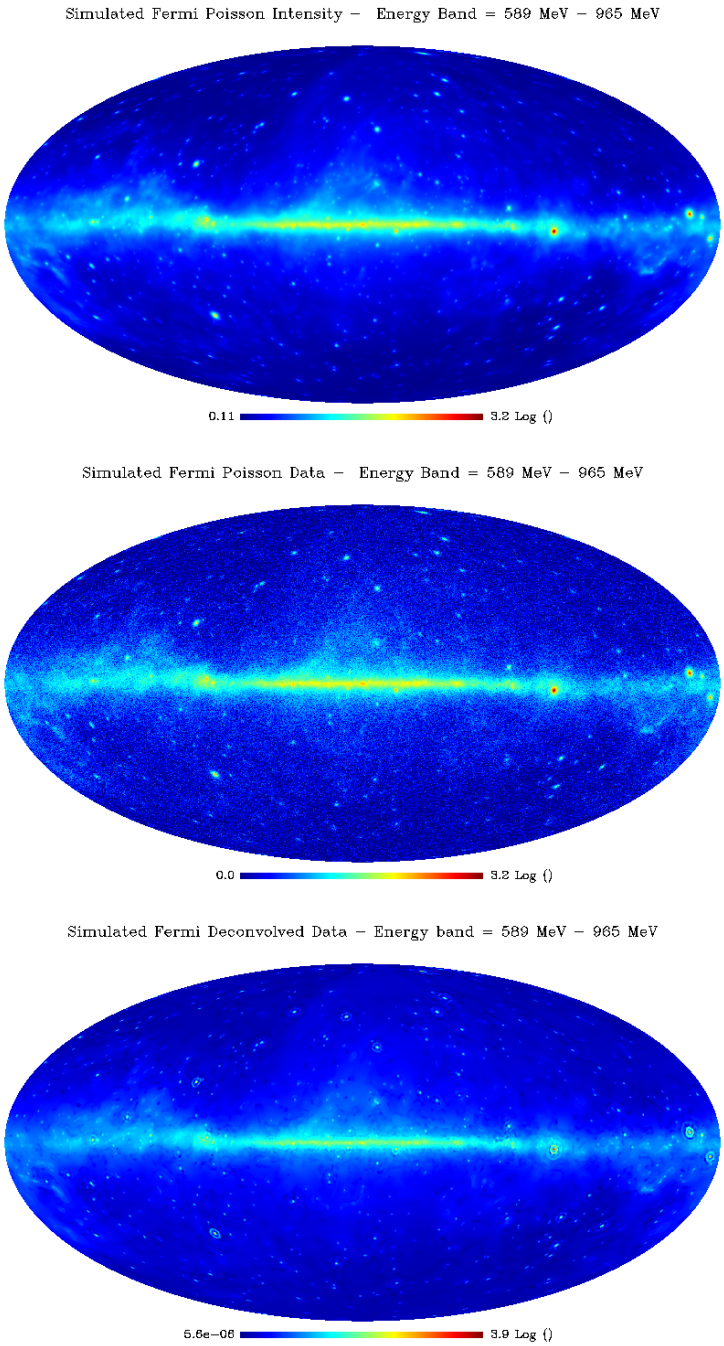


Figure 17.8: Result of the multichannel deconvolution algorithm on different energy bands (Top: Simulated Intensity skymap. Middle: noisy skymap. Bottom: deconvolved skymap). Energy band : 589 MeV - 965 MeV. Pictures are in logarithmic scale.

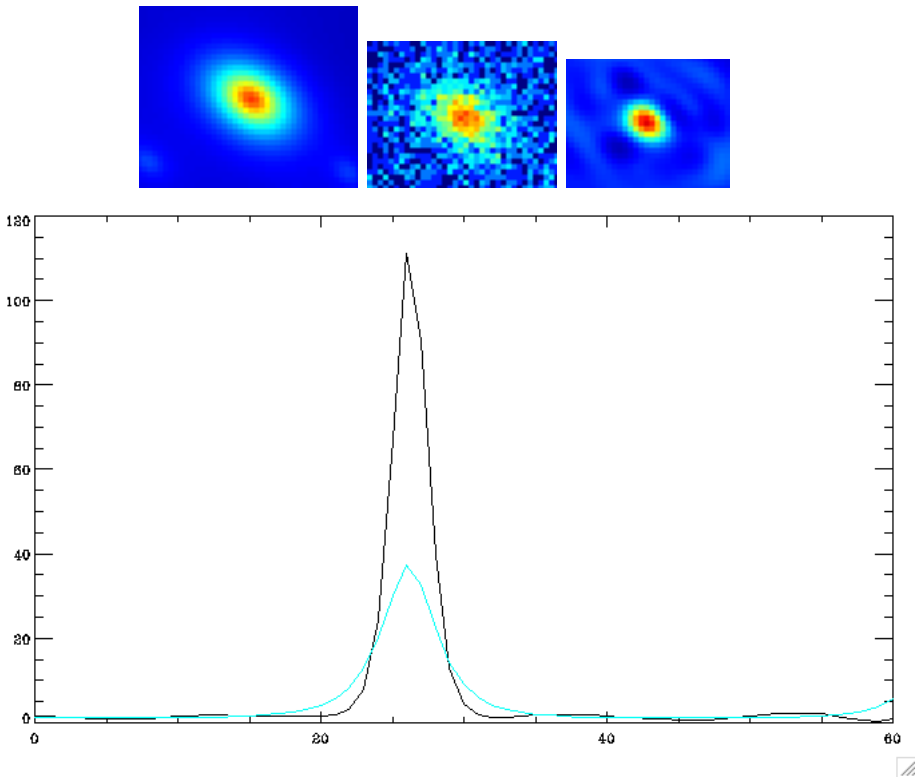


Figure 17.9: Effect of the multichannel deconvolution algorithm on a single gamma ray point source. Top: Single gamma ray point source on simulated Fermi data (Energy band: 360 MeV - 589 MeV) (Left: Intensity. Middle: Noisy data. Right: Deconvolved data.) Bottom: Profile of the point source. In blue, the source from intensity map. In black, the deconvolved source.

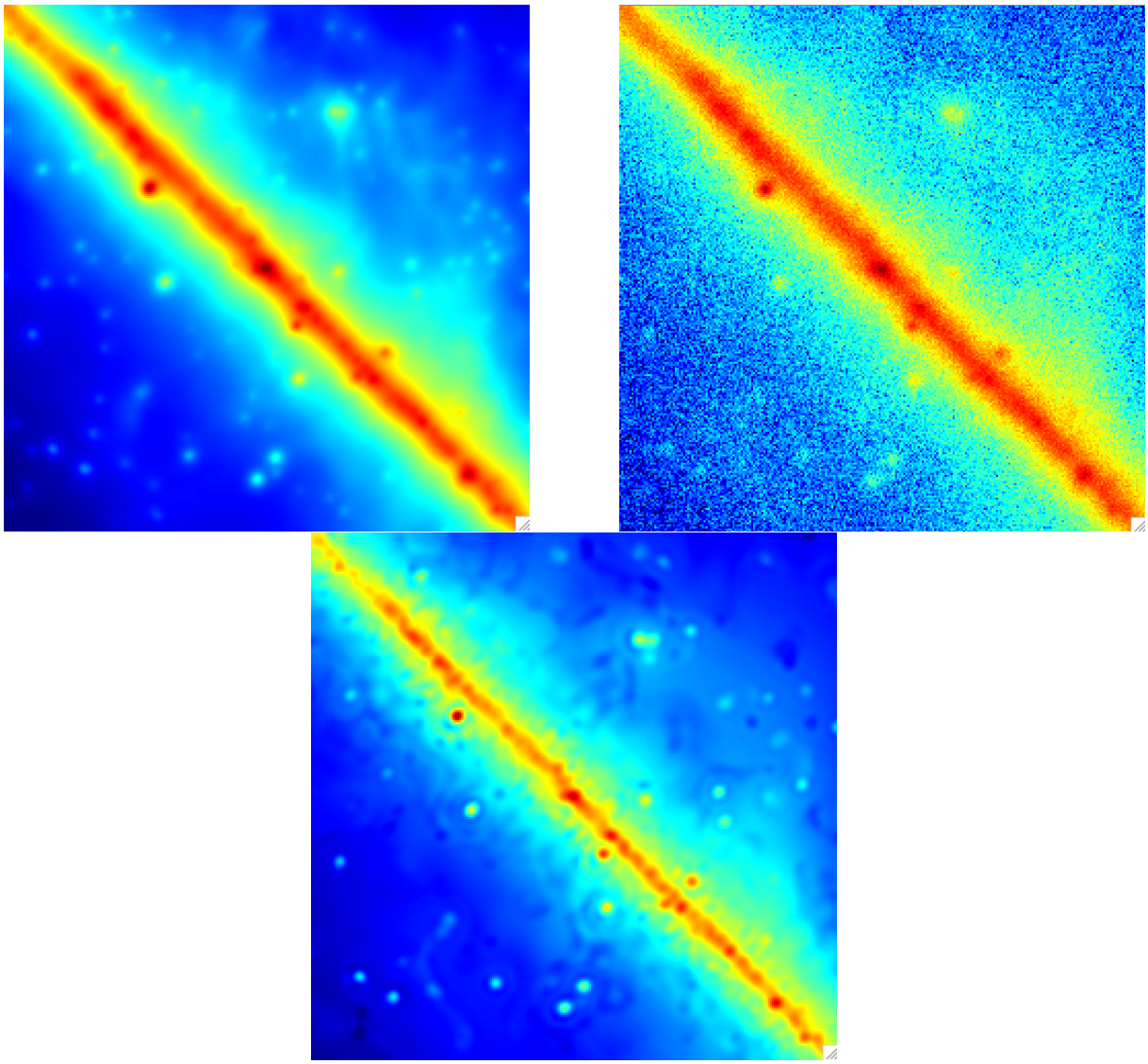


Figure 17.10: View on a single HEALPix face. Result of the deconvolution algorithm on the galactic plan. Top Left: Simulated Fermi Poisson intensity. Top Right: Simulated Fermi noisy data. Bottom: Fermi data deconvolved with multichannel MS-VSTS. Energy band: 360 MeV - 589 MeV. Pictures are in logarithmic scale.

Chapter 18

IDL Routines

18.1 Denoising using MS-VSTS + Isotropic Undecimated Wavelet Transform

18.1.1 Main routine

▶ Starting from a photon counts map

↬ **mrs_msvsts_IUWT_denoising.pro** : Compute Poisson denoising on spherical HEALPix data with MS-VSTS + Isotropic Undecimated Wavelet Transform method.

USAGE : mrs_msvsts_IUWT_denoising, image, image_reconstruite, NbrScale=NbrScale, niter=niter, HSD=HSD, coef_seuil=coef_seuil, coef_pos=coef_pos, First_Scale=First_Scale, mask=mask, filter=filter, pyr=pyr, background=background, expo=expo, alm=alm, curv=curv, separation=separation, back_reconstruit=back_reconstruit, update_support=update_support, split_support=split_support

INPUTS :

- Image = (IDL array) HEALPix data to be denoised
- (Optional) background = (IDL array) if set, substracts a background to the data
- (Optional) support = (IDL array) if set, use a given multi-resolution support instead of computing it with the procedure mrs_msvsts_hypothesis_testing

OUTPUTS :

- Image_reconstruite = (IDL array) HEALPix denoised image
- (optional) Support = (IDL array) multi-resolution support of the image
- (optional) Back_reconstruit = (IDL array) if set, returns the reconstructed background (need the keyword separation)

KEYWORD

- NbrScale : Number of scales (default is 4)

- niter : Number of iterations
- HSD : if set, the denoised image will be recontructed using the Hybrid Steepest Descent Method (soft thresholding at each iteration of the reconstruction)
- coef_seuil : determines the threshold for the detection of significant coefficients. For each scale i, the threshold is set to coef_seuil $* \sigma_i$ (default is 5)
- coef_pos : if set, negative wavelets coefficients are set to 0.
- First_Scale : if > 2, finer wavelet scales are set to 0. (default is 1)
- mask : if set, enables impainting with the given mask
- filter : if set, the inverse wavelet transform will be computed using filters. Else, it will be obtained by a simple addition of all wavelet scales.
- pyr : if set, use pyramidal wavelet transform for the soft thresholding
- expo : if set, decreases the thresold exponentially at each step of the HSD. Else, decreases the threshold linearly
- alm : if set, thresholding is made on alm coefficients instead of wavelet coefficients
- curvelets : if set, thresholding is made on curvelets coefficients instead of wavelet coefficients
- separation : if set, compute separately the sources and the background
- update_support : if set, update the multi-resoluation support at each iteration
- split_support : if set, splits the multi-resolution support

18.1.2 Subroutines

↬ **mrs_msvsts_IUWT_param_computing.pro** : For a given number of scales, determines the VST operator at each scale for the MS-VST transform with spherical isotropic Undecimated Wavelet Transform. At scale j, the VST operator is: $T_j(a_j) = b_{(j)} * sgn(a_j + c_{(j)}) * \sqrt{(|a_j + c_{(j)}|)}$ where a_j is the jth scale coefficient of the wavelet transform.

USAGE :
mrs_msvsts_IUWT_param_computing,nbr,c,b,h,tau1,tau2,tau3,sigma

INPUTS :

- nbr = (int) number of scales for the MS-VST transform

OUTPUTS :

- c = (1D IDL array) vector of the c(j) coefficients for each scale j

- b = (1D IDL array) vector of the b(j) coefficients for each scale j
- h = (IDL array) h[*,j] is the low pass filter which gives the jth scale from the original image
- tau1 = (1D IDL array) vector of the 1st order moments of h[*,j] for each scale j
- tau2 = (1D IDL array) vector of the 2st order moments of h[*,j] for each scale j
- tau3 = (1D IDL array) vector of the 3rd order moments of h[*,j] for each scale j
- sigma = (1D IDL array) vector of the asymptotic standard deviations of detail coefficients issued from locally homogeneous parts of a signal for each wavelet scale

↬ **mrs_msvsts_IUWT_transform.pro** : Computes the multi-scale variance stabilising transform on the sphere with undecimated isotropic wavelet transform, using the HEALPix representation (nested data representation). The wavelet function is zonal and its spherical harmonics coefficients a_{l0} follow a cubic box-spline profile. If DifInSH is set, wavelet coefficients are derived in the Spherical Harmonic Space, otherwise (default) they are derived in the direct space.

USAGE : mrs_msvsts_IUWT_transform, Imag, Trans, NbrScale=NbrScale, lmax=lmax, DifInSH=DifInSH

INPUTS :

- Imag = (IDL array) HEALPix data

OUTPUTS :

- Trans = IDL structure with the following fields:
  - NbrScale = (int) number of scales
  - nside = (int) Healpix nside parameter
  - lmax = (int) Maximum l value in the Spherical Harmonic Space (Healpix)
  - npix = (int) Number of pixels of the input image (12*nside*nside)
  - Coef = (IDL array) stabilised wavelet transform of the data
  - Coef[*,0] = stabilised wavelet coefficients of the finest scale (highest frequencies).
  - Coef[*,NbrScale-1] = coarsest scale (lowest frequencies).
  - lmax = (int) lmax parameter at the first scale

KEYWORDS :

- NbrScale = (int) Number of scales (default is 4)
- Lmax = (int) Number of spherical harmonics computed in the decomposition (default is 3*nside, should be between 2*nside and 4*nside)

- DifInSH : If set, compute the wavelet coefficients as the difference between two resolution in the spherical harmonics representation. Otherwise, the wavelet coefficients are computed as the difference between two resolutions in the initial representation.

↬ **mrs_msvsts_IUWT_hypothesis_testing.pro** : Computes the MS-VSTS + Isotropic Undecimate Wavelet Transform of a Poisson Image, perform hypothesis testing on coefficients, returns the multi-resolution support and the denoised image using direct reconstruction

USAGE : mrs_msvsts_IUWT_hypothesis_testing, image, image_vst, support, image_rec, NbrScale=NbrScale, coef_seuil=coef_seuil, First_Scale=First_Scale, background=background

INPUTS :

- Imag = (IDL array) HEALPix data
- (optional) background = (IDL array) if set, substracts a background to the image.

OUTPUTS :

- image_vst = (IDL structure) MS-VSTS transform of the image computed with mrs_msvsts_IUWT_transform
- support = (IDL array) multi-resolution support
- image_rec = (IDL array) directly reconstructed denoised image

KEYWORDS :

- NbrScale = (int) Number of scales (default is 4)
- coef_seuil = (int) determines the threshold for the detection of significant coefficients. For each scale i, the threshold is set to coef_seuil $* \sigma_i$ (default is 5)
- First_Scale = (int) if > 2, finer wavelet scales are set to 0. (default is 1)

18.2 Denoising using MS-VSTS + Curvelet Transform

18.2.1 Main routine

↬ **mrs_msvsts_curv_denoising.pro** : Compute Poisson denoising on spherical HEALPix data with MS-VSTS + Curvelet Transform method.

USAGE : mrs_msvsts_curv_denoising, image, image_reconstruite, support, nbrscale=nbrscale, coef_seuil=coef_seuil, suppr_scale1=suppr_scale1, hsd=hsd, niter=niter

INPUTS :

- Image = (IDL array) HEALPix data to be denoised

OUTPUTS :

- Image_reconstruite = (IDL array) HEALPix denoised image
- (optional) Support = (IDL array) multi-resolution support of the image

KEYWORD :

- NbrScale = Number of scales (default is 4)
- HSD = if set, the denoised estimate will be recontructed using the Hybrid Steepest Descent Method (soft thresholding at each iteration of the reconstruction). If not set, the estimate is direclty reconstructed.
- niter = Number of iterations
- coef_seuil = determines the threshold for the detection of significant coefficients. For each scale i, the threshold is set to coef_seuil $* \sigma_i$ (default is 5)
- coef_pos = if set, negative wavelets coefficients are set to 0.
- suppr_scale1 = if set, remove the finest scale from the reconstructed estimate.

18.2.2 Subroutines

↬ **mrs_msvsts_curv_transform.pro** : Compute the multi-scale variance stabilizing transform on the sphere with standard undecimated curvelet transform on the sphere, using the healPix pixel representation (nested data representation). A band of the curvelet transform is defined by two number, the 2D WT scale number and the ridgelet scale number. The output is a IDL structure.

USAGE : mrs_msvsts_curv_transform, Imag, Trans, lmax=lmax, NbrScale=NbrScale, FirstBlockSize=FirstBlockSize

INPUTS :

- Image = (IDL array) HEALPix data to be transformed

OUTPUTS :

- Trans = IDL structures with the following fields:
  - NBRSCALE = (INT) Nbr of the scale in the 2D WT
  - TABBLOCKSIZE = (INT) TABBLOCKSIZE[j], Block size in the ridgelet transform at scale j. $j = [0..NBRSCALE - 2]$
  - TABNBRSCALERID = (INT) TABNBRSCALERID[j], number of ridgelet band at scale j
  - TABNORM = (2D IDL ARRAY) Normalization array
  - RIDSCALE1 = (IDL STRUCT) ridgelet transform of the first wavelet scale (see mrs_ridtrans.pro for details)

- RIDSCALEj = (IDL STRUCT) ridgelet transform of the jth wavelet scale. $j = [0..NBRSCALE - 2]$
- LASTSCALE = (IDL 1D array) Healpix image of the coarsest scale
- WT = (IDL STRUCT) Wavelet structure (for internal use only)
- PYRTRANS = (INT) equal to 1 for a pyramidal curvelet transform and 0 otherwise

KEYWORD :

- NbrScale = (INT) Number of scale in the 2D wavelet transform (defaut 4)
- Undec = (INT) if set, an undecimated curvelet transform is used instead of the pyramidal curvelet transform
- FirstBlockSize = (INT) Block size in the ridgelet transform at the finest scale (default is 16)
- Lmax = (INT) Number of used spherical harmoniques used in the wavelet transform (defaut = 3*nside, should be between 2*nside and 4*nside)
- Overlap = (LONG) is equal to 1 if blocks are overlapping

18.3 Multichannel Denoising using MS-VSTS + Multichannel Wavelet Transform

▶ Starting from a set of photon counts maps

↬ **mrs_msvsts_multichannel_denoising.pro** : Compute multichannel Poisson denoising on spherical 2D-1D HEALPix data with MS-VSTS + multichannel Wavelet Transform method.

USAGE :
mrs_msvsts_multichannel_denoising,input,solution,NbrScale1=NbrScale1,NbrScale2=NbrSc

INPUT :

- Input = (IDL array) multichannel HEALPix data to be denoised

OUTPUT :

- Solution = (IDL array) multichannel HEALPix denoised image

KEYWORD

- NbrScale1 : Number of scales for the two spatial dimensions (default is 6)
- NbrScale2 : Number of scales for the non-spatial dimension (time or energy) (default is 6)
- niter : Number of iterations

18.4 Multichannel Deconvolution using MS-VSTS + Multichannel Wavelet Transform

▶ Starting from a set of photon counts maps

↬ **mrs_msvsts_multichannel_deconvolution.pro** : Compute multichannel Poisson deconvolution on spherical 2D-1D HEALPix data with MS-VSTS + multichannel Wavelet Transform method.

USAGE :
mrs_msvsts_multichannel_deconvolution,input,solution,NbrScale1=NbrScale1,NbrScale2=N

INPUT :

- Input = (IDL array) multichannel HEALPix data to be denoised
- beam = (IDL array) set of convolution beams

OUTPUT :

- Solution = (IDL array) multichannel HEALPix denoised image

KEYWORD

- NbrScale1 : Number of scales for the two spatial dimensions (default is 6)
- NbrScale2 : Number of scales for the non-spatial dimension (time or energy) (default is 6)
- niter : Number of iterations
- regularization : if set, uses a regularization parameter (set to 0.01) to improve the convergence speed of the algorithm

Chapter 19

Conclusion

We have presented recent methods for restoration of spherical data with noise following a Poisson distribution. A denoising method was proposed, which used a variance stabilization method and multiscale transforms on the sphere. Experiments have shown it is very efficient for the denoising of astrophysical data set such as Fermi data. Two spherical multiscale transforms, the wavelet and the curvelets, were used. Then, we have described an extension of the denoising method in order to take into account missing data, and we have shown that this inpainting method could be a useful tool to estimate the diffuse emission. Then, we have introduced a new denoising method the sphere which takes into account a background model. The simulated data have shown that it is relatively robust to errors in the model, and can therefore be used for diffuse background modeling and source detection. Finally, an extension to multichannel denoising and deconvolution has been proposed, which proved very efficient on simulated data.

Bibliography

Abdo, A. A., Ackermann, M., and al: 2009, Fermi/large area telescope bright gamma-ray source list, *ApJS* **183**, 46

Abramovich, F., Benjamini, Y., Donoho, D., and Johnstone, I.: 2006, Adapting to unknown sparsity by controlling the false discovery rate, *Annals of Statistics* 34(2), 584–653

Abrial, P., Moudden, Y., Starck, J., Bobin, J., Fadili, M., Afeyan, B., and Nguyen, M.: 2007, Morphological component analysis and inpainting on the sphere: Application in physics and astrophysics, *Journal of Fourier Analysis and Applications* 13(6), 729–748

Abrial, P., Moudden, Y., Starck, J., Fadili, M. J., Delabrouille, J., and Nguyen, M.: 2008, CMB data analysis and sparsity, *Statistical Methodology* 5(4), 289-298

Afeyan, B., Won, K., Starck, J. L., Stevens, R., Mapoles, E., Johnson, M., and Haan, S.: 2006, MODEM: Morphological diversity extraction method to identify, classify and characterize surface defects of spherical ICF targets via AFM and phase shifting diffraction interferometry, in *Proceedings of the 17th Target Fabrication Meeting, San Diego, CA*, 1–5

Aghanim, N. and Forni, O.: 1999, Searching for the non-Gaussian signature of the CMB secondary anisotropies, *Astronomy and Astrophysics* 347, 409–418

Aghanim, N., Kunz, M., Castro, P. G., and Forni, O.: 2003, Non-Gaussianity: Comparing wavelet and Fourier based methods, *Astronomy and Astrophysics* 406, 797-816

Aharon, M., Elad, M., and Bruckstein, A.: 2006, K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation, *IEEE Transactions on Signal Processing* 54(11), 4311–4322

Anscombe, F.: 1948, The transformation of Poisson, binomial and negative-binomial data, *Biometrika* 15, 246–254

Antoine, J., Demanet, L., Jacques, L., and Vandergheynst, P.: 2002, Wavelets on the sphere: Implementation and approximation, *Applied and Computational Harmonic Analysis* 13, 177–200

Antoine, J.-P.: 1999, The 2-D wavelet transform, physical applications and generalizations, in J. van den Berg (ed.), *Wavelets in Physics*, Cambridge University Press, 23–76

Antoine, J.-P. and Vandergheynst, P.: 1999, Wavelets on the 2-sphere: a group-theoretical approach, *Appl. Comput. Harmon. Anal.* 7(3), 262–291

Argyriou, A., Evgeniou, T., and Pontil, M.: 2008, Convex multi-task feature learning, *Machine Learning* 73(3), 243–272

Atzeni, S. and ter Vehn, J. M.: 2004, *The Physics of Inertial Fusion*, Oxford University Press

Bach, F. R.: 2008, Consistency of the group lasso and multiple kernel learning, *Journal of Machine Learning Research* 9, 1179–1225

Ballester, C., Bertalmio, M., Caselles, V., Sapiro, G., and Verdera, J.: August 2001, Filling-in by joint interpolation of vector fields and grey levels, *IEEE Trans. Image Processing* 10, 1200-1211

Banday, A. J., Zaroubi, S., and Górski, K. M.: 2000, On the Non-Gaussianity Observed in the COBE Differential Microwave Radiometer Sky Maps, *Astrophysical Journal* 533, 575-587

Barreiro, R. B. and Hobson, M. P.: 2001, The discriminating power of wavelets to detect non-Gaussianity in the cosmic microwave background, *Monthly Notices of the Royal Astronomical Society* 327, 813–828

Barreiro, R. B., Martínez-González, E., and Sanz, J. L.: 2001, Geometrical estimators as a test of Gaussianity in the cosmic microwave background, *Monthly Notices of the Royal Astronomical Society* 322, 411–418

Benjamini, Y. and Hochberg, Y.: 1995, Controlling the false discovery rate – a practical and powerful approach to multiple testing, *Journal of the Royal Statistical Society B* 57, 289–300

Benjamini, Y. and Yekutieli, Y.: 2001, The control of the false discovery rate in multiple testing under dependency, *Annals of Statistics* 29(4), 1165–1188

Bernardeau, F. and Uzan, J.: 2002, Non-gaussianity in multifield inflation, *Phys. Rev. D* **66**, 103506

Bernardeau, F., van Waerbeke, L., and Mellier, Y.: 2003, Patterns in the weak shear 3-point correlation function, *A&A* 397, 405-414

Bertalmio, M., Bertozzi, A., , and Sapiro, G.: 2001, Navier-Stokes, fluid dynamics, and image and video inpainting, in *in Proc.IEEE Computer Vision and Pattern Recognition (CVPR)*

Bertalmio, M., Sapiro, G., Caselles, V., , and Ballester, C.: July 2000, Image inpainting, *Comput. Graph.(SIGGRAPH 2000)*, 417–424

Bertin, E. and Arnouts, S.: 1996, SExtractor: Software for source extraction, *Astronomy and Astrophysics, Supplement Series* 117, 393–404

Bobin, J., Moudden, Y., Fadili, M. J., and Starck, J.-L.: 2009, Morphological diversity and sparsity for multichannel data restoration, *Journal of Mathematical Imaging and Vision* 33(2), 149–168

Bobin, J., Moudden, Y., Starck, J.-L., and Elad, M.: 2006, Morphological diversity and source separation, *IEEE Transactions on Signal Processing* 13(7), 409–412

Bobin, J., Moudden, Y., Starck, J. L., Fadili, M., and Aghanim, N.: 2008, SZ and CMB reconstruction using generalized morphological component analysis, *Statistical Methodology* 5(4), 307–317

Bobin, J., Starck, J.-L., Fadili, M. J., and Moudden, Y.: 2007a, Sparsity and morphological diversity in blind source separation, *IEEE Transactions on Image Processing* 16(11), 2662–2674

Bobin, J., Starck, J.-L., Fadili, M. J., Moudden, Y., and Donoho, D.: 2007b, Morphological component analysis: An adaptive thresholding strategy, *IEEE Transactions on Image Processing* 16(11), 2675–2681

Bobin, J., Starck, J.-L., Moudden, Y., and Fadili, M. J.: 2008, Blind source separation: The sparsity revolution, *Advances in Imaging and Electron Physics* 152, 221–306

Bogdanova, I., Vandergheynst, P., Antoine, J.-P., Jacques, L., and Mrovidone, M.: 2005, Stereographic wavelet frames on the sphere, *Applied and Computational Harmonic Analysis* 19(2), 223-252

Bouchet, F. R., Bennett, D. P., and Stebbins, A.: 1988, Patterns of the cosmic microwave background from evolving string networks, *Nature* 335, 410–414

Bromley, B. C. and Tegmark, M.: 1999, Is the Cosmic Microwave Background Really Non-Gaussian?, *Astrophysical Journal Letters* 524, L79–L82

Bruce, A., Sardy, S., and Tseng, P.: 1998, Block coordinate relaxation methods for nonparametric signal de-noising, in *Proceedings of the SPIE - The International Society for Optical Engineering*, Vol. 3391, 75–86

Bunn, E. F., Zaldarriaga, M., Tegmark, M., and de Oliveira-Costa, A.: 2003, E/B decomposition of finite pixelized CMB maps, *Phys. Rev. D* **67(2)**, 023501

Candès, E., Demanet, L., Donoho, D., and Ying, L.: 2006, Fast discrete curvelet transforms, *Multiscale Modeling and Simulation* 5(3), 861–899

Candès, E. and Donoho, D.: 1999a, Curvelets – a surprisingly effective nonadaptive representation for objects with edges, in A. Cohen, C. Rabut, and L. Schumaker (eds.), *Curve and Surface Fitting: Saint-Malo 1999*, Vanderbilt University Press, Nashville, TN

Candès, E. and Donoho, D.: 1999b, Ridgelets: the key to high dimensional intermittency?, *Philosophical Transactions of the Royal Society of London A* 357, 2495–2509

Cardoso, J.-F.: 1998, Blind signal separation: statistical principles, *Proceedings of the IEEE. Special issue on blind identification and estimation* **9(10)**, 2009

Cardoso, J.-F.: 1999, High-order contrasts for independent component analysis, *Neural Computation* 11(1), 157–192

Cardoso, J.-F.: 2001, The three easy routes to independent component analysis; contrasts and geometry, in *Proceedings ICA, Independent Component Analysis, 2001*

Cardoso, J.-F.: 2003, Dependence, correlation and non-Gaussianity in independent component analysis, *Journal of Machine Learning Research* 4, 1177–1203

Castro, P. G.: 2003, Bispectrum and the trispectrum of the ostriker-vishniac effect, *Phys. Rev. D* **67**, 123001

Cayón, L., Sanz, J. L., Barreiro, R. B., Martínez-González, E., Vielva, P., Toffolatti, L., Silk, J., Diego, J. M., and Argüeso, F.: 2000, Isotropic wavelets: a powerful tool to extract point sources from cosmic microwave background maps, *Monthly Notices of the Royal Astronomical Society* 315, 757-761

Cayón, L., Sanz, J. L., Martínez-González, E., Banday, A. J., Argüeso, F., Gallegos, J. E., Górski, K. M., and Hinshaw, G.: 2001a, Spherical Mexican hat wavelet: an application to detect non-Gaussianity in the COBE-DMR maps, *Monthly Notices of the Royal Astronomical Society* 326, 1243–1248

Cayón, L., Sanz, J. L., Martínez-González, E., Banday, A. J., Argüeso, F., Gallegos, J. E., Górski, K. M., and Hinshaw, G.: 2001b, Spherical Mexican hat wavelet: an application to detect non-Gaussianity in the COBE-DMR maps, *Monthly Notices of the Royal Astronomical Society* 326, 1243–1248

Chan, T. and Shen, J.: July 2001, Local inpainting models and tv inpainting, *SIAM Journal on Applied Mathematics* 62, 1019-1043

Chen, G. H.-G. and Rockafellar, R. T.: 1997, Convergence rates in forward–backward splitting, *SIAM Journal on Optimization* 7(2), 421–444

Chen, J. and Huo, X.: 2006, Theoretical results on sparse representations for multiple measurement vectors, *IEEE Transactions on Signal Processing* 54(12), 4634–4643

Chen, S. S., Donoho, D. L., and Saunders, M. A.: 1999, Atomic decomposition by basis pursuit, *SIAM Journal on Scientific Computing* 20(1), 33–61

Coifman, R. and Donoho, D.: 1995, Translation invariant de-noising, in A. Antoniadis and G. Oppenheim (eds.), *Wavelets and Statistics*, Springer-Verlag, 125–150

Cooray, A.: 2001, Non-gaussian aspects of thermal and kinetic sunyaev-zel'dovich effects, *Physical Review D* 64, 3514

Cotter, S., Rao, B., Engan, K., and Kreutz-Delgado, K.: 2005, Sparse solutions to linear inverse problems with multiple measurement vectors, *IEEE Transactions on Signal Processing* 53, 2477–2488

Crittenden, R. G.: 2000, Igloo pixelations of the sky, *Astrophysical Letters and Communications* 37, 377–382

Cruz, M., Martínez-González, E., Vielva, P., and Cayón, L.: 2005, Detection of a non-Gaussian spot in WMAP, *Monthly Notices of the Royal Astronomical Society* 356, 29-40

Daubechies, I.: 1988, Time-frequency localization operators: A geometric phase space approach, *IEEE Transactions on Information Theory* 34, 605–612

Daubechies, I. and Sweldens, W.: 1998, Factoring wavelet transforms into lifting steps, *Journal of Fourier Analysis and Applications* 4, 245–267

Delabrouille, J., Cardoso, J., Le Jeune, M., Betoule, M., Fay, G., and Guilloux, F.: 2008, A full sky, low foreground, high resolution CMB map from WMAP, *ArXiv preprint*

Delabrouille, J., Cardoso, J.-F., and Patanchon, G.: 2003, Multi–detector multi–component spectral matching and applications for CMB data analysis, *Monthly Notices of the Royal Astronomical Society* 346(4), 1089-1102

Dineen, P., Rocha, G., and Coles, P.: 2005, Non-random phases in non-trivial topologies, *MNRAS* 358, 1285-1289

Donoho, D.: 1993, Nonlinear wavelet methods for recovery of signals, densities, and spectra from indirect and noisy data, in A. M. Society (ed.), *Proceedings of Symposia in Applied Mathematics*, Vol. 47, 173–205

Donoho, D.: 2004, *For Most Large Underdetermined Systems of Linear Equations, the minimal ℓ^1-norm near-solution approximates the sparsest near-solution*, Technical report, Department of Statistics of Stanford Univ.

Donoho, D. and Duncan, M.: 2000, Digital curvelet transform: strategy, implementation and experiments, in H. Szu, M. Vetterli, W. Campbell, and J. Buss (eds.), *Aerosense 2000, Wavelet Applications VII*, Vol. 4056, SPIE, 12–29

Donoho, D. and Elad, M.: 2003, Optimally sparse representation in general (non-orthogonal) dictionaries via ℓ^1 minimization, *Proceedings of the National Academy of Sciences* 100, 2197–2202

Donoho, D., Elad, M., and Temlyakov, V.: 2006a, Stable recovery of sparse overcomplete representations in the presence of noise, *IEEE Transactions on Information Theory* 52(1), 6–18

Donoho, D. and Flesia, A.: 2002, Digital ridgelet transform based on true ridge functions, in J. Schmeidler and G. Welland (eds.), *Beyond Wavelets*, Academic Press

Donoho, D. and Huo, X.: 2001, Uncertainty principles and ideal atomic decomposition, *IEEE Transactions on Information Theory* 47(7), 2845–2862

Donoho, D. and Jin, J.: 2004a, Higher criticism for detecting sparse heterogeneous mixtures, *Ann. Statist.* **32(3)**, 962

Donoho, D. and Jin, J.: 2004b, *Optimality of excess kurtosis for detecting a non-Gaussian component in high-dimensional random vectors*, Technical report, Stanford University

Donoho, D. and Johnstone, I.: 1994, Ideal spatial adaptation via wavelet shrinkage, *Biometrika* 81, 425–455

Donoho, D., Tsaig, Y., Drori, I., and Starck, J.-L.: 2006b, Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit, *IEEE Transactions on Information Theory*, submitted

Donoho, D. L. and Jin, J.: 2006, Asymptotic minimaxity of false discovery rate thresholding for sparse exponential data, *Annals of Statistics* 34(6), 2980–3018

Doroshkevich, A. G., Naselsky, P. D., Verkhodanov, O. V., Novikov, D. I., Turchaninov, V. I., Novikov, I. D., Christensen, P. R., and Chiang, L.-Y.: 2005, Gauss-Legendre sky pixelization (GLESP) scheme for CMB maps, *International Journal of Modern Physics D* 14(2), 275–290

Elad, M.: 2006, Why simple shrinkage is still relevant for redundant representations?, *IEEE Transactions on Information Theory* 52(12), 5559–5569

Elad, M. and Bruckstein, A.: 2002, A generalized uncertainty principle and sparse representation in pairs of $\mathbf{r}^n$ bases, *IEEE Transactions on Information Theory* 48, 2558–2567

Elad, M., Starck, J.-L., Donoho, D., and Querre, P.: 2005, Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA), *Applied and Computational Harmonic Analysis* 19, 340–358

Eldar, Y. C. and Mishali, M.: 2009, Robust recovery of signals from a structured union of subspaces, *IEEE Transactions on Information Theory*, to appear

Escalera, E. and MacGillivray, H. T.: 1995, Topology in galaxy distributions: method for a multi-scale analysis. A use of the wavelet transform., *Astronomy and Astrophysics* 298, 1-+

Fang, L.-Z. and Feng, L.-l.: 2000, Measuring the Galaxy Power Spectrum and Scale-Scale Correlations with Multiresolution-decomposed Covariance. I. Method, *Astrophysical Journal* **539**, 5

Faÿ, G. and Guilloux, F.: 2008, Consistency of a needlet spectral estimator on the sphere, *ArXiv preprint*

Faÿ, G., Guilloux, F., Betoule, M., Cardoso, J.-F., Delabrouille, J., and Le Jeune, M.: 2008, CMB power spectrum estimation using wavelets, *Phys. Rev. D* 78(8), 083013

Forni, O. and Aghanim, N.: 1999, Searching for non-gaussianity: Statistical tests, *Astronomy and Astrophysics, Supplement Series* 137, 553–567

Freeden, W. and Maier, T.: 2002, On multiscale denoising of spherical functions: Basic theory and numerical aspects, *Electronic Transactions on Numerical Analysis (ETNA)* 14, 40-62

Freeden, W., Maier, T., and Zimmermann, S.: 2003, A survey on wavelet methods for (geo)applications, *Revista Mathematica Complutense* 16(1), 277–310

Freeden, W. and Schneider, F.: 1998, Regularization wavelets and multiresolution, *Inverse Problems* 14, 225–243

Freeden, W. and Windheuser, U.: 1997, Combined spherical harmonics and wavelet expansion – a future concept in Earth's gravitational potential determination, *Applied and Computational Harmonic Analysis* 4, 1–37

Fryźlewicz, P. and Nason, G. P.: 2004a, A Haar-Fisz algorithm for Poisson intensity estimation, *Journal of Computational and Graphical Statistics* 13, 621-638

Fryźlewicz, P. and Nason, G. P.: 2004b, A Haar-Fisz algorithm for Poisson intensity estimation, *Journal of Computational and Graphical Statistics* 13, 621–638

Fuchs, J.: 2005, Recovery of exact sparse representations in the presence of bounded noise, *IEEE Trans. On Information Theory* 51, 3601-3608

Górski, K. M., Hivon, E., Banday, A. J., Wandelt, B. D., Hansen, F. K., Reinecke, M., and Bartelmann, M.: 2005, HEALPix: A framework for high-resolution discretization and fast analysis of data distributed on the sphere, *Astrophysical Journal* 622, 759–771

Gribonval, R. and Nielsen, M.: 2003, Sparse representations in unions of bases, *IEEE Transactions on Information Theory* 49(12), 3320-3325

Gribonval, R. and Nielsen, M.: 2008, Beyond sparsity: recovering structured representations by 1-minimization and greedy algorithms. Application to the analysis of sparse underdetermined ICA, *Journal of Advances in Computational Mathematics* 28, 23–41

Gribonval, R., Rauhut, H., Schnass, K., and Vandergheynst, P.: 2008, Atoms of all channels, unite! Average case analysis of multi-channel sparse recovery using greedy algorithms, *Journal of Fourier Analysis and Applications* 14(5), 655–687

Herrmann, F., Moghaddam, P., and Stolk, C.: 2008, Sparsity- and continuity-promoting seismic image recovery with curvelet frames, *ACHA* 24(2), 150-173

Hobson, M. P., Jones, A. W., and Lasenby, A. N.: 1999, Wavelet analysis and the detection of non-Gaussianity in the cosmic microwave background, *Monthly Notices of the Royal Astronomical Society* 309, 125–140

Holschneider, M.: 1996, Wavelet analysis on the sphere, *Journal of Mathematical Physics* 37(8), 4156–4165

Hyvärinen, A., Karhunen, J., and Oja, E.: 2001, *Independent Component Analysis*, John Wiley, New York, 481+xxii pages

Jewell, J.: 2001, A Statistical Characterization of Galactic Dust Emission as a Non-Gaussian Foreground of the Cosmic Microwave Background, *Astrophysical Journal* 557, 700–713

Jin, J., Starck, J.-L., Donoho, D., Aghanim, N., and Forni, O.: 2005, Cosmological non-Gaussian signatures detection: Comparison of statistical tests, *EURASIP Journal* 15, 2470–2485

Kolaczyk, E. D.: 1999, Bayesian multiscale models for Poisson processes, *Journal of the American Statistical Association* 94(447), 920-933

Komatsu, E., Kogut, A., Nolta, M. R., Bennett, C. L., Halpern, M., Hinshaw, G., Jarosik, N., Limon, M., Meyer, S. S., Page, L., Spergel, D. N., Tucker, G. S., Verde, L., Wollack, E., and Wright, E. L.: 2003, First-Year Wilkinson Microwave Anisotropy Probe (WMAP) Observations: Tests of Gaussianity, *Astrophysical Journal, Supplement Series* 148, 119-134

Kovac, J. M., Leitch, E. M., Pryke, C., Carlstrom, J. E., Halverson, N. W., and Holzapfel, W. L.: 2002, Detection of polarization in the cosmic microwave background using DASI, *Nature* 420, 772-787

Kunszt, P. Z., Szalay, A. S., and Thakar, A. R.: 2001, The hierarchical triangular mesh, in A. J. Banday, S. Zaroubi, and M. Bartelmann (eds.), *Mining the Sky*, 631–637

Kunz, M., Banday, A. J., Castro, P. G., Ferreira, P. G., and Górski, K. M.: 2001, The Trispectrum of the 4 Year COBE DMR Data, *Astrophysical Journal Letters* 563, L99–L102

Lefkimmiaits, S., Maragos, P., and Papandreou, G.: 2009, Bayesian inference on multiscale models for poisson intensity estimation: Applications to photon-limited image denoising, *IEEE Transactions on Image Processing* vol. 20, no. 20

Lehmann, E. L.: 1986, *Testing Statistical Hypotheses, 2nd ed.*, John Wiley & Sons

Lindl, J.: 1997, *Inertial Confinement Fusion: The Quest for Ignition and Energy Gain Using Indirect Drive*, AIP, American Institute of Physics Press

López-Caniego, M., Herranz, D., Barreiro, R. B., and Sanz, J. L.: 2005, Filter design for the detection of compact sources based on the Neyman-Pearson detector, *Monthly Notices of the Royal Astronomical Society*, 368-+

Lounici, K., Pontil, M., Tsybakov, A. B., and van de Geer, S.: 2009, Taking advantage of sparsity in multi-task learning, in *22th Annual Conference on Learning Theory, COLT*

Malioutov, D., Çetin, M., and Willsky, A. S.: 2005, A sparse signal reconstruction perspective for source localization with sensor arrays, *IEEE Transactions on Image Processing* 53(8), 3010-3022

Mallat, S. and Zhang, Z.: 1993, Matching pursuits with time-frequency dictionaries, *IEEE Transactions on Signal Processing* 41(12), 3397–3415

Marinucci, D., Pietrobon, D., Balbi, A., Baldi, P., Cabella, P., Kerkyacharian, G., Natoli, P., Picard, D., and Vittorio, N.: 2008, Spherical needlets for cosmic microwave background data analysis, *Monthly Notices of the Royal Astronomical Society* 383, 539–545

Martínez, V. J., Starck, J.-L., Saar, E., Donoho, D. L., Reynolds, S. C., de la Cruz, P., and Paredes, S.: 2005, Morphology of the Galaxy Distribution from Wavelet Denoising, *Astrophysical Journal* 634, 744-755

Masnou, S. and Morel, J.: 1998, Level lines based disocclusion, in *IEEE International Conference on Image Processing*, Vol. III, 259–263

Masnou, S. and Morel, J.: 2002, Disocclusion: A variational approach using level lines, *IEEE Transactions on Image Processing* 11(2), 68–76

McEwen, J. D., Hobson, M. P., Mortlock, D. J., and Lasenby, A. N.: 2005, Fast directional continuous spherical wavelet transform algorithms, *ArXiv preprint*

McEwen, J. D., Vielva, P., Wiaux, Y., Barreiro, R. B., Cayon, L., Hobson, M. P., Lasenby, A. N., Martinez-Gonzalez, E., and Sanz, J. L.: 2007, Cosmological applications of a wavelet analysis on the sphere, *Journal of Fourier Analysis and Applications* 13, 495–510

Moudden, Y., Cardoso, J.-F., Starck, J.-L., and Delabrouille, J.: 2005, Blind component separation in wavelet space: Application to CMB analysis, *EURASIP Journal on Applied Signal Processing* 15, 2437–2454

Mukherjee, P. and Wang, Y.: 2003, Model-independent Reconstruction of the Primordial Power Spectrum from Wilkinson Microwave Anistropy Probe Data, *Astrophysical Journal* 599, 1-6

Mukherjee, P. and Wang, Y.: 2004, Wavelets and Wilkinson Microwave Anisotropy Probe Non-Gaussianity, *Astrophysical Journal* 613, 51-60

Myers, J.: 2009, Lat background models, *LAT Background Models*, `http://fermi.gsfc.nasa.gov/ssc/data/access/lat/BackgroundModels.html`

Myers, J.: 2010, Lat 1-year point source catalog, *LAT 1-year Point Source Catalog*, `http://fermi.gsfc.nasa.gov/ssc/data/access/lat/1yr_catalog/`

Naselsky, P., Chiang, L.-Y., Olesen, P., and Novikov, I.: 2005, Statistics of phase correlations as a test for non-Gaussianity of the CMB maps, *Phys. Rev. D* 72(6), 063512-+

Negahban, S. and Wainwright, M. J.: 2009, *Simultaneous support recovery in high dimensions: Benefits and perils of block ℓ_1/ℓ_∞-regularization*, Technical Report 774, UC Berkeley

Novikov, D., Schmalzing, J., and Mukhanov, V. F.: 2000, On non-Gaussianity in the cosmic microwave background, *Astronomy and Astrophysics* 364, 17–25

Nowak, R. D. and Kolaczyk, E. D.: 2000, A statistical multiscale framework for Poisson inverse problems, *IEEE Transactions on Information Theory* 46(5), 1811-1825

Peyré, G., Fadili, M. J., and Starck, J.-L.: 2007, Learning adapted dictionaries for geometry and texture separation, in *Wavelet XII*, SPIE, San Diego

Pham, D.-T.: 2001, Joint approximate diagonalization of positive definite matrices, *SIAM Journal on Matrix Analysis and Applications* 22(4), 1136-1152

Phillips, N. G. and Kogut, A.: 2001, Statistical Power, the Bispectrum, and the Search for Non-Gaussianity in the Cosmic Microwave Background Anisotropy, *Astrophysical Journal* 548, 540–549

Rahman, I. U., Drori, I., Stodden, V. C., Donoho, D. L., and Schröder, P.: 2005, Multiscale representations for manifold-valued data, *Multiscale Modeling & Simulation* 4(4), 1201-1232

Riazuelo, A., Uzan, J.-P., Lehoucq, R., and Weeks, J.: 2002, Simulating cosmic microwave background maps in multi-connected spaces, *Simulating Cosmic Microwave Background maps in multi-connected spaces*, astro-ph/0212223

Rocha, G., Cayón, L., Bowen, R., Canavezes, A., Silk, J., Banday, A. J., and Górski, K. M.: 2004, Topology of the Universe from COBE-DMR - a wavelet approach, *Monthly Notices of the Royal Astronomical Society* 351, 769-778

Romeo, A. B., Horellou, C., and Bergh, J.: 2003, N-body simulations with two-orders-of-magnitude higher performance using wavelets, *Monthly Notices of the Royal Astronomical Society* 342, 337-344

Romeo, A. B., Horellou, C., and Bergh, J.: 2004, A wavelet add-on code for new-generation n-body simulations and data de-noising (jofiluren), *Monthly Notices of the Royal Astronomical Society* 354, 1208-1222

Rousseeuw, P. and Croux, C.: 1993, Alternatives to the median absolute deviation., *Journal of the American Statistical Asssociation* 88, 1273-1283

Sanz, J. L., Herranz, D., and Martínez-Gónzalez, E.: 2001, Optimal Detection of Sources on a Homogeneous and Isotropic Background, *Astrophysical Journal* 552, 484-492

Schmitt, J., Starck, J. L., Casandjian, J. M., Fadili, J., and Grenier, I.: 2010, Poisson denoising on the sphere: application to the fermi gamma ray space telescope, *Astronomy and Astrophysics* 517

Schröder, P. and Sweldens, W.: 1995, Spherical wavelets: Efficiently representing functions on the sphere, *SIGGRAPH 95, Computer Graphics Proceedings*, 161–172

Shandarin, S. F.: 2002, Testing non-Gaussianity in cosmic microwave background maps by morphological statistics, *Monthly Notices of the Royal Astronomical Society* 331, 865+

Shorack, G. R. and Wellner, J. A.: 1986, *Empirical Processes with Applications to Statistics*, John Wiley & Sons

Slezak, E., de Lapparent, V., and Bijaoui, A.: 1993, Objective detection of voids and high density structures in the first CfA redshift survey slice, *Astrophysical Journal* 409, 517–529

Smoot, G. F., Bennett, C. L., Kogut, A., Wright, E. L., Aymon, J., Boggess, N. W., Cheng, E. S., de Amici, G., Gulkis, S., Hauser, M. G., Hinshaw, G., Jackson, P. D., Janssen, M., Kaita, E., Kelsall, T., Keegstra, P., Lineweaver, C., Loewenstein, K., Lubin, P., Mather, J., Meyer, S. S., Moseley, S. H., Murdock, T., Rokke, L., Silverberg, R. F., Tenorio, L., Weiss, R., and Wilkinson, D. T.: 1992, Structure in the COBE differential microwave radiometer first-year maps, *Astrophysical Journal Letters* 396, L1–L5

Starck, J., Fadili, J. M., Digel, S., Zhang, B., and Chiang, J.: 2009a, Source detection using a 3D sparse representation: application to the Fermi gamma-ray space telescope, *å* **504**, 641

Starck, J., Moudden, Y., and Bobin, J.: 2009b, Polarized wavelets and curvelets on the sphere, *A&A* **497**, 931

Starck, J.-L., Aghanim, N., and Forni, O.: 2004a, Detecting cosmological non-Gaussian signatures by multi-scale methods, *Astronomy and Astrophysics* 416, 9–17

Starck, J.-L., Bijaoui, A., Lopez, B., and Perrier, C.: 1994, Image reconstruction by the wavelet transform applied to aperture synthesis, *Astronomy and Astrophysics* 283, 349–360

Starck, J.-L., Candès, E., and Donoho, D.: 2002a, The curvelet transform for image denoising, *IEEE Transactions on Image Processing* 11(6), 131–141

Starck, J.-L., Candès, E., and Donoho, D.: 2003a, Astronomical image representation by the curvelet tansform, *Astronomy and Astrophysics* 398, 785–800

Starck, J.-L., Donoho, D. L., and Candès, E. J.: 2001, Very high quality image restoration by combining wavelets and curvelets, in *Wavelets: Applications in Signal and Image Processing IX*, Vol. 4478, SPIE, 9–19

Starck, J.-L., Elad, M., and Donoho, D.: 2004b, Redundant multiscale transforms and their application for morphological component analysis, *Advances in Imaging and Electron Physics* 132

Starck, J.-L., Fadili, J., and Murtagh, F.: 2007a, The undecimated wavelet decomposition and its reconstruction, *IEEE Transactions on Image Processing* 16, 297–309

Starck, J.-L., Fadili, M. J., and Murtagh, F.: 2007b, The Undecimated Wavelet Decomposition and its Reconstruction, *IEEE Transactions on Image Processing* 16(2), 297–309

Starck, J.-L., Martinez, V., Donoho, D., Levi, O., Querre, P., and Saar, E.: 2005, Analysis of the spatial distribution of galaxies by multiscale methods, *EURASIP Journal on Applied Signal Processing* 15, 2455–2469

Starck, J.-L., Moudden, Y., Abrial, P., and Nguyen, M.: 2006, Wavelets, ridgelets and curvelets on the sphere, *Astronomy and Astrophysics* 446, 1191–1204

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

Starck, J.-L. and Murtagh, F.: 2006, *Astronomical Image and Data Analysis*, Springer, 2nd edn.

Starck, J.-L., Murtagh, F., and Bertero, M.: 2011, Handbook of mathematical methods in imaging, *Handbook of Mathematical Methods in Imaging*, Chapt. The Starlet Transform in Astronomical Data Processing: Application to Source Detection and Image Deconvolution, pp 1489–1531, Springer

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

Starck, J.-L., Murtagh, F., Candès, E., and Donoho, D.: 2003b, Gray and color image contrast enhancement by the curvelet transform, *IEEE Transactions on Image Processing* 12(6), 706–717

Starck, J.-L., Murtagh, F., and Fadili, M.: 2010, *Sparse Image and Signal Processing*, Cambridge University Press

Starck, J.-L., Pantin, E., and Murtagh, F.: 2002b, Deconvolution in astronomy: a review, *Publications of the Astronomical Society of the Pacific* 114, 1051–1069

Starck, J.-L., Pires, S., and Réfrégier, A.: 2006, Weak lensing mass reconstruction using wavelets, *A&A* 451, 1139-1150

Sunyaev, R. A. and Zeldovich, I. B.: 1980, Microwave background radiation as a probe of the contemporary structure and history of the universe, *Annual Review of Astronomy and Astrophysics* 18, 537-560

Tegmark, M.: 1996, An icosahedron-based method for pixelizing the celestial sphere, *Astrophysical Journal Letters* 470, L81–L84

Tenorio, L., Jaffe, A. H., Hanany, S., and Lineweaver, C. H.: 1999, Applications of wavelets to the analysis of Cosmic Microwave Background maps, *Monthly Notices of the Royal Astronomical Society* 310, 823–834

Timmermann, K. E. and Nowak, R.: 1999, Multiscale modeling and estimation of Poisson processes with applications to photon-limited imaging, *IEEE Transactions on Signal Processing* 46, 886–902

Tropp, J. A.: 2006, Algorithms for simultaneous sparse approximation. Part II: Convex relaxation, *Signal Processing* 86(589-602)

Tseng, P.: 2001, Convergence of a block coordinate descent method for nondifferentiable minimizations, *Journal of Optimization Theory and Applications* 109(3), 457–494

Verde, L., Wang, L., Heavens, A. F., and Kamionkowski, M.: 2000, Large-scale structure, the cosmic microwave background and primordial non-Gaussianity, *Monthly Notices of the Royal Astronomical Society* 313, 141–147

Vielva, P., Martínez-González, E., Barreiro, R. B., Sanz, J. L., and Cayón, L.: 2004, Detection of non-Gaussianity in the Wilkinson Microwave Anisotropy Probe first-year data using spherical wavelets, *Astrophysical Journal* 609, 22–34

Vielva, P., Wiaux, Y., Martínez-González, E., and Vandergheynst, P.: 2006, Steerable wavelet analysis of CMB structures alignment, *New Astronomy Review* 50, 880–888

Vio, R., Tenorio, L., and Wamsteker, W.: 2002, On optimal detection of point sources in CMB maps, *Astronomy and Astrophysics* 391, 789-794

White, R. A. and Stemwedel, S. W.: 1992, The quadrilateralized spherical cube and quad-tree for all sky data, in D. M. Worrall, C. Biemesderfer, and J. Barnes (eds.), *Astronomical Data Analysis Software and Systems I*, Vol. 25 of *Astronomical Society of the Pacific Conference Series*, 379–381

Wiaux, Y., Jacques, L., and Vandergheynst, P.: 2005, Correspondence Principle between Spherical and Euclidean Wavelets, *Astrophysical Journal* 632, 15-28

Wiaux, Y., Jacques, L., and Vandergheynst, P.: 2007, Fast spin +-2 spherical harmonics transforms and application in cosmology, *J. Comput. Phys.* 226, 2359

Wiaux, Y., Jacques, L., Vielva, P., and Vandergheynst, P.: 2006, Fast directional correlation on the sphere with steerable filters, *Astrophysical Journal* 652, 820–832

Wiaux, Y., McEwen, J. D., Vandergheynst, P., and Blanc, O.: 2008, Exact reconstruction with directional wavelets on the sphere, *Monthly Notices of the Royal Astronomical Society* 388, 770–788

Wiaux, Y., McEwen, J. D., and Vielva, P.: 2007, Complex data processing: fast wavelet analysis on the sphere, *Journal of Fourier Analysis and Applications* 13, 477–493

Yamada, I.: 2001, The hybrid steepest descent method for the variational inequality problem over the intersection of fixed point sets of nonexpansive mappings, in D. Butnariu, Y. Censor, and S. Reich (eds.), *Inherently Parallel Algorithms in Feasibility and Optimization and Their Applications*, Elsevier

Zaldarriaga, M.: 1998, Cosmic Microwave Background Polarization Experiments, *Astrophysical Journal* 503, 1-+

Zaldarriaga, M. and Seljak, U.: 1997, All-sky analysis of polarization in the microwave background, *Phys. Rev. D* **55**, 1830

Zhang, B., Fadili, M. J., and Starck, J.-L.: 2008, Wavelets, ridgelets and curvelets for Poisson noise removal, *IEEE Transactions on Image Processing* 17(7), 1093–1108

Index

List of Algorithms