HAL
open science

3D Sparse Representations

François Lanusse, Jean-Luc Starck, Arnaud Woiselle, Jalal M. Fadili

To cite this version:

François Lanusse, Jean-Luc Starck, Arnaud Woiselle, Jalal M. Fadili. 3D Sparse Representations. Peter W. Hawkes. Advances in Imaging and Electron Physics, 183, Academic Press, Elsevier, pp.99 - 204, 2014, 10.1016/B978-0-12-800265-0.00003-5 . hal-01132660

HAL Id: hal-01132660

https://hal.science/hal-01132660

Submitted on 17 Mar 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

3D Sparse Representations

Lanusse F. [a] Starck J.-L. [a] Woiselle A. [c] Fadili M.J. [b]

[a] *Laboratoire AIM, UMR CEA-CNRS-Paris 7, Irfu, Service d'Astrophysique, CEA Saclay, F-91191 GIF-SUR-YVETTE Cedex, France.*

[b] *GREYC CNRS UMR 6072, Image Processing Group, ENSICAEN 14050, Caen Cedex, France.*

[c] *Sagem Defense Securite, 95101 Argenteuil CEDEX, France.*

Abstract

In this chapter we review a variety of 3D sparse representations developed in recent years and adapted to different kinds of 3D signals. In particular, we describe 3D wavelets, ridgelets, beamlets and curvelets. We also present very recent 3D sparse representations on the 3D ball adapted to 3D signal naturally observed in spherical coordinates. Illustrative examples are provided for the different transforms.

Key words: Sparse Representation, 3D transforms, Ridelet, Curvelet, 3D spherical data, Morphological diversity

Contents

1 Introduction

Sparse representations such as wavelets or curvelets have been very successful for 2D image processing. Impressive results were obtained for many applications such as compression (see [1] for an example of Surflet compression; the new image standard JPEG2000 is based on wavelets rather than DCT like JPEG), denoising [2, 3, 4], contrast enhancement [5], inpainting [6, 7] or deconvolution [8, 9]. Curvelets [3, 10], Bandelets [11] and Contourlets [12] were designed to well represent edges in an image while wavelets are especially efficient for isotropic feature analysis.

With the increasing computing power and memory storage capabilities of computers, it has become feasible to analyze 3D data as a volume and not only slice-by-slice, which would mistakingly miss the 3D geometrical nature of the data. Among the most simple transforms extended to 3D are the separable Wavelet transform (decimated, undecimated, or any other kind) and the Discrete Cosine transform, as these are separable transforms and thus the extension is straightforward. The DCT is mainly used in video compression, but has also been used in denoising [13]. As for the 3D wavelets, they have already been used in denoising applications in many domains [14, 15, 16].

However these separable transforms lack the directional nature which has made the success of 2D transforms like curvelets. Consequently, a lot of effort has been made in the last years to build sparse 3D data representations, which better represent geometrical features contained in the data. The 3D beamlet transform [17] and the 3D ridgelet transform [18] were respectively designed for 1D and 2D features detection. Video denoising using the ridgelet transform was proposed in [19]. These transforms were combined with 3D wavelets to build BeamCurvelets and RidCurvelets [20] which are extensions of the first generation curvelets [3]. Whereas most 3D transforms are adapted to plate-like features, the BeamCurvelet transform is adapted to filaments of different scales and different orientations. Another extension of the curvelets to 3D is the 3D fast curvelet transform [21] which consists in paving the Fourier domain with angular wedges in dyadic concentric squares using the parabolic scaling law to fix the number of angles depending on the scale, and has atoms designed for representing surfaces in 3D. The Surflet transform [22] – a d-dimensional extension of the 2D wedgelets [23, 24] – has been studied for compression purposes [1]. Surflets are an adaptive transform estimating each cube of a quad-tree decomposition of the data by two regions of constant value separated by a polynomial surface. Another possible representation uses the Surfacelets developed by Do and Lu [25]. It relies on the combination of a Laplacian pyramid and a d-dimensional directional filter bank. Surfacelets produce a tiling of the Fourier space in angular wedges in a way close to the curvelet transform, and can be interpreted as a 3D adaptation of the 2D

contourlet transform. This transformation has also been applied to video denoising [26]. More recently, Shearlets [27] have also been extended to 3D [28] and subsequently applied to video denoising and enhancement.

All these 3D transforms are developed on Cartesian grids and are therefore appropriate to process 3D cubes. However, in fields like geophysics and astrophysics, data is often naturally accessible on the sphere. This fact has led to the development of sparse representations on the sphere. Many wavelet transforms on the sphere have been proposed in the past years. [29] proposed an invertible isotropic undecimated wavelet transform (UWT) on the sphere, based on spherical harmonics. A similar wavelet construction [30, 31, 32] used the so-called needlet filters. [33] also proposed an algorithm which permits to reconstruct an image from its steerable wavelet transform. Since reconstruction algorithms are available, these tools have been used for many applications such as denoising, deconvolution, component separation [34, 35, 36] or inpainting [37, 38]. However they are limited to 2D spherical data.

Some signals on the sphere have an additional time or energy dependency independent of the angular dimension. They are not truly 3D but rather 2D-1D as the additional dimension is not linked to the spatial dimension. An extension of the wavelets on the sphere to this 2D-1D class of signals has been proposed in [39] with an application to Poisson denoising of multichannel data on the Sphere. More recently, fully 3D invertible wavelet transforms have been formulated in spherical coordinates [40, 41]. These transforms are suited to signals on the 3D ball (i.e. on the solid sphere) which arise for instance in astrophysics in the study of large scale distribution of galaxies when both angular and radial positions are available.

The aim of this chapter is to review different kinds of 3D sparse representations among those mentioned above, providing descriptions of the different transforms and examples of practical applications. In Section 2, we present several constructions of separable 3D and 2D-1D wavelets. Section 3 describes the 3D Ridgelet and Beamlet transforms which are respectively adapted to surfaces and lines spanning the entire data cube. These transforms are used as building blocks of the first generation 3D curvelets presented in Section 4 which can sparsely represent either plates or lines of different sizes, scales and orientations. In Section 5, the 3D Fast Curvelet is presented along with a modified Low-Redundancy implementation to address the issue of the prohibitively redundant original implementation. Section 6 introduces wavelets on the sphere and their extension to the 2D-1D case while providing some of the background necessary to build the wavelet on the 3D ball described in 7.

2 3D Wavelets

In this section we present two 3D discrete wavelet constructions based on filter banks to enable fast transform (in $O(N^3)$ where N^3 is the size of the data cube). These transforms, namely the 3D biorthogonal wavelet and the 3D Isotropic Undecimated Wavelet Transform, are built by separable tensor products of 1D wavelets and are thus simple extensions of the 2D transforms. They are complementary in the sense that the biorthogonal wavelet has no redundancy which is especially appreciable in 3D at the cost of low performance in data restoration purposes while the Isotropic Undecimated Wavelet Transform is redundant but performs very well in restoration applications.
We also present a 2D-1D wavelet transform in Cartesian coordinates. In the final part of this section, this 2D-1D transform is demonstrated in an application to time-varying source detection in the presence of Poisson noise.

2.1 3D biorthogonal wavelets

2.1.1 Discrete Wavelet Transform

The Discrete Wavelet Transform is based on Multiresolution analysis [42] which results from a sequence of embedded closed subspaces generated by interpolations at different scales.

We consider dyadic scales $a = 2^j$ for increasing integer values of j. From the function, $f(x) \in L_2(\mathbb{R})$, a ladder of approximation subspaces is constructed with the embeddings

$$\ldots \subset V_3 \subset V_2 \subset V_1 \subset V_0 \ldots \tag{1}$$

such that, if $f(x) \in V_j$ then $f(2x) \in V_{j+1}$.

The function $f(x)$ is projected at each level j onto the subspace V_j. This projection is defined by the approximation coefficient $c_j[l]$, the inner product of $f(x)$ with the dilated-scaled and translated version of the *scaling function* $\phi(x)$:

$$c_j[l] = \left\langle f, \phi_{j,l} \right\rangle = \left\langle f, 2^{-j}\phi(2^{-j}. - l) \right\rangle . \tag{2}$$

$\phi(t)$ is a scaling function which satisfies the property

$$\frac{1}{2}\phi\left(\frac{x}{2}\right) = \sum_k h[k]\phi(t-k) \ , \tag{3}$$

or equivalently in the Fourier domain

$$\hat{\phi}(2\nu) = \hat{h}(\nu)\hat{\phi}(\nu) \qquad \text{where } \hat{h}(\nu) = \sum_k h[k]e^{-2\pi i k\nu} \ . \tag{4}$$

Expression (3) allows the direct computation of the coefficients c_{j+1} from c_j. Starting from c_0, all the coefficients $(c_j[l])_{j>0,l}$ can be computed without directly evaluating any other inner product:

$$c_{j+1}[l] = \sum_k h[k-2l]c_j[k] \ . \tag{5}$$

At each level j, the number of inner products is divided by 2. Step-by-step the signal is smoothed and information is lost. The remaining information (details) can be recovered from the subspace W_{j+1}, the orthogonal complement of V_{j+1} in V_j. This subspace can be generated from a suitable wavelet function $\psi(t)$ by translation and dilation:

$$\frac{1}{2}\psi\left(\frac{t}{2}\right) = \sum_k g[k]\phi(t-k) \ , \tag{6}$$

or by taking the Fourier transform of both sides

$$\hat{\psi}(2\nu) = \hat{g}(\nu)\hat{\phi}(\nu) \qquad \text{where} \quad \hat{g}(\nu) = \sum_k g[k]e^{-2\pi i k\nu} \ . \tag{7}$$

The wavelet coefficients at level $j+1$ are computed from the approximation at level j as the inner products

$$\begin{aligned} w_{j+1}[l] &= \left\langle f, \psi_{j+1,l} \right\rangle = \left\langle f, 2^{-(j+1)}\psi(2^{-(j+1)}. - l) \right\rangle \\ &= \sum_k g[k-2l]c_j[k] \ . \end{aligned} \tag{8}$$

From (5) and (8), only half the coefficients at a given level are necessary to compute the wavelet and approximation coefficients at the next level. Therefore, at each level, the coefficients can be decimated without loss of information. If the notation $[\cdot]_{\downarrow 2}$ stands for the decimation by a factor 2 (i.e. only even samples are kept), and $\bar{h}[l] = h[-l]$, the relation between approximation and detail coefficients between two successive scales can be written:

$$\begin{aligned} c_{j+1} &= [\bar{h} \star c_j]_{\downarrow 2} \\ w_{j+1} &= [\bar{g} \star c_j]_{\downarrow 2}. \end{aligned} \tag{9}$$

This analysis constitutes the first part of a filter bank [43]. In order to recover the original data, we can use the properties of orthogonal wavelets, but the theory has been generalized to biorthogonal wavelet bases by introducing the

filters $\tilde{h}$ and $\tilde{g}$ [44], defined to be dual to h and g such that $(h, g, \tilde{h}, \tilde{g})$ is a perfect reconstruction filter bank i.e. the filters $\tilde{h}$ and $\tilde{g}$ must verify the biorthogonal conditions of dealiasing and exact reconstruction [45]:

- *Dealiasing*:

$$\hat{h}^* \left(\nu + \frac{1}{2}\right) \hat{\tilde{h}}(\nu) + \hat{g}^* \left(\nu + \frac{1}{2}\right) \hat{\tilde{g}}(\nu) = 0 \ . \tag{10}$$

- *Exact reconstruction:*

$$\hat{h}^*(\nu)\hat{\tilde{h}}(\nu) + \hat{g}^*(\nu)\hat{\tilde{g}}(\nu) = 1 \ . \tag{11}$$

Note that in terms of filter banks, the biorthogonal wavelet transform becomes orthogonal when $h = \tilde{h}$ and $g = \tilde{g}$, in which case h is a conjugate mirror filter.

The reconstruction of the signal is then performed by

$$\begin{aligned} c_j[l] &= 2 \sum_k \left(\tilde{h}[k + 2l] c_{j+1}[k] + \tilde{g}[k + 2l] w_{j+1}[k] \right) \\ &= 2(\tilde{h} \star [c_{j+1}]_{\uparrow 2} + \tilde{g} \star [w_{j+1}]_{\uparrow 2})[l] \ , \end{aligned} \tag{12}$$

where $[c_{j+1}]_{\uparrow 2}$ is the zero-interpolation of c_{j+1} defined by zero insertions

$$[c_{j+1}]_{\uparrow 2}[l] = \begin{cases} c_{j+1}[m] & \text{if } l = 2m \\ 0 & \text{otherwise.} \end{cases} \ , \tag{13}$$

Equations (9) and (12) are used to define the fast pyramidal algorithm associated with the biorthogonal wavelet transform, illustrated by Fig. 1. In the decomposition (9), c_{j+1} and w_{j+1} are computed by successively convolving c_j with the filters $\bar{h}$ (low pass) and $\bar{g}$ (high pass). Each resulting channel is then downsampled (decimated) by suppression of one sample out of two. The high frequency channel w_{j+1} is left, and the process is iterated with the low frequency part c_{j+1}. This is displayed in the upper part of Fig. 1. In the reconstruction or synthesis side, the coefficients are up-sampled by inserting a 0 between each sample, and then convolved with the dual filters $\tilde{h}$ and $\tilde{g}$, the resulting coefficients are summed and the result is multiplied by 2. The procedure is iterated up to the smallest scale as depicted in the lower part of Fig. 1.

This fast pyramidal algorithm for the biorthogonal discrete wavelet transform is computationally very efficient, requiring $O(N)$ operations for data with N samples as compared to $O(N \log N)$ of the FFT.

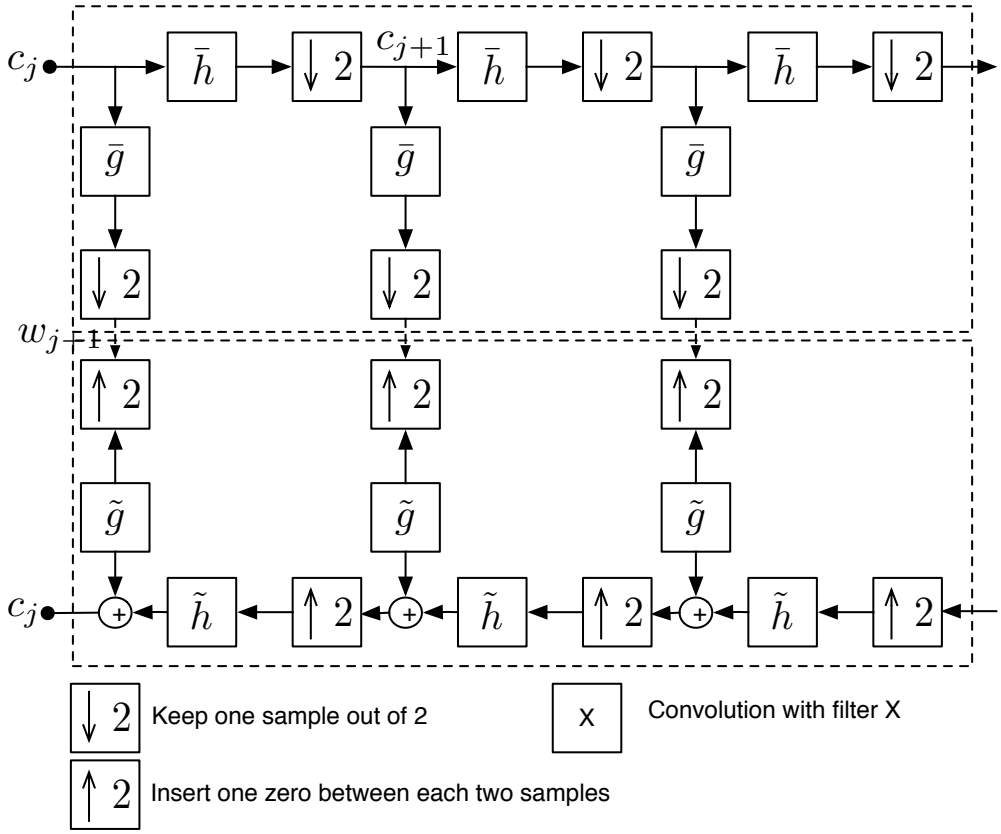


Fig. 1. Fast pyramidal algorithm associated with the biorthogonal wavelet transform. Top: Fast analysis transform with a cascade of filtering with $\bar{h}$ and $\bar{g}$ followed by factor 2 subsampling. Bottom: Fast inverse transform by progressively inserting zeros and filtering with dual filters $\tilde{h}$ and $\tilde{g}$.

2.1.2 Three-Dimensional Decimated Wavelet Transform

The above DWT algorithm can be extended to any dimension by *separable* (tensor) products of a scaling function ϕ and a wavelet ψ.

In the three-dimensional algorithm, the scaling function is defined by $\phi(x, y, z) = \phi(x)\phi(y)\phi(z)$, and the passage from one resolution to the next is achieved by:

$$\begin{aligned} c_{j+1}[k, l, m] &= \sum_{p,q,r} h[p - 2k]h[q - 2l]h[r - 2m]c_j[p, q, r] \\ &= [\bar{h}\bar{h}\bar{h} \star c_j]_{\downarrow 2,2,2}[k, l, m] \ , \end{aligned} \tag{14}$$

where $[.]_{\downarrow 2,2,2}$ stands for the decimation by factor 2 along all x-, y- and z- axes (i.e. only even pixels are kept) and $h_1 h_2 h_3 \star c_j$ is the 3D discrete convolution of c_j by the separable filter $h_1 h_2 h_3$ (i.e. convolution first along the x-axis by h_1, then convolution along the y-axis by h_2 and finally convolution allong the z-axis by h_3).

The detail coefficients are obtained from seven wavelets:

- x wavelet: $\psi^1(x,y,z) = \psi(x)\phi(y)\phi(z)$,
- x-y wavelet: $\psi^2(x,y,z) = \psi(x)\psi(y)\phi(z)$,
- y wavelet: $\psi^3(x,y,z) = \phi(x)\psi(y)\phi(z)$,
- y-z wavelet: $\psi^4(x,y,z) = \phi(x)\psi(y)\psi(z)$,
- x-y-z wavelet: $\psi^5(x,y,z) = \psi(x)\psi(y)\psi(z)$,
- x-z wavelet: $\psi^6(x,y,z) = \psi(x)\phi(y)\psi(z)$,
- z wavelet: $\psi^7(x,y,z) = \phi(x)\phi(y)\psi(z)$,

which leads to seven wavelet subcubes (subbands) at each resolution level (see Fig. 2):

$$
\begin{aligned}
w^1_{j+1}[k,l,m] &= \sum_{p,q,r} g[p-2k]h[q-2l]h[r-2m]c_j[p,q,r] = [\bar{g}\bar{h}\bar{h} \star c_j]_{\downarrow 2,2,2}[k,l,m] \\
w^2_{j+1}[k,l,m] &= \sum_{p,q,r} g[p-2k]g[q-2l]h[r-2m]c_j[p,q,r] = [\bar{g}\bar{g}\bar{h} \star c_j]_{\downarrow 2,2,2}[k,l,m] \\
w^3_{j+1}[k,l,m] &= \sum_{p,q,r} h[p-2k]g[q-2l]h[r-2m]c_j[p,q,r] = [\bar{h}\bar{g}\bar{h} \star c_j]_{\downarrow 2,2,2}[k,l,m] \\
w^4_{j+1}[k,l,m] &= \sum_{p,q,r} h[p-2k]g[q-2l]g[r-2m]c_j[p,q,r] = [\bar{h}\bar{g}\bar{g} \star c_j]_{\downarrow 2,2,2}[k,l,m] \\
w^5_{j+1}[k,l,m] &= \sum_{p,q,r} g[p-2k]g[q-2l]g[r-2m]c_j[p,q,r] = [\bar{g}\bar{g}\bar{g} \star c_j]_{\downarrow 2,2,2}[k,l,m] \\
w^6_{j+1}[k,l,m] &= \sum_{p,q,r} g[p-2k]h[q-2l]g[r-2m]c_j[p,q,r] = [\bar{g}\bar{h}\bar{g} \star c_j]_{\downarrow 2,2,2}[k,l,m] \\
w^7_{j+1}[k,l,m] &= \sum_{p,q,r} h[p-2k]h[q-2l]g[r-2m]c_j[p,q,r] = [\bar{h}\bar{h}\bar{g} \star c_j]_{\downarrow 2,2,2}[k,l,m] \ .
\end{aligned}
$$

For a discrete $N \times N \times N$ data cube X, the transform is summarized in Algorithm 1.

In a similar way to the 1D case in (12) and with the proper generalization to 3D, the reconstruction is obtained by

$$
\begin{aligned}
c_j = 8(&\tilde{h}\tilde{h}\tilde{h} \star [c_{j+1}]_{\uparrow 2,2,2} + \tilde{g}\tilde{h}\tilde{h} \star [w^1_{j+1}]_{\uparrow 2,2,2} + \tilde{g}\tilde{g}\tilde{h} \star [w^2_{j+1}]_{\uparrow 2,2,2} \\
&+ \tilde{h}\tilde{g}\tilde{h} \star [w^3_{j+1}]_{\uparrow 2,2,2} + \tilde{h}\tilde{g}\tilde{g} \star [w^4_{j+1}]_{\uparrow 2,2,2} + \tilde{g}\tilde{g}\tilde{g} \star [w^5_{j+1}]_{\uparrow 2,2,2} \\
&+ \tilde{g}\tilde{h}\tilde{g} \star [w^6_{j+1}]_{\uparrow 2,2,2} + \tilde{h}\tilde{h}\tilde{g} \star [w^7_{j+1}]_{\uparrow 2,2,2}) \ .
\end{aligned} \tag{15}
$$

2.2 3D Isotropic Undecimated Wavelet Transform

The main interest of the biorthogonal wavelet transform introduced in the previous section is its non redundancy: the transform of an $N \times N \times N$ cube is a cube of the same size. This property is particularly appreciable in three dimensions as the resources needed to process a 3D signal scale faster than in

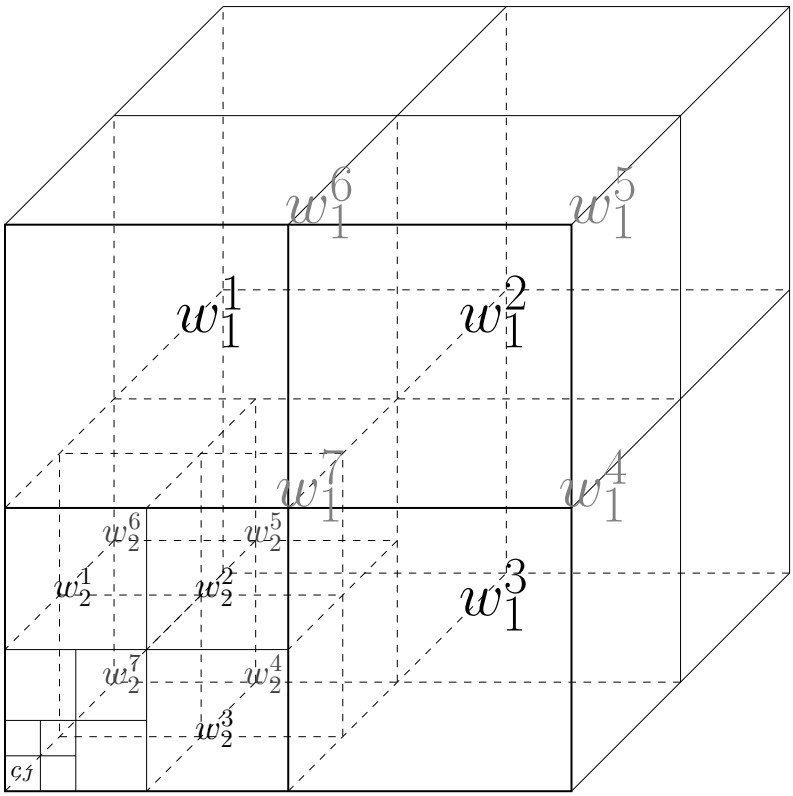


Fig. 2. Decomposition of initial data cube into pyramidal wavelet bands. The bottom left cube c_J is the smoothed approximation and the w_j^i are the different wavelet subbands at each scale j.

lower dimensions. However, this DWT is far from optimal for applications such as restoration (e.g. denoising or deconvolution), detection or more generally, analysis of data. Indeed, modifications of DWT coefficients introduce a large number of artefacts in the signal after reconstruction, mainly due to the loss of the translation-invariance in the DWT.

For this reason, for restoration and detection purposes, redundant transform are generally preferred. Here, we present the 3D version of the Isotropic Undecimated Wavelet Transform (IUWT) also known as the *starlet wavelet transform* because its 2D version is well adapted to the more or less isotropic features found in astronomical data [46, 47].

The starlet transform is based on a separable isotropic scaling function

$$\phi(x, y, z) = \phi_{1D}(x)\phi_{1D}(y)\phi_{1D}(z) \ , \tag{16}$$

where ϕ_{1D} is a 1D B-spline of order 3:

$$\phi_{1D}(x) = \frac{1}{12}\left(|x-2|^3 - 4|x-1|^3 + 6|x|^3 - 4|x+1|^3 + |x+2|^3\right). \tag{17}$$

The separability of ϕ is not a required condition but it allows to have fast

Algorithm 1: The 3D biorthogonal wavelet transform

Data: An $N \times N \times N$ data cube X

Result: $\mathcal{W} = \{w_1^1, w_1^2, ..., w_1^7, w_2^1, ..., w_J^1, ..., w_J^7, c_J\}$ the 3D DWT of X.

begin

$c_0 = X,\ J = \log_2 N$

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

Compute $c_{j+1} = \bar{h}\bar{h}\bar{h} \star c_j$, down-sample by a factor 2 in each dimension.

Compute $w_{j+1}^1 = \bar{g}\bar{h}\bar{h} \star c_j$, down-sample by a factor 2 in each dimension.

Compute $w_{j+1}^2 = \bar{g}\bar{g}\bar{h} \star c_j$, down-sample by a factor 2 in each dimension.

Compute $w_{j+1}^3 = \bar{h}\bar{g}\bar{h} \star c_j$, down-sample by a factor 2 in each dimension.

Compute $w_{j+1}^4 = \bar{h}\bar{g}\bar{g} \star c_j$, down-sample by a factor 2 in each dimension.

Compute $w_{j+1}^5 = \bar{h}\bar{g}\bar{g} \star c_j$, down-sample by a factor 2 in each dimension.

Compute $w_{j+1}^6 = \bar{g}\bar{h}\bar{g} \star c_j$, down-sample by a factor 2 in each dimension.

Compute $w_{j+1}^7 = \bar{h}\bar{h}\bar{g} \star c_j$, down-sample by a factor 2 in each dimension.

computation which is especially important for large scale data sets in three dimensions.

The wavelet function is defined as the difference between the scaling functions of two successive scales:

$$\frac{1}{8}\psi(\frac{x}{2}, \frac{y}{2}, \frac{z}{2}) = \phi(x, y, z) - \frac{1}{8}\phi(\frac{x}{2}, \frac{y}{2}, \frac{z}{2}). \tag{18}$$

This choice of wavelet function will allow for a very simple reconstruction formula where the original data cube can be recovered by simple co-addition of the wavelet coefficients and the last smoothed approximation. Furthermore, since the scaling function is chosen to be isotropic, the wavelet function is therefore also isotropic. Figure 3 shows an example of such 3D isotropic wavelet function.

The implementation of the starlet transform relies on the very efficient *à trous* algorithm, where this French term means "with holes" [48, 49]. Let h be the filter associated to ϕ:

$$h[k, l, m] = h_{1D}[k]h_{1D}[l]h_{1D}[m]\ , \tag{19}$$

$$h_{1D}[k] = [1, 4, 6, 4, 1]/16, \qquad k \in [\![-2, 2]\!]\ , \tag{20}$$

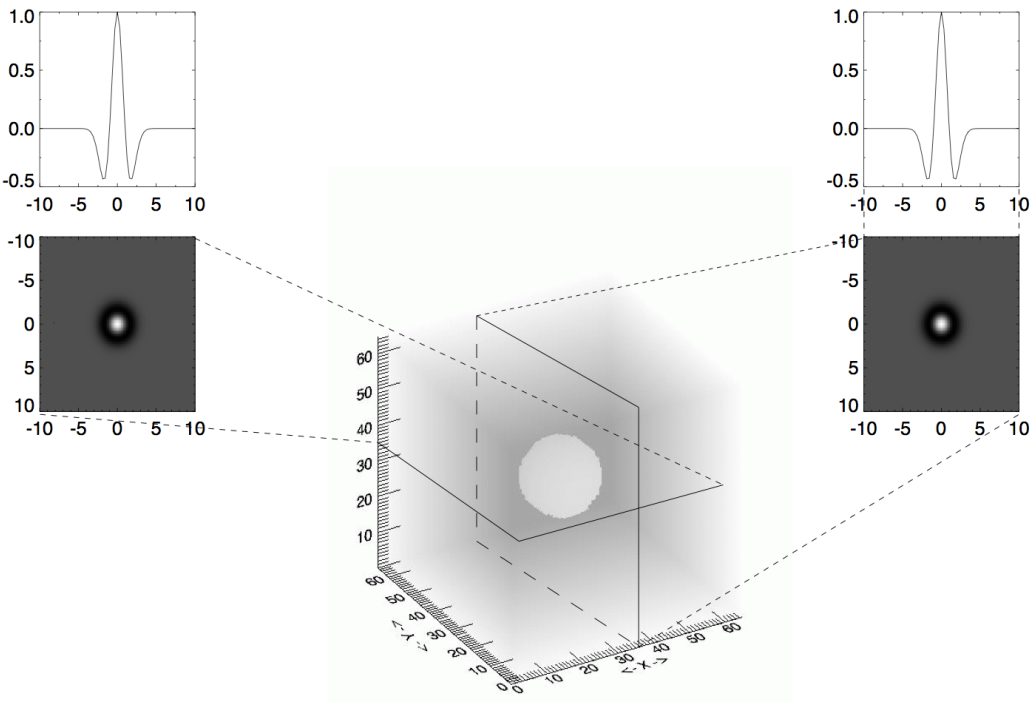


Fig. 3. 3D Isotropic wavelet function.

and g the filter associated to the wavelet ψ:

$$g[k,l,m] = \delta[k,l,m] - h[k,l,m] \ . \tag{21}$$

The à trou algorithm defines for each j a scaled versions $h_{1D}^{(j)}$ of the 1D filter h_{1D} such that:

$$h_{1D}^{(j)}[k] = \begin{cases} h_{1D}[k] & \text{if } k \in 2^j\mathbb{Z} \\ 0 & \text{otherwise} \ . \end{cases} \tag{22}$$

For example, we have

$$h_{1D}^{(1)} = [\ldots, h_{1D}[-2], 0, h_{1D}[-1], 0, h_{1D}[0], 0, h_{1D}[1], 0, h_{1D}[2], \ldots] \ . \tag{23}$$

Due to the separability of h, for each j we can also define

$$h^{(j)}[k,l,m] = h_{1D}^{(j)}[k]h_{1D}^{(j)}[l]h_{1D}^{(j)}[m] \tag{24}$$

$$g^{(j)}[k,l,m] = \delta[k,l,m] - h_{1D}^{(j)}[k]h_{1D}^{(j)}[l]h_{1D}^{(j)}[m] \ . \tag{25}$$

From the original data cube c_0, the wavelet and approximation coefficients

can now be recursively extracted using the filters $h^{(j)}$ and $g^{(j)}$:

$$\begin{aligned} c_{j+1}[k,l,m] &= (\bar{h}^{(j)} \star c_j)[k,l,m] \qquad (26) \\ &= \sum_{p,q,r} h_{1D}[p]h_{1D}[q]h_{1D}[r]c_j[k+2^jp, l+2^jq, m+2^jr] \\ w_{j+1}[k,l,m] &= (\bar{g}^{(j)} \star c_j)[k,l,m] \qquad (27) \\ &= c_j[k,l,m] - \sum_{p,q,r} h_{1D}[p]h_{1D}[q]h_{1D}[r]c_j[k+2^jp, l+2^jq, m+2^jr] \ . \end{aligned}$$

Finally, due to the choice of wavelet function, the reconstruction is obtained by a simple co-addition of all the wavelet scales and the final smooth subband:

$$c_0[k,l,m] = c_J[k,l,m] + \sum_{j=1}^{J} w_j[k,l,m] \ . \qquad (28)$$

The algorithm for the 3D starlet transform is provided in Algorithm 2.

At each scale j, the starlet transform provides only one subband w_j, instead of the 7 subbands produced by the biorthgonal transform. However, since the subbands are not decimated in this transform, each w_j as exactly the same number of voxels as the the input data cube. The redundancy factor of the 3D starlet transform is therefore $J+1$ where J is the number of scales. Although higher than the redundancy factor of the biorthogonal transform (equal to 1), the starlet transform offers a far reduced redundancy compared to a standard Undecimated Wavelet Transform (undecimated version of the DWT introduced in the previous section, see [50]) which would have a redundancy factor of $7J+1$.

2.3 2D-1D Wavelet Transform

So far, the 3D wavelet transforms we have presented are constructed to handle full 3D signals. However, in some situations the signals of interest are not intrinsically 3D but constructed from a set of 2D images where the third dimension is not spatial but can be temporal or in energy. In this case, analysing the data with the previous 3D wavelets makes no sense and a separate treatment of the third dimension, not connected to the spatial domain, is required. One can define an appropriate wavelet for this kind of data by tensor product of a 2D spatial wavelet and a 1D temporal (or energy) wavelet:

$$\psi(x,y,z) = \psi^{(xy)}(x,y)\psi^{(z)}(z) \ . \qquad (29)$$

where $\psi^{(xy)}$ is the spatial wavelet and $\psi^{(z)}$ the temporal wavelet (resp energy). In the following, we will consider only isotropic spatial scale and dyadic scale,

Algorithm 2: 3D Starlet transform algorithm.

Data: An $N \times N \times N$ data cube X

Result: $\mathcal{W} = \{w_1, w_2, ..., w_J, c_J\}$ the 3D starlet transform of X.

begin

$c_0 = X,\ J = \log_2 N, h_{1\mathrm{D}}[k] = [1, 4, 6, 4, 1]/16,\ k = -2, \ldots, 2.$

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

for ***each*** $k, l = 0$ ***to*** $N - 1$ **do**

Carry out a 1D discrete convolution of the cube c_j with periodic or reflexive boundary conditions, using the 1D filter $h_{1\mathrm{D}}$. The convolution is an interlaced one, where the $h_{1\mathrm{D}}^{(j)}$ filter's sample values have a gap (growing with level, j) between them of 2^j samples, giving rise to the name à trous ("with holes").

$$\alpha[k, l, \cdot] = h_{1\mathrm{D}}^{(j)} \star c_j[k, l, \cdot]\ .$$

for ***each*** $k, m = 0$ ***to*** $N - 1$ **do**

Carry out a 1D discrete convolution of α, using 1D filter $h_{1\mathrm{D}}$:

$$\beta[k, \cdot, m] = h_{1\mathrm{D}}^{(j)} \star \alpha[k, \cdot, m].$$

for ***each*** $l, m = 0$ ***to*** $N - 1$ **do**

Carry out a 1D discrete convolution of β, using 1D filter $h_{1\mathrm{D}}$:

$$c_{j+1}[\cdot, l, m] = h_{1\mathrm{D}}^{(j)} \star \beta[\cdot, l, m].$$

From the smooth subband c_j, compute the IUWT detail coefficients,

$$w_{j+1} = c_j - c_{j+1}.$$

and we note j_1 the spatial scale index (i.e. scale $= 2^{j_1}$), j_2 the time (resp energy) scale index,

$$\psi_{j_1,k_x,k_y}^{(xy)}(x, y) = \frac{1}{2^{j_1}} \psi^{(xy)}(\frac{x - k_x}{2^{j_1}}, \frac{y - k_y}{2^{j_1}}) \tag{30}$$

$$\psi_{j_2,k_z}^{(z)}(z) = \frac{1}{\sqrt{2^{j_2}}} \psi^{(z)}(\frac{z - k_z}{2^{j_2}})\ . \tag{31}$$

Hence, given a continuous data set D, we derive its 2D-1D wavelet coefficients $w_{j_1,j_2}(k_x, k_y, k_z)$ (k_x and k_y are spatial indices and k_z is a time (resp energy)

index) according to:

$$
\begin{aligned}
w_{j_1,j_2}(k_x,k_y,k_z) &= \frac{1}{2^{j_1}}\frac{1}{\sqrt{2^{j_2}}}\iiint_{-\infty}^{+\infty} D(x,y,z) \quad \psi^{(z)*}\left(\frac{z-k_z}{2^{j_2}}\right) \\
&\quad \times \psi^{(xy)*}\left(\frac{x-k_x}{2^{j_1}},\frac{y-k_y}{2^{j_1}}\right) dxdydz \\
&= \left\langle D, \psi^{(xy)}_{j_1,k_x,k_y}\psi^{(z)}_{j_2,k_z}\right\rangle \ . \qquad (32)
\end{aligned}
$$

2.3.1 Fast Undecimated 2D-1D Decomposition/Reconstruction

In order to have a fast algorithm, wavelet functions associated to a filter bank are preferred. Given a discrete data cube $D[k,l,m]$ this wavelet decomposition consists in applying first a 2D isotropic wavelet transform for each frame m. Using the 2D version of the Isotropic Undecimated Wavelet Transform described in the previous section, we have:

$$
\forall m, \quad D[\cdot,\cdot,m] = c_{J_1}[\cdot,\cdot,m] + \sum_{j_1=1}^{J_1-1} w_{j_1}[\cdot,\cdot,m] \ , \qquad (33)
$$

where J_1 is the number of spatial scales. Then, for each spatial location $[k,l]$ and for each 2D wavelet scale scale j_1, an undecimated $1D$ wavelet transform along the third dimension is applied on the spatial wavelet coefficients $w_{j_1}[k,l,\cdot]$

$$
\forall k,l, \quad w_j[k,l,\cdot] = w_{j_1,J_2}[k,l,\cdot] + \sum_{j_2=1}^{J_2-1} w_{j_1,j_2}[k,l,\cdot] \ , \qquad (34)
$$

where J_2 is the number of scales along the third dimension. The same processing is also applied on the coarse spatial scale $c_{J_1}[k,l,\cdot]$, and we have:

$$
\forall k,l, \quad c_{J_1}[k,l,\cdot] = c_{J_1,J_2}[k,l,\cdot] + \sum_{j_2=1}^{J_2-1} w_{J_1,j_2}[k,l,\cdot] \ . \qquad (35)
$$

Hence, we have a 2D-1D undecimated wavelet representation of the input data D:

$$
\begin{aligned}
D[k,l,m] = c_{J_1,J_2}[k,l,m] &+ \sum_{j_2=1}^{J_2-1} w_{J_1,j_2}[k,l,m] \\
&+ \sum_{j_1=1}^{J_1-1} w_{j_1,J_2}[k,l,m] + \sum_{j_1=1}^{J_1-1}\sum_{j_2=1}^{J_2-1} w_{j_1,j_2}[k,l,m] \ . \qquad (36)
\end{aligned}
$$

In this decomposition, four kinds of coefficients can be distinguished:

- Detail-Detail coefficient ($j_1 < J_1$ and $j_2 < J_2$).

$$w_{j_1,j_2}[k,l,\cdot] = (\delta - \overline{h}_{1\mathrm{D}}) \star \left(h_{1\mathrm{D}}^{(j_2-1)} \star c_{j_1-1}[k,l,\cdot] - h_{1\mathrm{D}}^{(j_2-1)} \star c_{j_1}[k,l,\cdot] \right) \ .$$

- Approximation-Detail coefficient ($j_1 = J_1$ and $j_2 < J_2$).

$$w_{J_1,j_2}[k,l,\cdot] = h_{1\mathrm{D}}^{(j_2-1)} \star c_{J_1}[k,l,\cdot] - h_{1\mathrm{D}}^{(j_2)} \star c_{J_1}[k,l,\cdot] \ .$$

- Detail-Approximation coefficient ($j_1 < J_1$ and $j_2 = J_2$).

$$w_{j_1,J_2}[k,l,\cdot] = h_{1\mathrm{D}}^{(J_2)} \star c_{j_1-1}[k,l,\cdot] - h_{1\mathrm{D}}^{(J_2)} \star c_{j_1-1}[k,l,\cdot] \ .$$

- Approximation-Appoximation coefficient ($j_1 = J_1$ and $j_2 = J_2$).

$$c_{J_1,J_2}[k,l,\cdot] = h_{1\mathrm{D}}^{(J_2)} \star c_{J_1}[k,l,\cdot] \ .$$

As this 2D-1D transform is fully linear, a Gaussian noise remains Gaussian after transformation. Therefore, all thresholding strategies wich have been developed for wavelet Gaussian denoising are still valid with the 2D-1D wavelet transform. Denoting δ, the thresholding operator, the denoised cube is obtained by:

$$\begin{aligned}\tilde{D}[k,l,m] = c_{J_1,J_2}[k,l,m] + \sum_{j_1=1}^{J_1-1} \delta(w_{j_1,J_2}[k,l,m]) \\ + \sum_{j_2=1}^{J_2-1} \delta(w_{J_1,j_2}[k,l,m]) + \sum_{j_1=1}^{J_1-1} \sum_{j_2=1}^{J_2-1} \delta(w_{j_1,j_2}[k,l,m]) \ . \quad (37)\end{aligned}$$

A typical operator is the hard threshold, i.e. $\delta_T(x) = 0$ is $|x|$ is below a given threshold T, and $\delta_T(x) = x$ is $|x| \geq T$. The threshold T is generally chosen between 3 and 5 times the noise standard deviation [47].

2.4 Application: Time-varying source detection

An application of the 2D-1D wavelets presented in the previous section has been developed in [51] in the context of source detection for the Large Area Telescope (LAT) instrument aboard the Fermi Gamma-ray Space Telescope. Source detection in the high-energy gamma-ray band observed by the LAT is made complicated by three factors: the low fluxes of point sources relative to the celestial foreground, the limited angular resolution and the intrinsic variability of the sources.

The fluxes of celestial gamma rays are low, especially relative to the $\sim$1 m^2 effective area of the LAT (by far the largest effective collecting area ever in the GeV range). An additional complicating factor is that diffuse emission from the

Milky Way itself (which originates in cosmic-ray interactions with interstellar gas and radiation) makes a relatively intense, structured foreground emission. The few very brightest gamma-ray sources provide approximately 1 detected gamma ray per minute when they are in the field of view of the LAT while the diffuse emission of the Milky Way typically provide about 2 gamma rays per second. Furthermore, in this energy band, the gamma-ray sky is quite dynamic, with a large population of sources such as gamma-ray blazars (distant galaxies whose gamma-ray emission is powered by accretion onto supermassive black holes), episodically flaring. The time scales of flares, which can increase the flux by a factor of 10 or more, can be minutes to weeks; the duty cycle of flaring in gamma rays is not well determined yet, but individual blazars can go months or years between flares and in general we will not know in advance where on the sky the sources will be found.

For previous high-energy gamma-ray missions, the standard method of source detection has been model fitting — maximizing the likelihood function while moving trial point sources around in the region of the sky being analyzed. This approach has been driven by the limited photon counts and the relatively limited resolution of gamma-ray telescopes.

Here, we present the different approach adopted in [51] which is based on a non parametric method combining a MutliScale Variance Stabilization Transform (MS-VST) proposed for Poisson data denoising by [52] and a 2D-1D representation of the data. Using the time as the 1D component of the 2D-1D transform, the resulting filtering method is particularly adapted to the rapidly varying time varying low-flux sources in the Fermi LAT data.

Extending the MS-VST developed for the Isotropic Undecimated Wavelet Transform in [52], the 2D-1D MS-VST is implemented by applying a square root Variance Stabilization Transform (VST) $\mathcal{A}_{j_1,j_2}$ to the approximation coefficients c_{j_1,j_2} before computing the wavelet coefficients as the difference of stabilized approximation coefficients. The VST operator $\mathcal{A}_{j_1,j_2}$ is entirely determined by the filter h used in the wavelet decomposition and by the scales j_1, j_2 (see [52] for complete expression).

Plugging the MS-VST into the 2D-1D transform, yields four kinds of coefficients:

- Detail-Detail coefficient ($j_1 < J_1$ and $j_2 < J_2$).

$$w_{j_1,j_2}[k,l,\cdot] = (\delta - \overline{h}_{1\mathrm{D}})\star \\ \left(\mathcal{A}_{j_1-1,j_2-1}\left[h_{1\mathrm{D}}^{(j_2-1)} \star c_{j_1-1}[k,l,\cdot]\right] - \mathcal{A}_{j_1,j_2-1}\left[h_{1\mathrm{D}}^{(j_2-1)} \star c_{j_1}[k,l,\cdot]\right]\right) .$$

- Approximation-Detail coefficient ($j_1 = J_1$ and $j_2 < J_2$).

$$w_{J_1,j_2}[k,l,\cdot] = \mathcal{A}_{J_1,j_2-1}\left[h_{\mathrm{1D}}^{(j_2-1)} \star c_{J_1}[k,l,\cdot]\right] - \mathcal{A}_{J_1,j_2}\left[h_{\mathrm{1D}}^{(j_2)} \star c_{J_1}[k,l,\cdot]\right] \ .$$

- Detail-Approximation coefficient ($j_1 < J_1$ and $j_2 = J_2$).

$$w_{j_1,J_2}[k,l,\cdot] = \mathcal{A}_{j_1-1,J_2}\left[h_{\mathrm{1D}}^{(J_2)} \star c_{j_1-1}[k,l,\cdot]\right] - \mathcal{A}_{j_1-1,J_2}\left[h_{\mathrm{1D}}^{(J_2)} \star c_{j_1-1}[k,l,\cdot]\right] \ .$$

- Approximation-Appoximation coefficient ($j_1 = J_1$ and $j_2 = J_2$).

$$c_{J_1,J_2}[k,l,\cdot] = h_{\mathrm{1D}}^{(J_2)} \star c_{J_1}[k,l,\cdot] \ .$$

All wavelet coefficients are now stabilized, and the noise on all wavelet coefficients w is Gaussian. Denoising is however not straightforward because there is no reconstruction formulae as the stabilizing operators $\mathcal{A}_{j_1,j_2}$ and the convolution operators along (x,y) and z do not commute. To circumvent this difficulty, this reconstruction problem can be solved by defining the multiresolution support [53] from the stabilized coefficients, and by using an iterative reconstruction scheme.

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 $w_{j_1,J_2}[k,l,m]$ is significant, i.e., not due to noise, if its absolute value is larger k, where k is typically between 3 and 5. The multiresolution support will be obtained by detecting at each scale the significant coefficients. The multiresolution support for $j_1 \leq J_1$ and $j_2 \leq J_2$ is defined by:

$$M_{j_1,j_2}[k,l,m] = \begin{cases} 1 & \text{if } w_{j_1,j_2}[k,l,m] \text{ is significant} \\ 0 & \text{if } w_{j_1,j_2}[k,l,m] \text{ is not significant} \ . \end{cases} \tag{38}$$

We denote $\mathcal{W}$ the 2D-1D isotropic wavelet transform, $\mathcal{R}$ the inverse wavelet transform and Y the input data. We want our solution X to reproduce 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 in the 2D-1D MS-VST (i.e. $M\mathcal{W}X = M\mathcal{W}Y$). 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. Therefore the reconstruction can be formulated as a constrained sparsity-promoting minimization problem that can be written as follows

$$\min_X \parallel \mathcal{W}X \parallel_1 \quad \text{subject to} \quad \begin{cases} M\mathcal{W}X = M\mathcal{W}Y \\ X \geq 0 \ , \end{cases} \tag{39}$$

where $\| \, . \, \|_1$ is the ℓ_1-norm playing the role of regularization and is well known to promote sparsity [54]. This problem can be efficiently solved using the hybrid steepest descent algorithm [55, 52], and requires around 10 iterations.

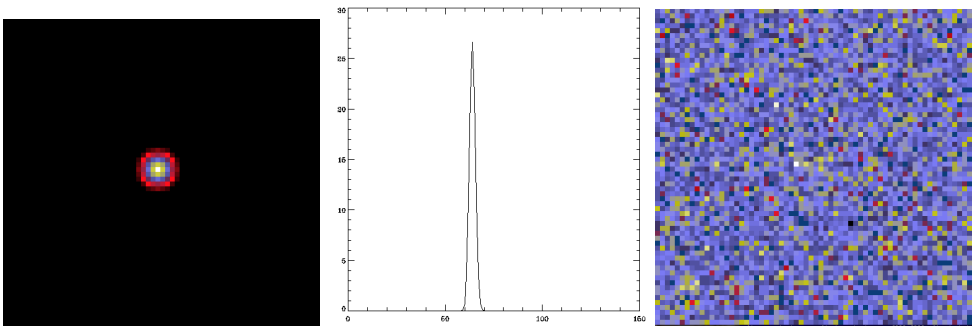


Fig. 4. Simulated time-varying source. From left to right, simulated source, temporal flux, and co-added image along the time axis of noisy data.

This filtering method is tested on a simulated a time varying source in a cube of size $64 \times 64 \times 128$, as a Gaussian centered at $(32, 32, 64)$ with a spatial standard deviation equals to 1.8 (pixel unit) and a temporal standard deviation equal to 1.2. The total flux of the source (i.e. spatial and temporal integration) is 100. A background level of 0.1 is added to the data cube and Poisson noise is generated. Figure 5 shows respectively from left to right an image of the source, the flux per frame and the integration of all frames along the time axis. As it can be seen, the source is hardly detectable in the co-added image.

By running the 2D MS-VST denoising method on the co-added frame, the source cannot be recovered whereas the 2D-1D MS-VST denoising method is able to recover the source at 6σ from the noisy 3D data set. Figure 5 left shows one frame (frame 64) of the denoised cube, and Figure 5 right shows the flux of the recovered source per frame.

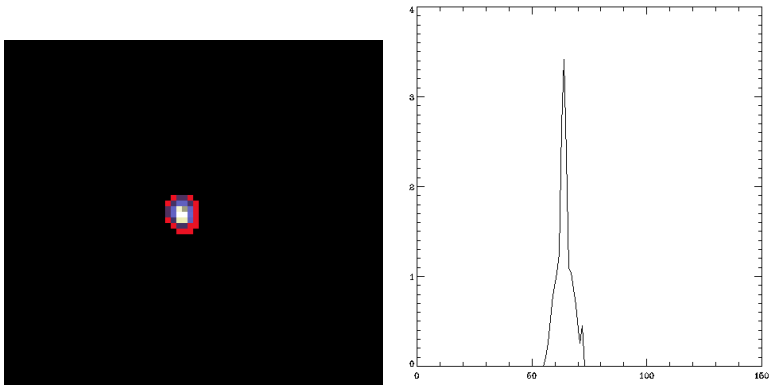


Fig. 5. Revovered time-vaying source after 2D-1D MS-VST denoising. Left, one frame of the denoised cube and right, flux per frame.

3 3D Ridgelets and Beamlets

Wavelets rely on a dictionary of roughly isotropic elements occurring at all scales and locations. They do not describe well highly anisotropic elements, and contain only a fixed number of directional elements, independent of scale. Despite the fact that they have had wide impact in image processing, they fail to efficiently represent objects with highly anisotropic elements such as lines or curvilinear structures (e.g. edges). The reason is that wavelets are non-geometrical and do not exploit the regularity of the edge curve. Following this reasoning, new constructions in 2D have been proposed such as ridgelets [56] and beamlets [57]. Both transforms were developed as an answer to the weakness of the separable wavelet transform in sparsely representing what appears to be simple building-block atoms in an image, that is, lines and edges.

In this section, we present the 3D extension of these transforms. In 3D, the ridgelet atoms are sheets while the beamlet atoms are lines. Both transforms share a similar fast implementation using the projection-slice theorem [58] and will constitute the building blocks of the first generation 3D curvelets presented in Section 4. An application of ridgelets and beamlets to the statistical study of the spatial distribution of galaxies is presented in the last part of this section.

3.1 The 3D Ridgelet Transform

3.1.1 Continuous 3D Ridgelet Transform

The continuous ridgelet transform can be defined in 3D as a direct extension of the 2D transform following [56]. Pick a smooth univariate function $\psi : \mathbb{R} \to \mathbb{R}$ with vanishing mean $\int \psi(t)dt = 0$ and sufficient decay so that it verifies the *3D admissibility* condition:

$$\int |\hat{\psi}(\nu)|^2 |\nu|^{-3} d\nu < \infty \ . \tag{40}$$

Under this condition, one can further assume that ψ is normalized so that $\int |\hat{\psi(\nu)}|^2 |\nu|^{-3} d\nu = 1$. For each scale $a > 0$, each position $b \in \mathbb{R}$ and each orientation $(\theta_1, \theta_2) \in [0, 2\pi[\times[0, \pi[$, we can define a trivariate ridgelet function $\psi_{a,b,\theta_1,\theta_2} : \mathbb{R}^3 \to \mathbb{R}$ by

$$\psi_{a,b,\theta_1,\theta_2}(\mathbf{x}) = a^{-1/2} \psi \left(\frac{x_1 \cos\theta_1 \sin\theta_2 + x_2 \sin\theta_1 \sin\theta_2 + x_3 \cos\theta_2 - b}{a} \right) \ , \tag{41}$$

where $\mathbf{x} = (x_1, x_2, x_3) \in \mathbb{R}^3$. This 3D ridgelet function is now constant along the planes defined by $x_1 \cos\theta_1 \sin\theta_2 + x_2 \sin\theta_1 \sin\theta_2 + x_3 \cos\theta_2 = \text{const}$. However, transverse to these ridges, it is a wavelet.
While the 2D ridgelet transform was adapted to detect lines in an image, the 3D ridgelet transform allows us to detect sheets in a cube.

Given an integrable trivariate function $f \in L_2(\mathbb{R}^3)$, its 3D ridgelet coefficients are defined by:

$$\mathcal{R}_f(a, b, \theta_1, \theta_2) := \langle f, \psi_{a,b,\theta_1,\theta_2} \rangle = \int_{\mathbb{R}^3} f(\mathbf{x}) \psi^*_{a,b,\theta_1,\theta_2}(\mathbf{x}) d\mathbf{x} \ . \tag{42}$$

From these coefficients we have the following reconstruction formula:

$$f(\mathbf{x}) = \int_0^{\pi} \int_0^{2\pi} \int_{-\infty}^{\infty} \int_0^{\infty} \mathcal{R}_f(a, b, \theta_1, \theta_2) \psi_{a,b,\theta_1,\theta_2}(\mathbf{x}) \frac{da}{a^4} db \frac{d\theta_1 d\theta_2}{8\pi^2} \ , \tag{43}$$

which is valid almost everywhere for functions that are both integrable and square integrable. This representation of "any" function as a superposition of 'ridge' functions is furthermore stable as it obeys the following Parseval relation,

$$|f|_2^2 = \int_0^{\pi} \int_0^{2\pi} \int_{-\infty}^{\infty} \int_0^{\infty} |\mathcal{R}_f(a, b, \theta_1, \theta_2)|^2 \frac{da}{a^4} db \frac{d\theta_1 d\theta_2}{8\pi^2}. \tag{44}$$

Just like for the 2D ridgelets, the 3D ridgelet analysis can be constructed as a wavelet analysis in the Radon domain. In 3D, the Radon transform $\mathbf{R}(f)$ of f is the collection of hyperplane integrals indexed by $(\theta_1, \theta_2, t) \in [0, 2\pi[\times[0, \pi[\times\mathbb{R}$ given by

$$\mathbf{R}(f)(\theta_1, \theta_2, t) = \int_{\mathbb{R}^3} f(\mathbf{x}) \delta(x_1 \cos\theta_1 \sin\theta_2 + x_2 \sin\theta_1 \sin\theta_2 + x_3 \cos\theta_2 - t) d\mathbf{x} \ , \tag{45}$$

where $\mathbf{x} = (x_1, x_2, x_3) \in \mathbb{R}^3$ and δ is the Dirac distribution. Then the 3D ridgelet transform is exactly the application of a 1D wavelet transform along the slices of the Radon transform where the plane angle (θ_1, θ_2) is kept constant but t is varying:

$$\mathcal{R}_f(a, b, \theta_1, \theta_2) = \int \psi^*_{a,b}(t) \mathbf{R}(f)(\theta_1, \theta_2, t) dt \ , \tag{46}$$

where $\psi_{a,b}(t) = \psi((t-b)/a)/\sqrt{a}$ is a 1-dimensional wavelet.

Therefore, the basic strategy for calculating the continuous ridgelet transform in 3D is again to compute first the Radon transform $\mathbf{R}(f)(\theta_1, \theta_2, t)$ and second to apply a 1-dimensional wavelet to the slices $\mathbf{R}(f)(\theta_1, \theta_2, \cdot)$.

3.1.2 Discrete 3D Ridgelet transform

A fast implementation of the Radon transform can be proposed in the Fourier domain thanks to the projection-slice theorem. In 3D, this theorem states that the 1D Fourier transform of the projection of a 3D function onto a line is equal to the slice in the 3D Fourier transform of this function passing by the origin and parallel to the projection line.

$$\mathbf{R}(f)(\theta_1, \theta_2, t) = \mathbf{F}_{1D}^{-1}\left(u \in \mathbb{R} \mapsto \mathbf{F}_{3D}(f)(\theta_1, \theta_2, u)\right) \ . \tag{47}$$

The 3D Discrete Ridgelet Transform can be built in a similar way to the RectoPolar 2D transform (see [50]) by applying a Fast Fourier Transform to the data in order to extract lines in the discrete Fourier domain. Once the lines are extracted, the ridgelet coefficient are obtained by applying a 1D wavelet transform along these lines. However, extracting lines defined in spherical coordinates on the Cartesian grid provided by the Fast Fourier Transform is not trivial and requires some kind of interpolation scheme. The 3D ridgelet is summarised in Algorithm 3 and in the flowgraph in Figure 6.

Algorithm 3: The 3D Ridgelet Transform

Data: An $N \times N \times N$ data cube X.
Result: 3D Ridgelet Transform of X
begin
- Apply a 3D FFT to X to yield $\hat{X}[k_x, k_y, k_z]$. ;
- Perform Cartesian-to-Spherical Conversion using an interpolation scheme to sample $\hat{X}$ in spherical coordinates $\hat{X}[\rho, \theta_1, \theta_2]$. ;
- Extract $3N^2$ lines (of size N) passing through the origin and the boundary of $\hat{X}$. ;
- **for *each line*** $[\theta_1, \theta_2]$ **do**
  - apply an inverse 1D FFT ;
  - apply a 1D wavelet transform to get the Ridgelet coefficients ;

3.1.3 Local 3D Ridgelet Transform

The ridgelet transform is optimal to find sheets of the size of the cube. To detect smaller sheets, a partitioning must be introduced [59]. The cube c is decomposed into blocks of lower side-length b so that for a $N \times N \times N$ cube, we count N/b blocks in each direction. After the block partitioning, the tranform is tuned for sheets of size $b \times b$ and of thickness a_j, a_j corresponding to the different dyadic scales used in the transformation.

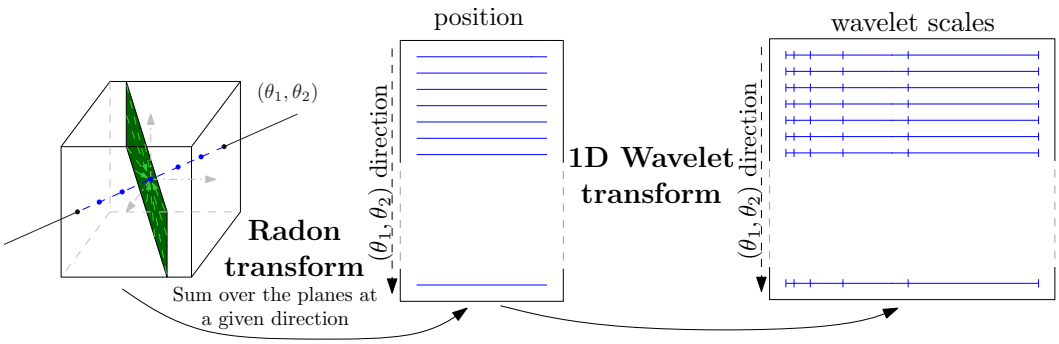


Fig. 6. Overview of the 3D Ridgelet transform. At a given direction, sum over the normal plane to get a • point. Repeat over all its parallels to get the (θ_1, θ_2) line and apply a 1D wavelet transform on it. Repeat for all the directions to get the 3D Ridgelet transform.

3.2 The 3D Beamlet Transform

The X-ray transform $\mathbf{X}f$ of a continuous function $f(x, y, z)$ with $(x, y, z) \in \mathbb{R}^3$ is defined by

$$(\mathbf{X}f)(L) = \int_L f(p)dp \ , \tag{48}$$

where L is a line in $\mathbb{R}^3$, and p is a variable indexing points in the line. The transformation contains all line integrals of f. The Beamlet Transform (BT) can be seen as a multiscale digital X-ray transform. It is a multiscale transform because, in addition to the multiorientation and multilocation line integral calculation, it integrated also over line segments at different length. The 3D BT is an extension to the 2D BT, proposed by Donoho and Huo [57].

The transform requires an expressive set of line segments, including line segments with various lengths, locations and orientations lying inside a 3D volume.

A seemingly natural candidate for the set of line segments is the family of *all* line segments between each voxel corner and every other voxel corner, the set of *3D beams*. For a 3D data set with n^3 voxels, there are $O(n^6)$ 3D beams. It is infeasible to use the collection of 3D beams as a basic data structure since any algorithm based on this set will have a complexity with lower bound of n^6 and hence be unworkable for typical 3D data size.

3.2.1 The Beamlet System

A dyadic cube $C(k_1, k_2, k_3, j) \subset [0, 1]^3$ is the collection of 3D points

$$\{(x_1, x_2, x_3) : [k_1/2^j, (k_1 + 1)/2^j] \times [k_2/2^j, (k_2 + 1)/2^j] \times [k_3/2^j, (k_3 + 1)/2^j]\} \ ,$$

where $0 \leq k_1, k_2, k_3 < 2^j$ for an integer $j \geq 0$, called the scale.

Such cubes can be viewed as descended from the unit cube $C(0,0,0,0) = [0,1]^3$ by recursive partitioning. Hence, the result of splitting $C(0,0,0,0)$ in half along each axis is the eight cubes $C(k_1, k_2, k_3, 1)$ where $k_i \in \{0,1\}$, splitting those in half along each axis we get the 64 subcubes $C(k_1, k_2, k_3, 2)$ where $k_i \in \{0,1,2,3\}$, and if we decompose the unit cube into n^3 voxels using a uniform n-by-n-by-n grid with $n = 2^J$ dyadic, then the individual voxels are the n^3 cells $C(k_1, k_2, k_3, J)$, $0 \leq k_1, k_2, k_3 < n$.

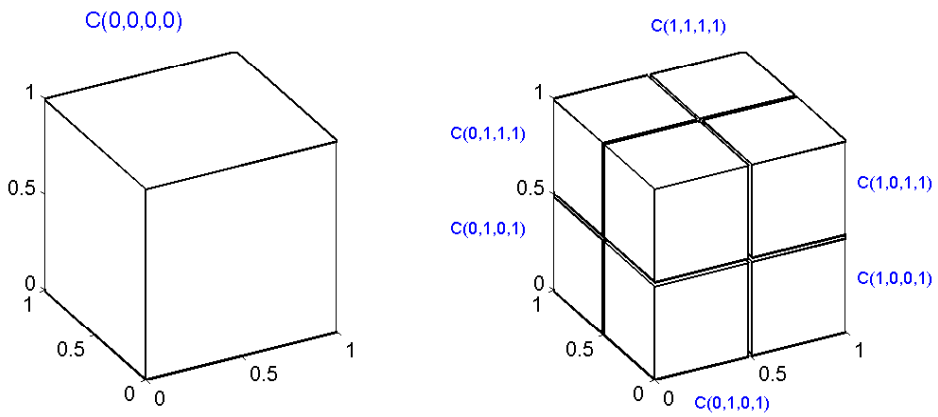


Fig. 7. Dyadic cubes

Associated to each dyadic cube we can build a system of line segments that have both of their end-points lying on the cube boundary. We call each such segment *a beamlet*. If we consider all pairs of boundary voxel corners, we get $O(n^4)$ beamlets for a dyadic cube with a side length of n voxels (we actually work with a slightly different system in which each line is parametrized by a slope and an intercept instead of its end-points as explained below). However, we will still have $O(n^4)$ cardinality. Assuming a voxel size of $1/n$ we get $J+1$ scales of dyadic cubes where $n = 2^J$, for any scale $0 \leq j \leq J$ there are 2^{3j} dyadic cubes of scale j and since each dyadic cube at scale j has a side length of 2^{J-j} voxels we get $O(2^{4(J-j)})$ beamlets associated with the dyadic cube and a total of $O(2^{4J-j}) = O(n^4/2^j)$ beamlets at scale j. If we sum the number of beamlets at all scales we get $O(n^4)$ beamlets.

This gives a multi-scale arrangement of line segments in 3D with controlled cardinality of $O(n^4)$, the scale of a beamlet is defined as the scale of the dyadic cube it belongs to so lower scales correspond to longer line segments and finer scales correspond to shorter line segments. Figure 8 shows 2 beamlets at different scales.

To index the beamlets in a given dyadic cube we use slope-intercept coordinates. For a data cube of $n \times n \times n$ voxels consider a coordinate system with the cube center of mass at the origin and a unit length for a voxel. Hence, for (x, y, z) in the data cube we have $|x|, |y|, |z| \leq n/2$. We can consider three

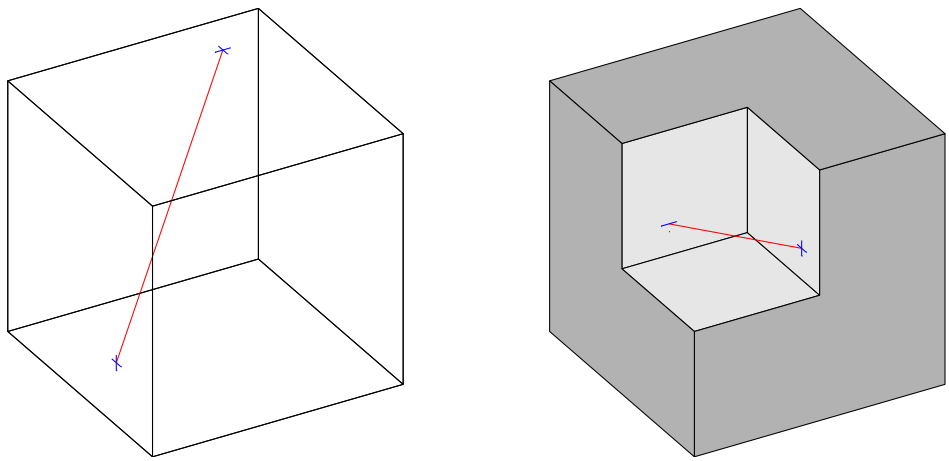


Fig. 8. Examples of Beamlets at two different scales. (a) Scale 0 (coarsest scale) (b) Scale 1 (next finer scale).

kinds of lines: *x-driven, y-driven, and z-driven*, depending on which axis provides the shallowest slopes. An x-driven line takes the form

$$\begin{cases} z = s_z x + t_z \\ y = s_y x + t_y \ , \end{cases} \tag{49}$$

with slopes s_z,s_y, and intercepts t_z and t_y. Here the slopes $|s_z|, |s_y| \leq 1$. y- and z-driven lines are defined with an interchange of roles between x and y or z, as the case may be. The slopes and intercepts run through equispaced sets:

$$\begin{aligned} s_x, s_y, s_z &\in \{2\ell/n : \ell = -n/2, \ldots, n/2 - 1\}, \\ t_x, t_y, t_z &\in \{\ell : -n/2, \ldots, n/2 - 1\}. \end{aligned}$$

Beamlets in a data cube of side n have lengths between $n/2$ and $\sqrt{3}n$ (the main diagonal).

Computational aspects

Beamlet *coefficients* are line integrals over the set of beamlets. A digital 3D image can be regarded as a 3D piece-wise constant function and each line integral is just a weighted sum of the voxel intensities along the corresponding line segment. Donoho and Levi [60] discuss in detail different approaches for computing line integrals in a 3D digital image. Computing the beamlet coefficients for real application data sets can be a challenging computational task since for a data cube with $n \times n \times n$ voxels we have to compute $O(n^4)$ coefficients. By developing efficient cache aware algorithms we are able to handle 3D data sets of size up to $n = 256$ on a typical desktop computer in less than a day running time. We will mention that in many cases there is no interest in the

coarsest scales coefficient that consumes most of the computation time and in that case the over all running time can be significantly faster. The algorithms can also be easily implemented on a parallel machine of a computer cluster using a system such as MPI in order to solve bigger problems.

3.2.2 The FFT-based transformation

Let $\psi \in L_2(\mathbb{R}^2)$ a smooth function satisfying the *admissibility* condition:

$$\int |\hat{\psi}(\boldsymbol{\nu})|^2 |\boldsymbol{\nu}|^{-3} d\boldsymbol{\nu} < \infty \ . \tag{50}$$

In this case, one can further assume that ψ is normalized so that $\int |\hat{\psi}(\boldsymbol{\nu})|^2 |\boldsymbol{\nu}|^{-3} d\boldsymbol{\nu} = 1$. For each scale a, each position $\mathbf{b} = (b_1, b_2) \in \mathbb{R}^2$ and each orientation $(\theta_1, \theta_2) \in [0, 2\pi[\times[0, \pi[$, we can define a trivariate *beamlet* function $\psi_{a,b_1,b_2,\theta_1,\theta_2} : \mathbb{R}^3 \to \mathbb{R}$ by:

$$\begin{aligned}\psi_{a,\mathbf{b},\theta_1,\theta_2}(x_1, x_2, x_3) = a^{-1/2} \cdot \psi((-x_1 \sin\theta_1 + x_2 \cos\theta_1 + b_1)/a, \\ (x_1 \cos\theta_1 \cos\theta_2 + x_2 \sin\theta_1 \cos\theta_2 - x_3 \sin\theta_2 + b_2)/a) \ .\end{aligned} \tag{51}$$

The three-dimensional continuous beamlet transform of a function $f \in L_2(\mathbb{R}^3)$ is given by:

$$\begin{aligned}\mathcal{B}_f : &\mathbb{R}_+^* \times \mathbb{R}^2 \times [0, 2\pi[\times[0, \pi[\to \mathbb{R} \\ &\mathcal{B}_f(a, \mathbf{b}, \theta_1, \theta_2) = \int_{\mathbb{R}^3} \psi^*_{a,\mathbf{b},\theta_1,\theta_2}(\mathbf{x}) f(\mathbf{x}) d\mathbf{x}.\end{aligned} \tag{52}$$

Figure 9 shows an example of beamlet function. It is constant along lines of direction (θ_1, θ_2), and a 2D wavelet function along plane orthogonal to this direction.

The 3D beamlet transform can be built using the "Generalized projection-slice theorem" [58]. Let $f(\mathbf{x})$ be a function on $\mathbb{R}^n$; and let $\mathbf{R}_m f$ denote the m-dimensional partial Radon transform along the first m directions, $m < n$. $\mathbf{R}_m f$ is a function of $(p, \boldsymbol{\mu}_m; x_{m+1}, ..., x_n)$, $\boldsymbol{\mu}_m$ a unit directional vector in $\mathbb{R}^n$ (note that for a given projection angle, the m dimensional partial Radon transform of $f(\mathbf{x})$ has $(n - m)$ untransformated spatial dimensions and a (n-m+1) dimensional projection profile). The Fourier transform of the m dimensional partial radon transform $\mathbf{R}_m f$ is related to $\mathbf{F}f$ the Fourier transform of f by the projection-slice relation

$$\{\mathbf{F}_{n-m+1}\mathbf{R}_m f\}(k, k_{m+1}, ..., k_n) = \{\mathbf{F}f\}(k\boldsymbol{\mu}_m, k_{m+1}, ..., k_n) \ . \tag{53}$$

Since the 3D Beamlet transform corresponds to wavelets applied along planes orthogonal to given directions (θ_1, θ_2), one can use the 2D partial Radon transform to extract planes on which to apply a 2D wavelet transform. Thanks to

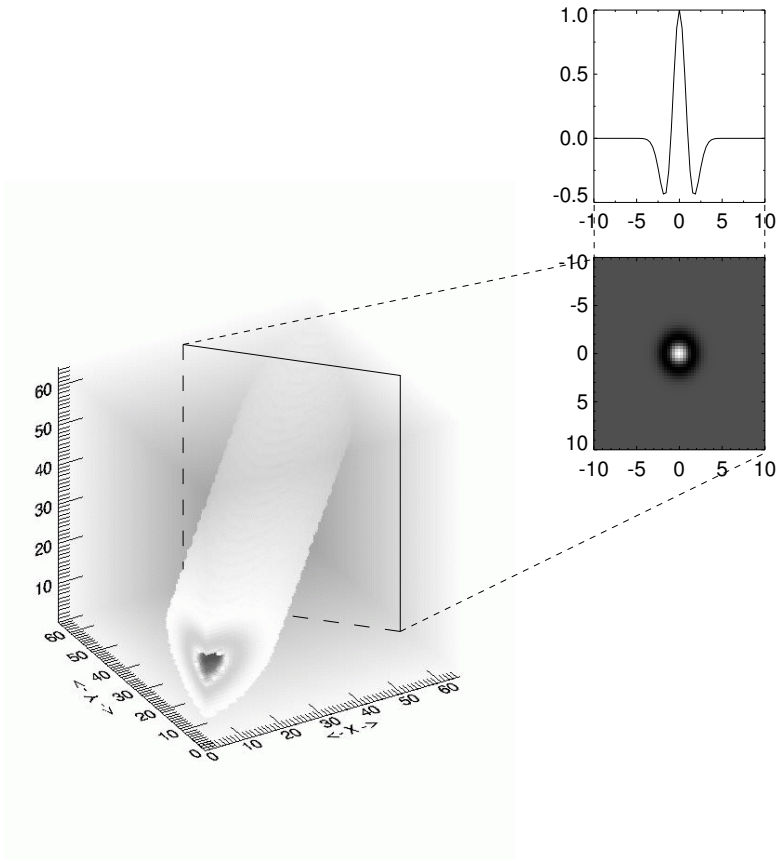


Fig. 9. Example of a beamlet function.

the projection-slice theorem this partial Radon transform in this case can be efficiently performed by taking the inverse 2D Fast Fourier Transforms on planes orthogonal to the direction of the Beamlet extracted from the 3D Fourier space. The FFT based 3D Beamlet transform is summarised in Algorithm 4.

Algorithm 4: The 3D Beamlet Transform

Data: An $N \times N \times N$ data cube X.
Result: 3D Beamlet Transform of X
begin
- Apply a 3D FFT to X to yield $\hat{X}[k_x, k_y, k_z]$. ;
- Perform Cartesian-to-Spherical Conversion using an interpolation scheme to sample $\hat{X}$ in spherical coordinates $\hat{X}[\rho, \theta_1, \theta_2]$. ;
- Extract $3N^2$ planes (of size $N \times N$) passing through the origin, orthogonal to the lines used in the 3D ridgelet transform. ;

for *each plane defined by* $[\theta_1, \theta_2]$ **do**
- apply an inverse 2D FFT ;
- apply a 2D wavelet transform to get the Beamlet coefficients ;

Figure 10 gives the 3D beamlet transform flowgraph. The 3D beamlet transform allows us to detect filaments in a cube. The beamlet transform algorithm presented in this section differs from the one presented in [61]; see the discussion in [60].

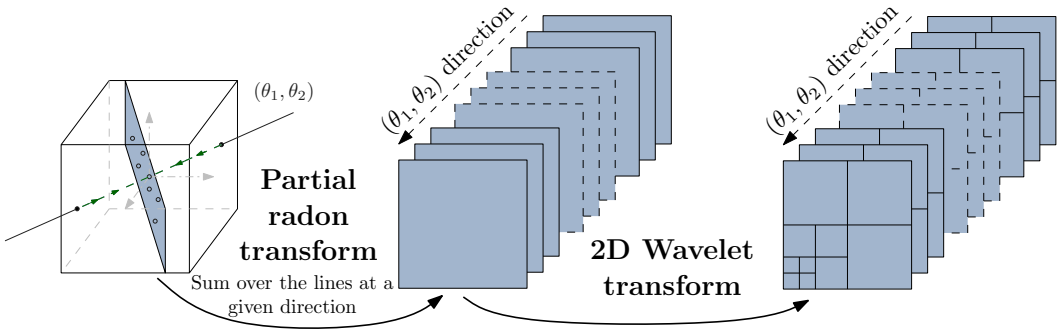


Fig. 10. Schematic view of a 3D Beamlet transform. At a given direction, sum over the (θ_1, θ_2) line to get a $\circ$ point. Repeat over all its parallels to get the dark plane and apply a 2D wavelet transform within that plane. Repeat for all the directions to get the 3D Beamlet transform. See the text (section 4.3) for a detailed explanation and implementation clues.

3.3 Application: Analysis of the Spatial Distribution of Galaxies

To illustrate the two transforms introduced in this section, we present an application of 3D ridgelets and beamlets to the statistical study of the galaxy distribution which was investigated in [62]. Throughout the universe, galaxies are arranged in interconnected walls and filaments forming a cosmic web encompassing huge, nearly empty, regions between the structures. The distribution of these galaxies is of great interest in cosmology as it can be used to constrain cosmological theories. The standard approach for testing different models is to define a point process which can be characterized by statistical descriptors. In order to compare models of structure formation, the different distribution of dark matter particles in N-body simulations could be analyzed as well, with the same statistics.

Many statistical methods have been proposed in the past in order to describe the galaxy distribution and discriminate the different cosmological models. The most widely used statistic if the two-point correlation function $\xi(r)$ which is a primary tool for quantifying large-scale cosmic structure [63].

To go further than the two-point statistics, the 3D Isotropic Undecimated Wavelet Transform (see Section 2.2), the 3D ridgelet transform and the 3D beamlet transform can used to build statistics which measure in a coherent and statistically reliable way, the degree of clustering, filamentarity, sheetedness,

and voidedness of a dataset.

3.3.1 Structure detection

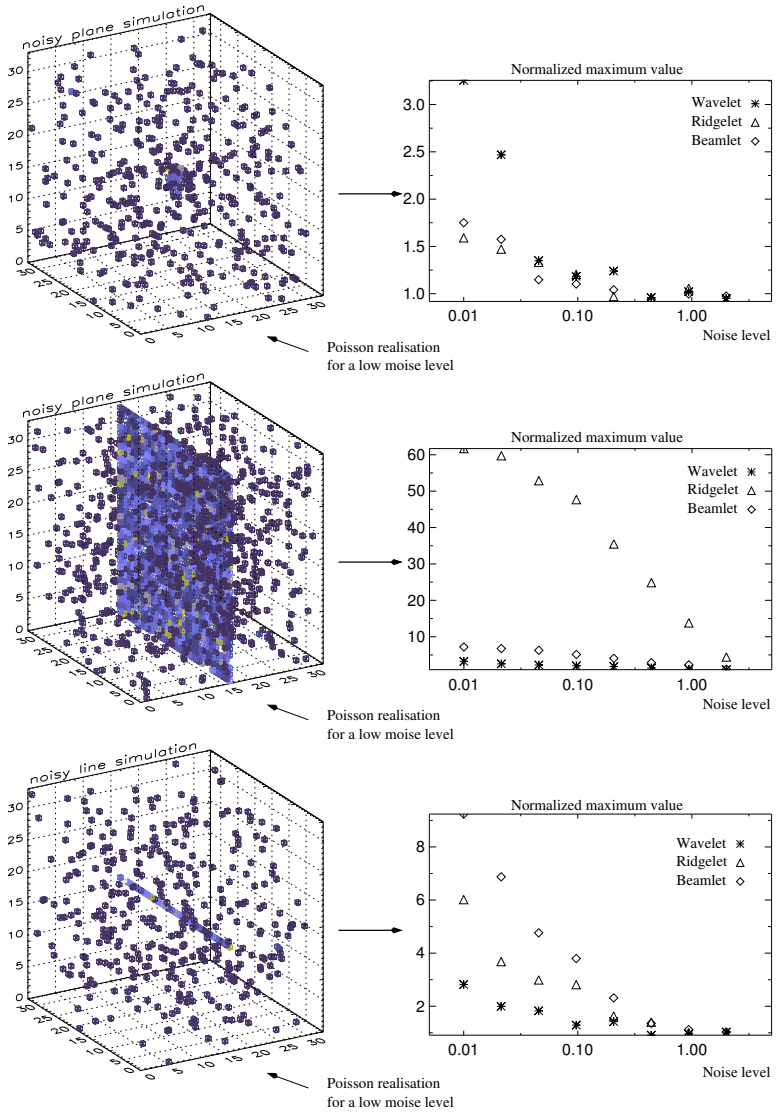


Fig. 11. Simulation of cubes containing a cluster (top), a plane (middle) and a line (bottom).

Three data sets are generated containing respectively a cluster, a plane and a line. To each data set, Poisson noise is added with eight different background levels. After applying wavelets, beamlets and ridgelets to the 24 resulting data sets, the coefficient distribution from each transformation is normalized using twenty realizations of a Poisson noise having the same number of counts as in the data.

Figure 11 shows, from top to bottom, the maximum value of the normalized distribution versus the noise level for our three simulated data set. As

expected, wavelets, ridgelets and beamlets are respectively the best for detecting clusters, sheets and lines. A feature can typically be detected with a very high signal-to-noise ratio in a matched transform, while remaining indetectible in some other transforms. For example, the wall is detected at more than 60σ by the ridgelet transform, but less than 5σ by the wavelet transform. The line is detected almost at 10σ by the beamlet transform, and with worse than 3σ detection level by wavelets. These results show the importance of using several transforms for an optimal detection of all features contained in a data set.

3.3.2 Process discrimination using higher order statistics

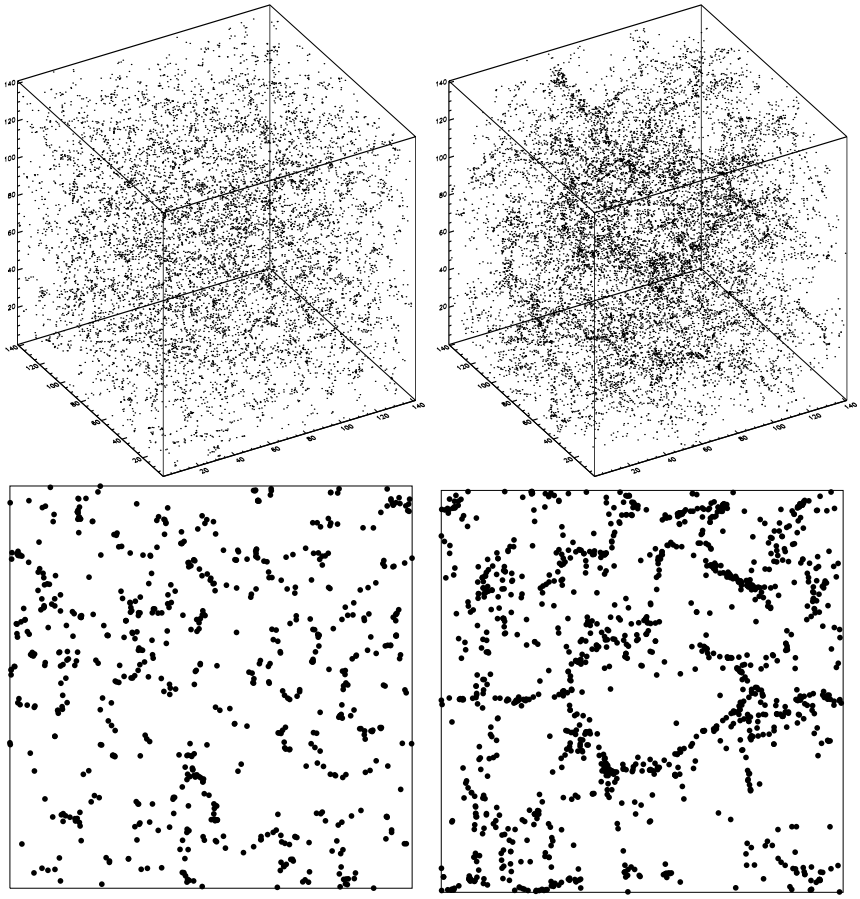


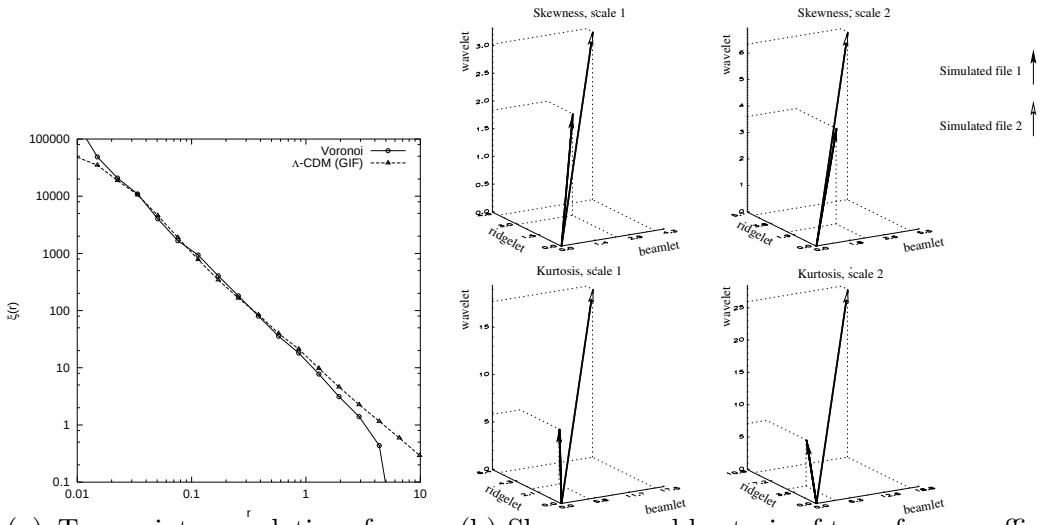
Fig. 12. Simulated data sets. Top, the Voronoi vertices point pattern (left) and the galaxies of the GIF Λ-CDM N-body simulation (right). The bottom panels show one 10 h^{-1} width slice of the each data set.

For this experiment, two simulated data sets are used to illustrate the discriminative power of multiscale methods. The first one is a simulation from stochastic geometry. It is based on a Voronoi model. The second one is a mock catalog of the galaxy distribution drawn from a Λ-CDM N-body cosmological model [64]. Both processes have very similar two-point correlation functions

at small scales, although they look quite different and have been generated following completely different algorithms.

- The first comes from Voronoi simulation: We locate a point in each of the vertices of a Voronoi tessellation of 1.500 cells defined by 1500 nuclei distributed following a binomial process. There are 10085 vertices lying within a box of 141.4 h^{-1} Mpc side.
- The second point pattern represents the galaxy positions extracted from a cosmological Λ-CDM N-body simulation. The simulation has been carried out by the Virgo consortium and related groups [1]. The simulation is a low-density ($\Omega = 0.3$) model with cosmological constant $\Lambda = 0.7$. It is, therefore, an approximation to the real galaxy distribution[64]. There are 15445 galaxies within a box with side 141.3 h^{-1} Mpc. Galaxies in this catalog have stellar masses exceeding 2×10^{10} $M_{\odot}$.

Figure 12 shows the two simulated data sets, and Figure 13(a) left shows the two-point correlation function curve for the two point processes. The two point fields are different, but as can be seen in Figure 13(a), both have very similar two-point correlation functions in a huge range of scales (2 decades).



(a) Two-point correlation function

(b) Skewness and kurtosis of transform coefficients

Fig. 13. The two-point correlation function and skewness and kurtosis of the Voronoi vertices process and the GIF Λ-CDM N-body simulation. The correlation functions are very similar in the range [0.02,2] h^{-1} Mpc while skewness and kurtosis are very different.

After applying the three transforms to each data set, the skewness vector $S = (s_w^j, s_r^j, s_b^j)$ and the kurtosis vector $K = (k_w^j, k_r^j, k_b^j)$ are calculated at each scale j. s_w^j, s_r^j, s_b^j are respectively the skewness at scale j of the wavelet

[1] see http://www.mpa-garching.mpg.de/Virgo

coefficients, the ridgelet coefficients and the beamlet coefficients. k_w^j, k_r^j, k_b^j are respectively the kurtosis at scale j of the wavelet coefficients, the ridgelet coefficients and the beamlet coefficients. Figure 13(b) shows the kurtosis and the skewness vectors of the two data sets at the two first scales. In contrast to the case with the two-point correlation function, this figure shows strong differences between the two data sets, particularly on the wavelet axis, which indicates that the second data contains more or higher density clusters than the first one.

4 First Generation 3D Curvelets

In image processing, edges are curved rather than straight lines and ridgelets are not able to effectively represent such images. However, one can still deploy the ridgelet machinery in a localized way, at fine scales, where curved edges are almost straight lines. This is the idea underlying the first generation 2D curvelets [65]. These curvelets are built by first applying an isotropic wavelet decomposition on the data followed by a local 2D ridgelet transform on each wavelet scale.

In this section we describe a similar construction in the 3D case [20]. In 3D, the 2D ridgelet transform can either be extended using the 3D ridgelets or 3D beamlets introduced in the previous section. Combined with a 3D wavelet transform, the 3D ridgelet gives rise to the RidCurvelet while the 3D beamlet will give rise to BeamCurvelets.

We begin by presenting the frequency-space tiling used by both transforms before describing each one. In the last part of this section, we present denoising applications of these transforms.

4.1 Frequency-space tiling

Following the strategy of the first generation 2D curvelet transform, both 3D curvelets presented in this section are based on a tiling of both frequency space and the unit cube $[0,1]^3$.

Partitioning of the frequency space can be achieved using a filter-bank in order to separate the signal into spectral bands. From an adequate smooth function $\psi \in L_2(\mathbb{R}^3)$ we define for all s in $\mathbb{N}^*$, $\psi_{2s} = 2^{6s}\psi(2^{2s}\cdot)$ which extracts the frequencies around $|\boldsymbol{\nu}| \in [2^{2s}, 2^{2s+2}]$, and a low-pass filter ψ_0 for $|\boldsymbol{\nu}| \leq 1$. We

get a partition of unity in the frequency domain :

$$\forall \boldsymbol{\nu} \in \mathbb{R}^3, \quad |\hat{\psi}_0(\boldsymbol{\nu})|^2 + \sum_{s>0} |\hat{\psi}_{2s}(\boldsymbol{\nu})|^2 = 1 \ . \tag{54}$$

Let $P_0 f = \psi_0 * f$ and $\Delta_s f = \psi_{2s} * f$, where $*$ is the convolution product. We can represent any signal f as $(P_0 f, \Delta_1 f, \Delta_2 f, ...)$.

In the spatial domain, the unit cube $[0,1]^3$ is tiled at each scale s with a finite set $\mathcal{Q}_s$ of $n_s \geq 2^s$ regions Q of size 2^{-s}:

$$Q = Q(s, k_1, k_2, k_3) = \left[\frac{k_1}{2^s}, \frac{k_1+1}{2^s}\right] \times \left[\frac{k_2}{2^s}, \frac{k_2+1}{2^s}\right] \times \left[\frac{k_3}{2^s}, \frac{k_3+1}{2^s}\right] \subset [0,1]^3. \tag{55}$$

Regions are allowed to overlap (for $n_s > 2^s$) to reduce the impact of block effects in the resulting 3D transform. However, the higher the level of overlapping, the higher the redundancy of the final transform. To each region Q is associated a smooth window w_Q so that at any point $\boldsymbol{x} \in [0,1]^3$, $\sum_{Q \in \mathcal{Q}_s} w_Q^2(\boldsymbol{x}) = 1$, with

$$\mathcal{Q}_s = \left\{ Q(s, k_1^i, k_2^i, k_3^i) | \quad \forall i \in [\![0, n_s]\!], (k_1^i, k_2^i, k_3^i) \in [0, 2^s[^3 \right\}. \tag{56}$$

Each element of the frequency-space $w_Q \Delta_s$ is transported to $[0,1]^3$ by the transport operator $T_Q : L_2(Q) \to L_2([0,1]^3)$ applied to $f' = w_Q \Delta_s f$

$$\begin{aligned} T_Q :& L_2(Q) \to L_2([0,1]^3) \\ & (T_Q f')(x_1, x_2, x_3) = 2^{-s} f' \left(\frac{k_1 + x_1}{2^s}, \frac{k_2 + x_2}{2^s}, \frac{k_3 + x_3}{2^s} \right). \end{aligned} \tag{57}$$

For each scale s, we have a space-frequency tiling operator g_Q, the output of which lives on $[0,1]^3$

$$g_Q = T_Q w_Q \Delta_s. \tag{58}$$

Using this tiling operator, we can now build the 3D BeamCurvelet and 3D RidCuvelet transform by respectively applying a 3D Beamlet and 3D Ridgelet transform on each space-frequency block.

4.2 The 3D BeamCurvelet Transform

Given the frequency-space tiling defined in the previous section, a 3D Beamlet transform [17, 66] can now be applied on each block of each scale. Let $\phi \in L_2(\mathbb{R}^2)$ a smooth function satisfying the following admissibility condition

$$\sum_{s \in \mathbb{Z}} \phi^2(2^s \boldsymbol{u}) = 1, \quad \forall \boldsymbol{u} \in \mathbb{R}^2. \tag{59}$$

For a scale parameter $a \in \mathbb{R}$, location parameter $\mathbf{b} = (b_1, b_2) \in \mathbb{R}^2$ and orientation parameters $\theta_1 \in [0, 2\pi[, \theta_2 \in [0, \pi[$, we define $\beta_{a,\mathbf{b},\theta_1,\theta_2}$ the beamlet function (see Section 3.2) based on ϕ :

$$\beta_{a,\mathbf{b},\theta_1,\theta_2}(x_1, x_2, x_3) = a^{-1/2}\phi((-x_1 \sin\theta_1 + x_2 \cos\theta_1 + b_1)/a, \\ (x_1 \cos\theta_1 \cos\theta_2 + x_2 \sin\theta_1 \cos\theta_2 - x_3 \sin\theta_2 + b_2)/a). \quad (60)$$

The BeamCurvelet transform of a 3D function $f \in L_2([0,1]^3)$ is

$$\mathcal{BC} f = \{\langle (T_Q w_Q \Delta_s) f, \beta_{a,\mathbf{b},\theta_1,\theta_2} \rangle : s \in \mathbb{N}^*, Q \in \mathcal{Q}_s\}. \quad (61)$$

As we can see, a BeamCurvelet function is parametrized in scale (s, a), position $(Q, \mathbf{b})$, and orientation (θ_1, θ_2). The following sections describe the discretization and the effective implementation of such a transform.

4.2.1 Discretization

For convenience, and as opposed to the continuous notations, the scales are now numbered from 0 to J, from the finest to the coarsest. As seen in the continuous formulation, the transform operates in four main steps.

(1) First the frequency decomposition is obtained by applying a 3D wavelet transform on the data with a wavelet compactly supported in Fourier space like the pyramidal Meyer wavelets with low redundancy [67], or using the 3D isotropic *à trou* wavelets (see Section 2.2).
(2) Each wavelet scale is then decomposed in small cubes of a size following the parabolic scaling law, forcing the block size B_s with the scale size N_s according to the formula
$$\frac{B_s}{N_s} = 2^{s/2}\frac{B_0}{N_0}, \quad (62)$$
where N_0 and B_0 are the finest scale's dimension and block size.
(3) Then, we apply a partial 3D Radon transform on each block of each scale. This is accomplished by integrating the blocks along lines at every direction and position. For a fixed direction (θ_1, θ_2), the summation gives us a plane. Each point on this plane represents a line in the original cube. We obtain projections of the blocks on planes passing through the origin at every possible angle.
(4) At last, we apply a two-dimensional wavelet transform on each Partial Radon plane.

Steps 3 and 4 represent the Beamlet transform of the blocks. The 3D Beamlet atoms aim at representing filaments crossing the whole 3D space. They are constant along a line and oscillate like ϕ in the radial direction. Arranged blockwise on a 3D isotropic wavelet transform, and following the parabolic

scaling, we obtain the BeamCurvelet transform.
Figure 9 summarizes the beamlet transform, and Figure 14 the global BeamCurvelet transform.

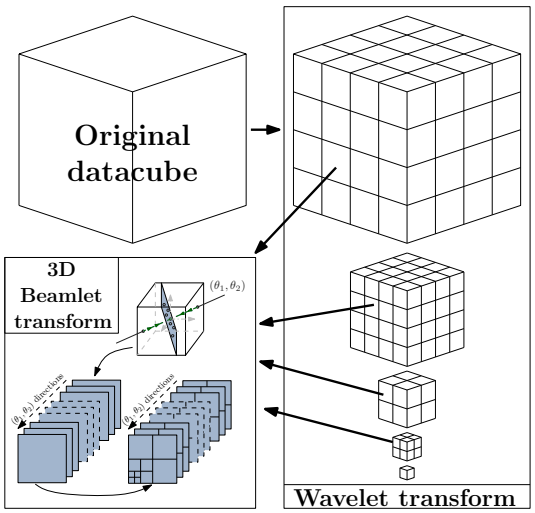


Fig. 14. Global flow graph of a 3D BeamCurvelet transform.

4.2.2 Algorithm summary

As for the 2D Curvelets, the 3D BeamCurvelet transform is implemented effectively in the Fourier domain. Indeed, the integration along the lines (3D partial Radon transform) becomes a simple plane extraction in Fourier space, using the d-dimensional projection-slice theorem, which states that the Fourier transform of the projection of a d-dimensional function onto an m-dimensional linear submanifold is equal to an m-dimensional slice of the d-dimensional Fourier transform of that function through the origin in the Fourier space which is parallel to the projection submanifold. In our case, $d = 3$ and $m = 2$. Algorithm 5 summarizes the whole process.

4.2.3 Properties

As a composition of invertible operators, the BeamCurvelet transform is invertible. As the wavelet and Radon transform are both tight frames, so is the BeamCurvelet transform.

Given a Cube of size $N \times N \times N$, a cubic block of length B_s at scale s, and $J + 1$ scales, the redundancy can be calculated as follows :
According to the parabolic scaling, $\forall s > 0 : B_s/N_s = 2^{s/2} B_0/N_0$. The redun-

Algorithm 5: The BeamCurvelet Transform

Data: A data cube X and an initial block size B
Result: BeamCurvelet transform of X
begin
  Apply a 3D isotropic wavelet transform ;
  for *all scales from the finest to the second coarsest* do
    Partition the scale into small cubes of size B ;
    for *each block* do
      Apply a 3D FFT ;
      Extract planes passing through the origin at every angle (θ_1, θ_2) ;
      for *each plane* (θ_1, θ_2) do
        apply an inverse 2D FFT ;
        apply a 2D wavelet transform to get the BeamCurvelet coefficients ;
    if *the scale number is even* then
      according to the parabolic scaling : ;
      $B = 2B$ (in the undecimated wavelet case) ;
      $B = B/2$ (in the pyramidal wavelet case) ;

dancy induced by the 3D wavelet tansform is

$$R_w = \frac{1}{N^3} \sum_{s=0}^{J} N_s^3, \tag{63}$$

with $N_s = 2^{-s}N$ for pyramidal Meyer wavelets, and thus $B_s = 2^{-s/2}B_0$ according to the parabolic scaling (see equation 62).
The partial Radon transform of a cube of size B_s^3 has a size $3B_s^2 \times B_s^2$ to which we apply 2D decimated orthogonal wavelets with no redundancy. There are $(\rho N_s/B_s)^3$ blocks in each scale because of the overlap factor ($\rho \in [1, 2]$) in each direction. So the complete redundancy of the transform using the Meyer wavelets is

$$R = \frac{1}{N^3} \sum_{s=0}^{J-1} \left(\rho \frac{N_s}{B_s}\right)^3 3B_s^4 + \frac{N_J^3}{N^3} \quad = \quad 3\rho^3 \sum_{i=0}^{J-1} B_s 2^{-3s} + 2^{-3J} \tag{64}$$

$$= 3\rho^3 B_0 \sum_{s=0}^{J-1} 2^{-7s/2} + 2^{-3J} \tag{65}$$

$$= O\left(3\rho^3 B_0\right) \quad \text{when } J \to \infty \tag{66}$$

$$R(J = 1) = 3\rho^3 B_0 + \frac{1}{8} \tag{67}$$

$$R(J = \infty) \approx 3.4\rho^3 B_0 \tag{68}$$

For a typical block size $B_0 = 17$, we get for $J \in [1, \infty[$:

$$R \in [51.125, 57.8[\quad \text{without overlapping} \tag{69}$$
$$R \in [408.125, 462.4[\quad \text{with 50\% overlapping } (\rho = 2). \tag{70}$$

4.2.4 Inverse BeamCurvelet Transform

Because all its components are invertible, the BeamCurvelet transform is invertible and the reconstruction error is comparable to machine precision. Algorithm 6 details the reconstruction steps.

Algorithm 6: The Inverse BeamCurvelet Transform

Data: An initial block size B, and the BeamCurvelet coefficients : series of wavelet-space planes indexed by a scale, angles (θ_1, θ_2), and a 3D position $(\mathcal{B}_x, \mathcal{B}_y, \mathcal{B}_z)$

Result: The reconstructed data cube X

begin

- **for *all scales from the finest to the second coarsest* do**
  - Create a 3D cube the size of the current scale (according to the 3D wavelets used in the forward transform) ;
  - **for *each block position*** $(\mathcal{B}_x, \mathcal{B}_y, \mathcal{B}_z)$ **do**
    - Create a block $\mathcal{B}$ of size $B \times B \times B$;
    - **for *each plane*** (θ_1, θ_2) ***indexed with this position* do**
      - Apply an inverse 2D wavelet transform ;
      - Apply a 2D FFT ;
      - Put the obtained Fourier plane to the block, such that the plane passes through the origin of the block with normal angle (θ_1, θ_2)

      ;
    - Apply a 3D IFFT ;
    - Add the block to the wavelet scale at the position $(\mathcal{B}_x, \mathcal{B}_y, \mathcal{B}_z)$, using a weighted function if overlapping is involved;
  - **if *the scale number is even* then**
    - according to the parabolic scaling : ;
    - $B = 2B$ (in the undecimated wavelet case) ;
    - $B = B/2$ (in the pyramidal wavelet case) ;
- Apply a 3D inverse isotropic wavelet transform ;

An example of a 3D BeamCurvelet atom is represented in Figure 15. The BeamCurvelet atom is a collection of straight smooth segments well localized in space. Across the transverse plane, the BeamCurvelets exhibit a wavelet-like oscillating behavior.

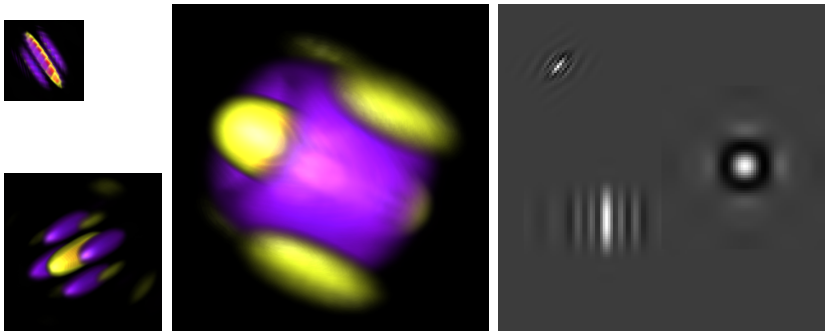


Fig. 15. Examples of a BeamCurvelet atoms at different scales and orientations. These are 3D density plots : the values near zero are transparent, and the opacity grows with the absolute value of the voxels. Positive values are red/yellow, and negative values are blue/purple. The right map is a slice of a cube containing these three atoms in the same position as on the left. The top left atom has an arbitrary direction, the bottom left is in the slice, and the right one is normal to the slice.

4.3 The 3D RidCurvelet Transform

As referred to in 4.2, the second extension of the curvelet transform in 3D is obtained by using the 3D Ridgelet transform [68] defined in Section 3 instead of the Beamlets.

The continuous RidCurvelet is thus defined in much the same way as the BeamCurvelet. Given a smooth function $\phi \in L_2(\mathbb{R})$ verifying the following admissibility condition:

$$\sum_{s\in\mathbb{Z}} \phi^2(2^s u) = 1, \quad \forall u \in \mathbb{R}\ , \tag{71}$$

a three-dimensional ridge function (see Section 3) is given by :

$$\rho_{\sigma,\kappa,\theta_1,\theta_2}(x_1,x_2,x_3) = \sigma^{-1/2}\phi\left(\frac{1}{\sigma}(x_1\cos\theta_1\cos\theta_2 + x_2\sin\theta_1\cos\theta_2 + x_3\sin\theta_2 - \kappa)\right), \tag{72}$$

where σ and κ are respectively the scale and position parameters.

Then the RidCurvelet transform of a 3D function $f \in L_2([0,1]^3)$ is

$$\mathcal{RC}\, f = \{\langle (T_Q w_Q \Delta_s)\, f, \rho_{\sigma,\kappa,\theta_1,\theta_2}\rangle : s \in \mathbb{N}^*, Q \in \mathcal{Q}_s\}\,. \tag{73}$$

4.3.1 Discretization

The discretization is made the same way, the sums over lines becoming sums over the planes of normal direction (θ_1, θ_2), which gives us a line for each direction. The 3D Ridge function is useful for representing planes in a 3D space. It is constant along a plane and oscillates like ϕ in the normal direction. The

main steps of the Ridgelet transform are depicted in figure 6.

4.3.2 Algorithm summary

The RidCurvelet transform is also implemented in Fourier domain, the integration along the planes becoming a line extraction in the Fourier domain. The overall process is shown in Figure 16, and Algorithm 7 summarizes the implementation.

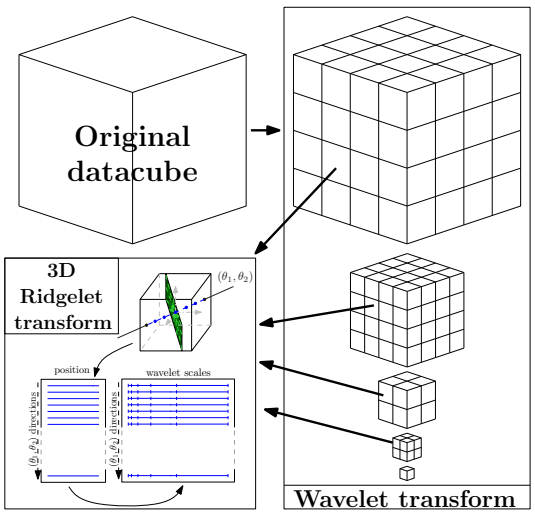


Fig. 16. Global flow graph of a 3D RidCurvelet transform.

4.3.3 Properties

The RidCurvelet transform forms a tight frame. Additionally, given a 3D cube of size $N \times N \times N$, a block of size-length B_s at scale s, and $J+1$ scales, the redundancy is calculated as follows :
The Radon transform of a cube of size B_s^3 has a size $3B_s^2 \times B_s$, to which we apply a pyramidal 1D wavelet of redundancy 2, for a total size of $3B_s^2 \times 2B_s = 6B_s^3$. There are $(\rho N_s/B_s)^3$ blocks in each scale because of the overlap factor ($\rho \in [1,2]$) in each direction. Therefore, the complete redundancy of the transform using many scales of 3D Meyer wavelets is

$$R = \sum_{s=0}^{J-1} 6B_s^3 \left(\rho \frac{N_s}{B_s}\right)^3 + 2^{-3J} \quad = \quad 6\rho^3 \sum_{s=0}^{J-1} 2^{-3s} + 2^{-3J} \tag{74}$$

$$R = O(6\rho^3) \quad \text{when } J \to \infty \ . \tag{75}$$

$$R(J=1) = \quad 6\rho^3 + 1/8 \tag{76}$$

$$R(J=\infty) \approx \quad 6.86\rho^3 \ . \tag{77}$$

Algorithm 7: The RidCurvelet Transform

Data: A data cube x and an initial block size B
Result: RidCurvelet transform of X
begin
  Apply a 3D isotropic wavelet transform ;
  for *all scales from the finest to the second coarsest* do
    Cut the scale into small cubes of size B ;
    for *each block* do
      Apply a 3D FFT ;
      Extract lines passing through the origin at every angle (θ_1, θ_2) ;
      for *each line* (θ_1, θ_2) do
        apply an inverse 1D FFT ;
        apply a 1D wavelet transform to get the RidCurvelet coefficients ;
    if *the scale number is even* then
      according to the parabolic scaling : ;
      $B = 2B$ (in the undecimated wavelet case) ;
      $B = B/2$ (in the pyramidal wavelet case) ;

4.3.4 Inverse RidCurvelet Transform

The RidCurvelet transform is invertible and the reconstruction error is comparable to machine precision. Algorithm 8 details the reconstruction steps.

An example of a 3D RidCurvelet atom is represented in Figure 17. The RidCurvelet atom is composed of planes with values oscillating like a wavelet in the normal direction, and well localized due to the smooth function used to extract blocks on each wavelet scale.

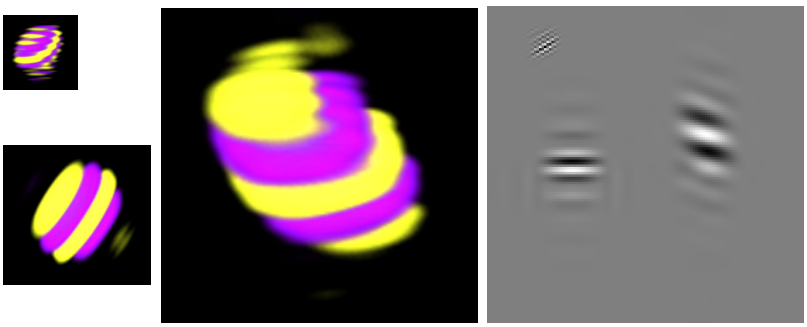


Fig. 17. Examples of RidCurvelet atoms at different scales and orientation. The rendering is similar to that of figure 15. The right plot is a slice from a cube containing the three atoms shown here.

Algorithm 8: The Inverse RidCurvelet Transform

Data: An initial block size B, and the RidCurvelet coefficients : series of wavelet-space lines indexed by a scale, angles (θ_1, θ_2), and a 3D position $(\mathcal{B}_x, \mathcal{B}_y, \mathcal{B}_z)$

Result: The reconstructed data cube X

```
begin
    for all scales from the finest to the second coarsest do
        Create a 3D cube the size of the current scale (according to the 3D
        wavelets used in the forward transform) ;
        for each block position (B_x, B_y, B_z) do
            Create a block B of size B × B × B ;
            for each line (θ_1, θ_2) indexed with this position do
                – Apply an inverse 1D wavelet transform ;
                – Apply a 1D FFT ;
                – Put the obtained Fourier line to the block, such that the line
                passes through the origin of the block with the angle (θ_1, θ_2)
            ;
            – Apply a 3D IFFT ;
            – Add the block to the wavelet scale at the position (B_x, B_y, B_z),
            using a weighted function if overlapping is involved;
        if the scale number is even then
            according to the parabolic scaling : ;
            B = 2B (in the undecimated wavelet case) ;
            B = B/2 (in the pyramidal wavelet case) ;
    Apply a 3D inverse isotropic wavelet transform ;
```

4.4 Application: Structure Denoising

In sparse representations, the simplest denoising methods are performed by a simple thresholding of the discrete curvelet coefficients. The threshold level is usually taken as three times the noise standard deviation, such that for an additive gaussian noise, the thresholding operator kills all noise coefficients except a small percentage, keeping the big coefficients containing information. The threshold we use is often a simple $\kappa\sigma$, with $\kappa \in [3, 4]$, which corresponds respectively to 0.27% and $6.3 \cdot 10^{-5}$ false detections. Sometimes we use a higher κ for the finest scale [3]. Other methods exist, that estimate automatically the threshold to use in each band like the False Discovery Rate (see [69, 70]). The correlation between neighbor coefficients intra-band and/or inter-band may also be taken into account (see [71, 72]). In order to evaluate the different transforms, a $\kappa\sigma$ Hard Thresholding is used in the following experiments.

A way to assess the power of each transform when associated to the right

structures is to denoise a synthetic cube containing plane- and filament-like structures. Figure 18 shows a cut and a projection of the test cube containing parts of spherical shells and a spring-shaped filament. Then this cube is denoised using wavelets, RidCurvelets and BeamCurvelets.

Fig. 18. From left to right : a 3D view of the cube containing pieces of shells and a spring-shaped filament, a slice of the previous cube, and finally a slice from the noisy cube.

As shown in figure 19, the RidCurvelets denoise correctly the shells but poorly the filament, the BeamCurvelets restore the helix more properly while slightly underperforming for the shells, and wavelets are poor on the shell and give a dotted result and misses the faint parts of both structures. The PSNRs obtained with each transform are reported in Table 1. Here, the Curvelet transforms did very well for a given kind of features, and the wavelets were better on the signal power. In the framework of 3D image denoising, it was advocated in [2] to combine several transforms in order to benefit from the advantages of each of them.

Fig. 19. From left to right : a slice from the filtered test-cube (orignial in figure 18) by the wavelet transform (isotropic undecimated), the RidCurvelets and the BeamCurvelets.

	Wavelets	RidCurvelets	BeamCurvelets
Shells & spring	40.4dB	40.3dB	43.7dB

Table 1
PSNR of the denoised synthetic cube using wavelets, RidCurvelets or BeamCurvelets

5 Fast Curvelets

Despite their interesting properties, the first generation curvelet constructions presents some drawbacks. In particular, the spatial partitioning uses overlapping windows to avoid blocking effects. This leads to an increased redundancy of the transforms which is a crucial factor in 3D. In contrast, the second generation curvelets [73, 74], exhibit a much simpler and natural indexing structure with three parameters: scale, orientation (angle) and location, hence simplifying mathematical analysis. The second generation curvelet transform also implements a tight frame expansion [73] and has a much lower redundancy. Unlike the first generation, the discrete second generation implementation will not use ridgelets yielding a faster algorithm [73, 74].

The 3D implementation of the fast curvelets was proposed in [21, 75] with a public code distributed (including the 2-D version) in Curvelab, a C++/Matlab toolbox available at `www.curvelet.org`. This 3D fast curvelet transform has found applications mainly in seismic imaging, for instance for denoising [76] and inpainting [77]. However, a major drawback of this transform is its high redundancy factor, of approximately 25. As a straightforward and somewhat naive remedy to this problem, the authors in [21, 75] suggest to use wavelets at the finest scale instead of curvelets, which indeed reduces the redundancy dramatically to about 5.4 (see Section 5.3 for details). However, this comes at the price of the loss of directional selectivity of fine details. On the practical side, this entails poorer performance in restoration problems compared to the full curvelet version. Note that directional selectivity was one of the main reasons curvelets were built at the first place.

In this section, we begin by describing the original 3D Fast Curvelet transform [21, 75]. The FCT of a 3D object consists of a low-pass approximation subband, and a family of curvelet subbands carrying the curvelet coefficients indexed by their scale, position and orientation in 3D. These 3D FCT coefficients are formed by a proper tiling of the frequency domain following two steps (see Figure 22):

- Cartesian coronization or multiscale separation: first decompose the object into (Cartesian) dyadic coronae in the Fourier domain based on concentric cubes;
- Angular separation: each corona is separated into anisotropic wedges of

trapezoidal shape obeying the so-called parabolic scaling law (to be defined shortly). The 3D FCT coefficients are obtained by an inverse Fourier transform of applied to each wedge appropriately wrapped to fit into a 3D rectangular parallelepipeds.

After detailing these two steps, we express the redundancy factor of the original 3D FCT which will motivate the Low-Redundancy implementation [78] presented afterwards.

In the last part of this section, we present a few application of the 3D Fast Curvelet transform.

5.1 Cartesian coronization

The multiscale separation is achieved using a 3D Meyer wavelet transform [67, 79], where the Meyer wavelet and scaling functions are defined in Fourier domain with compactly supported Fourier transforms.

Let's denote ψ_j as the Meyer wavelet at scale $j \in \{0, \cdots, J-1\}$, and ϕ_{J-1} the scaling function at the coarsest scale. The Meyer wavelets $\hat{\psi}(\xi)$ are defined in Fourier domain as follows :

$$\hat{\psi}(\xi) = \begin{cases} \exp^{-i2\pi\xi} \sin(\frac{\pi}{2}\nu(6|\xi|-1)), & \text{if } 1/6 < |\xi| \leq 1/3 \\ \exp^{-i2\pi\xi} \cos(\frac{\pi}{2}\nu(3|\xi|-1)), & \text{if } 1/3 < |\xi| \leq 2/3 \\ 0 & \text{elsewhere} \end{cases},$$

where ν is a smooth function, that goes from 0 to 1 on $[0, 1]$ and satisfies $\nu(x) + \nu(1-x) = 1$. Associated to this wavelet is the Meyer scaling functions defined by

$$\hat{\phi}(\xi) = \begin{cases} 1, & \text{if } |\xi| \leq 1/6 \\ \cos(\frac{\pi}{2}\nu(6|\xi|-1)), & \text{if } 1/6 < |\xi| \leq 1/3 \\ 0 & \text{if } |\xi| > 1/3 \end{cases}.$$

Figure 20 displays in solid lines the graphs of the Fourier transforms of the Meyer scaling and wavelet functions at three scales.

There is a pair of conjugate mirror filters (h, g) associated to (ϕ, ψ) whose Fourier transforms $(\hat{h}, \hat{g})$ can be easily deduced from $(\hat{\phi}, \hat{\psi})$. $\hat{h}$ and $\hat{g}$ are thus compactly supported. As a consequence, the Meyer wavelet transform is usually implemented in the Fourier domain by a classical cascade of multiplications by $\hat{h}$ and $\hat{g}$. However, the wavelet at the finest scale is supported on

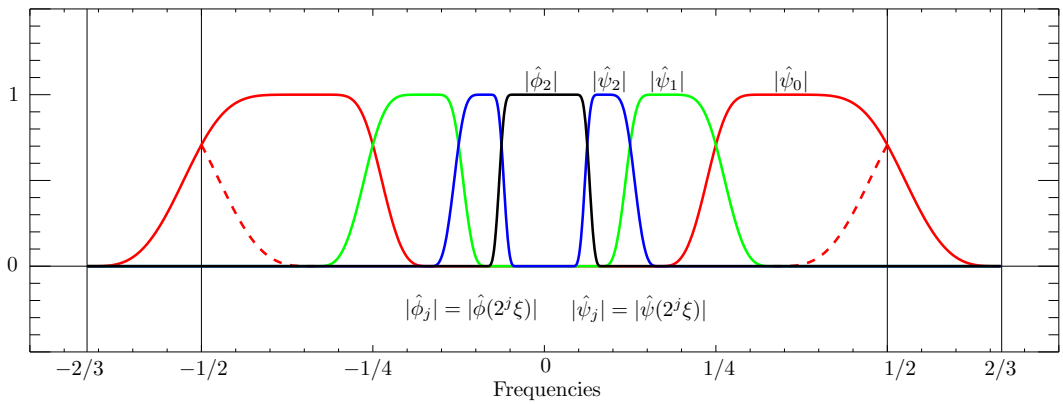


Fig. 20. Meyer scaling and wavelets functions in Fourier domain. In the discrete case, we only have access to the Fourier samples inside the Shannon band $[-1/2, 1/2]$, while the wavelet corresponding to the finest scale (solid red line) exceeds the Shannon frequency band to 2/3. In the original Fast Curvelet implementation, the Meyer wavelet basis is periodized in Fourier, so that the exceeding end of the finest scale wavelet is replaced with the mirrored dashed line on the plot.

$[-2/3, -1/6[\cup]1/6, 2/3]$, hence exceeding the Shannon band. This necessitates to know signal frequencies that we do not have access to.

As the FCT makes central use of the FFT, it implicitly assumes periodic boundary conditions. Moreover, it is known that computing the wavelet transform of a periodized signal is equivalent to decomposing the signal in a periodic wavelet basis. With this in mind, the exceeding end of the finest scale wavelet is replaced with its mirrored version around the vertical axis at $|\xi| = 1/2$, as shown in dashed line on Figure 20. Consequently, the support of the data to treat is 4/3 larger than the original one, hence boosting the redundancy by a factor $(4/3)^d$ in d-D.

Denote $M_j = \hat{\psi}_j = 2^{-3j/2}\hat{\psi}(2^{-j}\cdot)$ and $M_J = \hat{\phi}_{J-1} = 2^{-3(J-1)/2}\hat{\phi}(2^{-(J-1)}\cdot)$ their Fourier transforms. M_J is a lowpass and the wavelet functions $\{M_j\}_{0\leq j<J}$ is a family of bandpass frequency localized windows that form a uniform partition of the unity. Applied to a 3D object, the family $\{M_j\}_{0\leq j<J}$ separates it into Cartesian coronae (annuli), and M_J selects its low frequency content (see Figure 21). This coarsest subband is kept unaltered, and after an inverse Fourier transform, provides us with the first curvelet coefficients, which corresponds to coarse scale isotropic atoms. Only the next detail scales, i.e. those corresponding to $\{M_j\}_{0\leq j<J}$, have to be processed further.

5.2 Angular separation

The FCT isolates, in the frequency domain, oriented and localized 3D wedges. There is a symmetry on a 3D Cartesian grid : the cube has six faces which

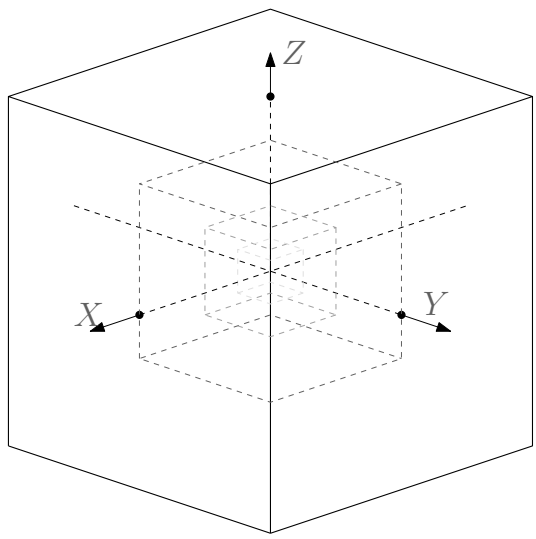


Fig. 21. Cartesian coronization in Fourier space using compactly supported Meyer wavelets for $J = 2$. The central cube corresponds to the isotropic coarsest subband M_J.

can be processed in a similar way. Let $\boldsymbol{\omega} = (\omega_1, \omega_2, \omega_3) \in [-1/2, 1/2]^3$ be a frequency in the 3D Shannon band. Exploiting the above symmetry, we will only focus on the subspace $\{\omega_1 > 0, |\omega_2/\omega_1| < 1, |\omega_3/\omega_1| < 1\}$ which is a squared-based pyramid. The five other parts can be dealt with exactly in the same way by symmetry around the origin and exchange of axes.

Let N_a be the number of angles on one edge of one face of the finest scale, for a total of N_a^2 angles on each face, and thus six times more bands for the entire considered scale (see Figure 22). This number varies with the scale because the number of angles decreases with the scales becoming coarser to obey the parabolic scaling law of curvelets [75], with a parabolic scaling matrix $\mathrm{diag}(2^{-j}, 2^{-j/2}, 2^{-j/2})$ (one short direction and two long ones). This property is essential to ensure that the 3D curvelets are a basis for sparsely representing smooth trivariate functions with 2-D smooth surface-like singularities.

The vector indexing the angular locations on a face at the jth scale may be expressed with $\mathbf{l} = (l, l') \in \{0, \cdots, 2^{\lfloor -j/2 \rfloor} N_a - 1\}^2$, where $\lfloor \cdot \rfloor$ is the integer part of its argument [2]. Recall that a wedge is the trapezoidal region sharply localized along a given angle at a given scale, see the dark gray area in Figure 22. The center of the wedge is on the line going from the origin to the point $(1, \theta_l, \theta'_{l'})$, with

$$\theta_l = \left(-1 + \frac{2l+1}{2^{\lfloor -j/2 \rfloor} N_a}\right), \theta'_{l'} = \left(-1 + \frac{2l'+1}{2^{\lfloor -j/2 \rfloor} N_a}\right). \tag{78}$$

[2] On a Cartesian grid, the slopes are equispaced not the angles.

We can now define the angular separation by multiplying the dyadic annuli corresponding to the wavelet detail subbands by the smooth angular windows $V_{j,\mathbf{l}}$ in the Fourier domain. The angular windows are built from a smooth real-valued function V supported on $[-1,1]$ and satisfying the partition property

$$\sum_{l=-\infty}^{\infty} V^2(t-2l) = 1 \quad \forall t \in \mathbb{R}. \tag{79}$$

The angular window at scale j and orientation $\mathbf{l} = (l, l')$ is then constructed as

$$V_{j,\mathbf{l}}(\boldsymbol{\omega}) = V\left(2^{\lfloor -j/2 \rfloor} N_a \frac{\omega_2 - \theta_l \omega_1}{\omega_1}\right) \cdot V\left(2^{\lfloor -j/2 \rfloor} N_a \frac{\omega_3 - \theta'_{l'} \omega_1}{\omega_1}\right), \tag{80}$$

where θ_l and $\theta'_{l'}$ are defined in (78). Note the scaling factor $2^{-j/2}$ as dictated by the parabolic scaling. Owing to (79), the family of angular windows $\{V_{j,\mathbf{l}}\}_{\mathbf{l}}$ makes a uniform partition of the dyadic annulus at scale j, i.e.

$$\sum_{\mathbf{l}} V_{j,\mathbf{l}}^2(\boldsymbol{\omega}) = 1 \ . \tag{81}$$

However, because of the support constraint on V, this relation does not hold for all $\boldsymbol{\omega}$, and a special care should be taken at the corners where only three out of usually four windows overlap. We thus need to redefine them for (81) to hold for any $\boldsymbol{\omega}$. Here is a simple remedy to this problem. Let $\mathbf{l}_a, \mathbf{l}_b, \mathbf{l}_c$ be the indices of the three corner windows, we redefine them on their overlapping domain Ω as

$$\forall \boldsymbol{\omega} \in \Omega, \forall \mathbf{l} \in \{\mathbf{l}_a, \mathbf{l}_b, \mathbf{l}_c\}, \quad V_{j,\mathbf{l}}(\boldsymbol{\omega}) \leftarrow \frac{V_{j,\mathbf{l}}(\boldsymbol{\omega})}{\sqrt{V_{j,\mathbf{l}_a}^2(\boldsymbol{\omega}) + V_{j,\mathbf{l}_b}^2(\boldsymbol{\omega}) + V_{j,\mathbf{l}_c}^2(\boldsymbol{\omega})}}. \tag{82}$$

Piecing all ingredients together, the *scale-angular* wedge at scale-orientation $(j, \mathbf{l})$ is extracted by the frequency window

$$W_{j,\mathbf{l}}(\boldsymbol{\omega}) = M_j(\boldsymbol{\omega}) \cdot V_{j,\mathbf{l}}(\boldsymbol{\omega}) \ , \tag{83}$$

which is sharply localized near the trapezoid

$$\left\{(\omega_1, \omega_2, \omega_3) : 2^{j+1} < \omega_1 < 2^j, \left|\frac{\omega_2}{\omega_1} - \theta_l\right| < 2^{\lfloor j/2 \rfloor}/N_a, \left|\frac{\omega_3}{\omega_1} - \theta'_{l'}\right| < 2^{\lfloor j/2 \rfloor}/N_a\right\}.$$

Once a wedge is extracted, an inverse Fourier transform must be applied in order to get the curvelet coefficients at the corresponding scale and orientation. Prior to this, the trapezoidal wedge has to be transformed to a convenient form for which the 3D FFT algorithm applies. As it can be seen from Figure 22, the wedge can be inscribed inside a 3D parallelepiped which is $\sim 2^{j/2}$ long on the (ω_2, ω_3) coordinates (i.e. tangentially), and 2^j on ω_1, i.e. radially. Although this expands the area including the wedge, we can still wrap it inside a rectangular

parallelepiped of dimensions $\sim (2^j, 2^{j/2}, 2^{j/2})$ centered at the origin aligned with the axes of the grid (see Section 5.4.2 for further details about wrapping). With appropriate choice of the size of the rectangular parallelepiped, the data does not overlap with itself after wrapping. With the wrapping trick, an inverse 3D FFT can be readily applied to the rectangular parallelepiped to obtain the curvelet coefficients at the selected scale and orientation.

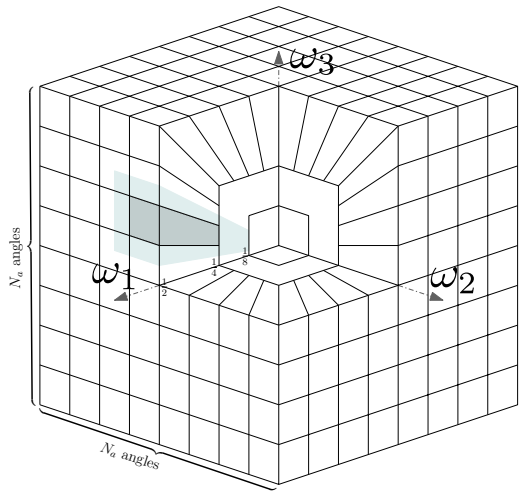


Fig. 22. Example in 3D of how the number of directions N_a is defined along an edge, (eight in this example), and showing the overlapping region around one wedge. The dark gray area is the main part of the wedge, and the light one represents the overlapping region.

Algorithm 9 summarizes the implementation of the 3D FCT and outlines its main steps. In order to show an atom of the 3D FCT, we set to zero all the FastCurvelet coefficients except one, and then perform the inverse transform. We obtain a single FastCurvelet atom, which can be observed in the Fourier domain as well. Figure 23 displays a 3D curvelet atom in the spatial and Fourier domain.

Algorithm 9: The 3D Fast Curvelet Transform

Data: A 3D data object X of size $\mathbf{N} = (N_x, N_y, N_z)$.

Input: Number of scales J, number of angles N_a on each face at the finest scale.

begin

(1) **Multiscale separation:** apply the 3D Meyer wavelet transform in Fourier domain, get cubes of sizes $\mathbf{N}$, $\mathbf{N}/2$, ..., $\mathbf{N}/2^J$.;

(2) **Angular separation:**;

foreach *scale* $j = 0$ ***to*** J **do**

foreach *orientation* $\mathbf{l} = (l, l')$ **do**

Multiply the wavelet cube at scale j with the angular window $V_{j,\mathbf{l}}$ in Fourier;;

Wrapping: wrap the result in a rectangular parallelepiped centered at the origin of minimal size $(2^j 3/8 \times 2^{j/2+1}/N_a \times 2^{j/2+1}/N_a)$;;

Apply a 3D inverse FFT to the rectangular parallelepiped to collect the curvelet coefficients ;;

Result: 3D FCT of X.

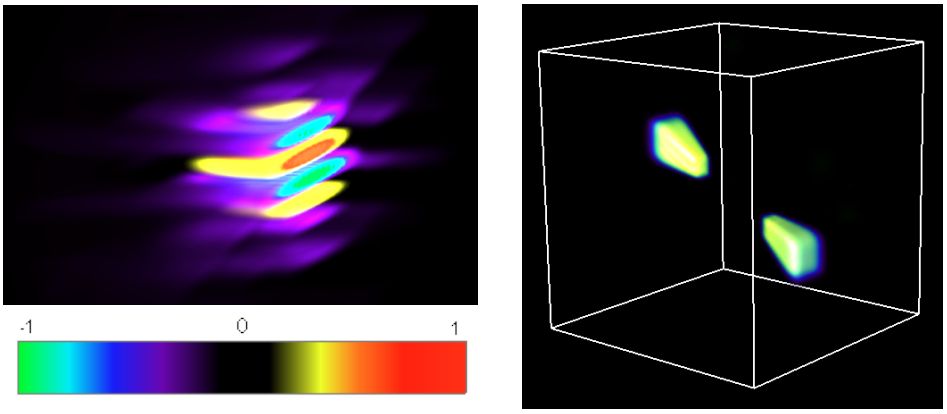


Fig. 23. Left : volume rendering of a 3D curvelet atom in the spatial domain corresponding to our implementation, cut by a vertical plane to see its inner structure. Right : the magnitude if its Fourier transform. The colorbar scale is valid only for the left image.

5.3 Redundancy

Here, we quantify analytically the redundancy of the FCT in any dimension d. Without loss of generality, we assume that the data object is a d-D hypercube of unit side. Let N_a be the number of angles along an edge on one face at the finest scale, for a total of N_a^{d-1} orientations on each face. Let $N_f = 2d$ be the number of faces of the d-D data hypercube.

The redundancy of the Cartesian coronization or multiscale separation assuming a dyadic frequency tiling is given by

$$\sum_{j=0}^{J} \left(\frac{1}{2^d}\right)^j, \tag{84}$$

which is upper-bounded by $R_w = \frac{2^d}{2^d-1}$. As explained in Section 5.1, an extra-redundancy R_{add} comes into play in the Meyer wavelet transform with the original FCT implementation:

$$R_{add} = \left(\frac{4}{3}\right)^d. \tag{85}$$

At the finest scale and on each face, there are N_a^{d-1} wedges, where the size of each of them is

$$\frac{3}{8} \times \underbrace{\frac{2}{N_a} \times ... \times \frac{2}{N_a}}_{d-1 \text{ times}}. \tag{86}$$

The factor 3/8 corresponds to the radial depth of the scale; see Section 5.1 and Figure 20. In the other orthogonal directions, a wedge has a size of $\frac{1}{N_a}$ which we double because of overlapping. The redundancy of a face at the finest scale is then

$$R_f = N_a^{d-1} \cdot \frac{3}{8}\left(\frac{2}{N_a}\right)^{d-1} = 3 \cdot 2^{d-4}. \tag{87}$$

As it can be seen, the R_f redundancy is independent of N_a, and is therefore valid at all scales. For a large enough number of scales, it can be reasonably assumed that coarsest (wavelet) scale has the same redundancy as a curvelet subband at the same scale. Consequently, the overall redundancy of the FCT is upper-bounded by (see (84))

$$\begin{aligned} R &= N_f \cdot R_f \cdot R_w \cdot R_{add} \\ &= 3d\frac{2^{2d-3}}{2^d-1} \cdot R_{add}. \end{aligned} \tag{88}$$

In the case where wavelets are used instead of curvelets at the finest scale, the redundancy upper-bound is changed to

$$\begin{aligned} R' &= (N_f \cdot R_f \cdot (R_w - 1) + 1) \cdot R_{add} \\ &= \left(3d\frac{2^{d-3}}{2^d-1} + 1\right) \cdot R_{add}. \end{aligned} \tag{89}$$

Table 2 compares numerically the redundancy of the original and the Low Redundancy FCT introduced in the next section in 2D and 3D, when wavelets (W) or curvelets (C) are implemented at the finest scale. It may be worth mentioning that for the practitioner, the memory storage requirement of the original FCT (as implemented in Curvelab) is twice larger than the one predicted by the redundancy formula. Indeed, the original curvelets are complex

	Original FCT		LR-FCT	
	C	W	C	W
2-D	7.11	3.56	4.00	2.00
3D	24.38	5.42	10.29	2.29

Table 2
Redundancy of the original FCT and the Low Redundancy one in 2-D and 3D, when wavelets (W) or curvelets (C) are used at the finest scale.

and real curvelets are obtained by hermitian symmetry. This explains the redundancy 40 [3] claimed by [26] emphasizing the need of lower redundancy curvelets.

5.4 Low redundancy implementation

In this section, we detail the Low Redundancy FCT (hereafter LR-FCT) introduced in [80]. The overall implementation of this FCT differs from the original one in several points. The main difference lays in the way to apply the Meyer wavelet transform to the data. Other changes have been introduced as described hereafter.

5.4.1 The multiscale separation

The extra redundancy of the curvelets as implemented in Curvelab originates mainly from the way the radial window is implemented, especially at the finest scale. As explained in Section 5.1, the Meyer wavelet at the finest scale is supported on $[-2/3, -1/6[\cup]1/6, 2/3[$, hence exceeding the Shannon band. In the original FCT, the exceeding end of the finest scale wavelet is replaced with its mirrored version around the vertical axis at $|\xi| = 1/2$.

In the LR-FCT implementation, a different approach is adopted. First, the supports of the scaling and wavelet functions (hence filters) are shrank by a factor of 4/3. Furthermore, to maintain the uniform partition of unity, which plays an important role for isometry of the transform, following [67], the finest scale wavelet is modified by suppressing its decreasing tail so that the wavelet becomes a constant over $]-1/2, -1/4]\cup]1/4, 1/2]$ (see the dashed line on Figure 24). The right part of Figure 25 shows the impact of the proposed modifications on the 2-D curvelets in the frequency domain. This strategy and the conclusions carry over to the 3D case.

This modification to the Meyer wavelets reduces the redundancy of the trans-

[3] In fact, it should be ≈ 50 as can be read from Table 2.

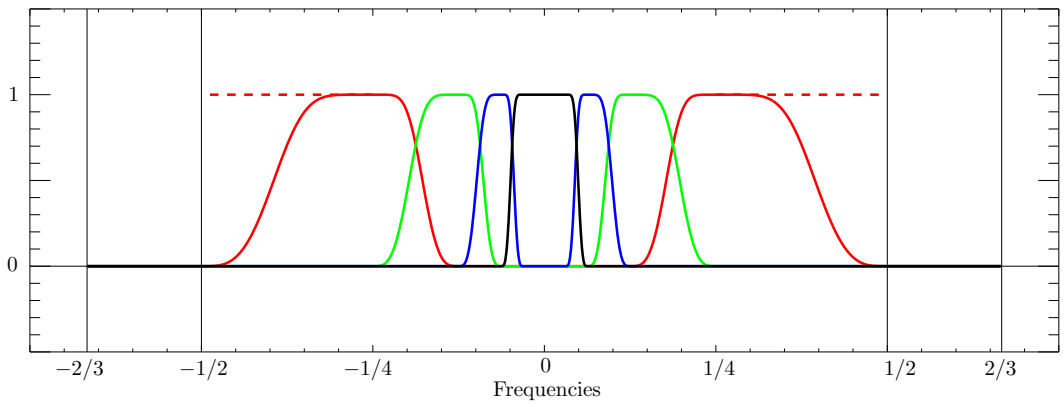


Fig. 24. Modified Meyer filters in Fourier domain. A scaling function is plotted in black, and in plain lines are the amplitude of the three following wavelet functions. In the Low Redundancy implementation,all the filters are shrunk to fit in the $[-1/2, 1/2]$ box, and the decreasing ends of the finest scale filters are replaced by a constant (dashed red line) to keep all the information.

form. Indeed, as was shown in Section 5.3, the redundancy of the transform is proportional to a factor R_{add} due to the Meyer wavelet transform. With this modified version of the transform however, the wavelets do not add any redundancy and $R_{add} = 1$ instead of $(4/3)^d$ for the original transform.

However, this comes at the cost of some changes undergone by the curvelet atoms at the finest scale. First of all, they are less sharply supported in the spatial domain than the original curvelets because of the discontinuity of their Fourier transform, while the decay of the other curvelets remain unchanged. Secondly, they obey a parabolic scaling but with a different constant compared to the curvelets at the other scales.

5.4.2 Ensuring zero-mean subbands

In the original wrapping-based FCT [75, 81], the wedges are wrapped around the origin using a simple modulo operator, which makes every point fit into a well-sized rectangular parallelepiped centered at the origin whose size is designed so as the data does not overlap with itself after wrapping. However, nothing prevents the center of the parallelepiped from receiving a significant non-zero wrapped Fourier coefficient. After an inverse FFT of the wrapped wedge, it is likely to obtain curvelet coefficient subbands with non-zero means. This is obviously unsuitable since curvelet coefficients are expected to represent high frequency content, and typical thresholding-based processing (e.g. denoising) will be hampered in such a situation. Hopefully, the size and position of the wedges are so that this misleading phenomenon is generally prevented in practice. Nevertheless, this is not guaranteed in general.

Therefore, in order to ensure zero-mean curvelet subbands, a straightforward

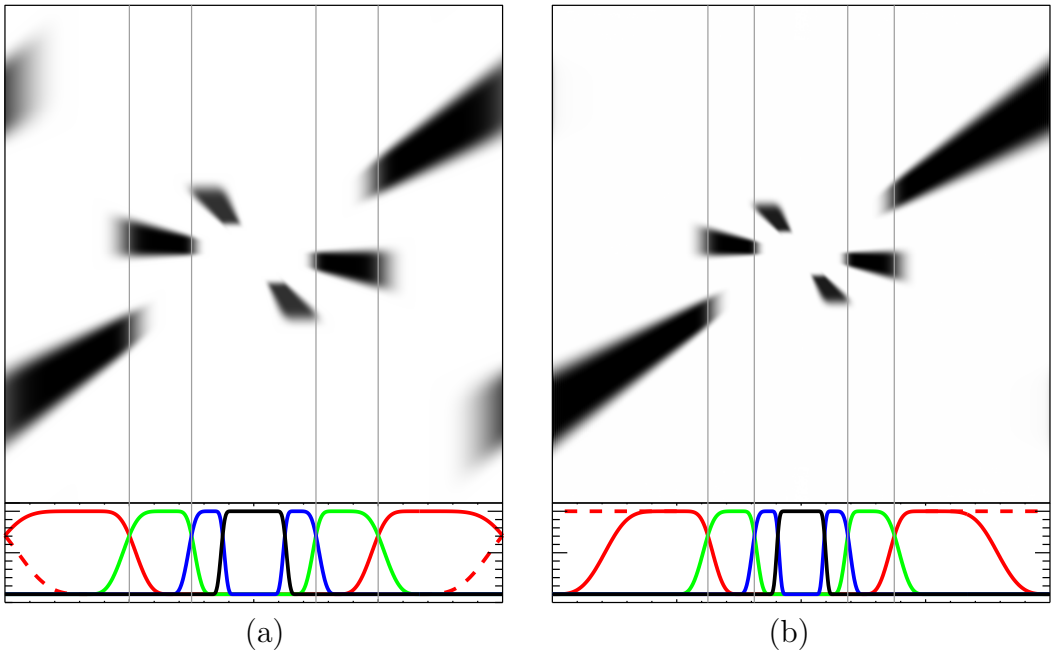


Fig. 25. Top : Examples of 2-D (real) curvelets in Fourier domain at three consecutive scales and different orientations, with the zero frequency at the center. From the outermost edge to the inside, a finest scale curvelet and two lower scale curvelets. (a) According to the Curvelab implementation, and (b) with our modified low-redundancy implementation. Bottom : the corresponding Fourier transforms of 1-D Meyer scaling and wavelet functions.

solution is to translate each rectangular parallelepiped where a wedge has to be wrapped in such a way that the center (zero frequency) gets a true zero coefficient, i.e. a point out of the wedge support, and then to wrap the data around the translated box. Doing so, the curvelet subbands are ensured to be zero-mean valued after wrapping. Figure 26 illustrates the difference between the two wrapping strategies in 2-D for the sake of legibility. The technique extends readily to the 3D case.

5.4.3 Properties

This section enumerates the main properties enjoyed by the LR-FCT implementation.

- **Reduced redundancy**: with a reduction factor of $(4/3)^d$ compared to the original version. This is one of the distinctive properties of the LR-FCT, and was the main goal underlying these modifications in the first place. For example, the LR-FCT with full curvelets at all scales is (almost) as redundant as the original one with wavelets at the finest scale. In 3D, redundancy implied by this implementation is 2.5 times lower that the original FCT with curvelets at all scales. In short, this implementation achieves a low redundancy while maintaining the directional selectivity property at the

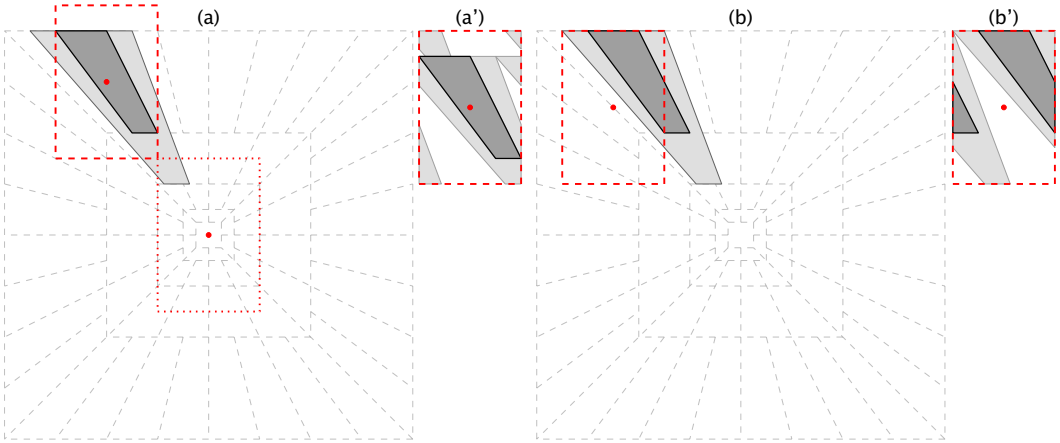


Fig. 26. (a) Representation of the influence of a wedge and its overlapping region. The centered dotted rectangle corresponds to the minimal size in which the wedge will be wrapped. (a') The result of the wrapping. (b) The wrapping in a rectangle of the same size, but whose center is chosen such that the zero frequency (big dot) falls outside the support of the wedge. (b') The corresponding wedge after translation and wrapping.

finest scale unlike the original FCT where wavelets are advocated at the finest scale to lower the redundancy [21, 75].

- **Isometry and fast exact reconstruction**: owing to the uniform partition property of the Meyer wavelets, and the coverage property (79) of the angular window, the collection of curvelets in Fourier obtained by multiplication of the scale and angular windows also ensures a uniform partition of the unity. Therefore, with a proper normalization of the FFT (wrapping is a simple reindexing), the proposed FCT corresponds to a Parseval tight frame (PTF), i.e. the frame operator $\mathrm{CC}^* = \mathrm{I}$, where C^* is the FCT analysis operator and C its adjoint. With the PTF property, C turns out to be also the inverse transform operator associated to a fast reconstruction algorithm (each step of the forward transform is easily invertible).
- **Parabolic scaling**: by construction, the curvelets obey the parabolic scaling law with one short and two long sides $\sim (2^{-j}, 2^{-j/2}, 2^{-j/2})$. Although at the finest scale, this property is less faithful to the continuous construction compared to the original FCT (see also the discussion at the end of Section 5.4.1).
- **Non-equal ℓ_2 norm atoms**: although the LR-FCT implements a PTF, the modified curvelets at the finest scale do not have the same ℓ_2 norm as the curvelets in the other (coarsest) scales. These ℓ_2 norms can nonetheless be calculated analytically so as to normalize the associated curvelets coefficients, which is important for instance in every processing which involves thresholding.
- **Guaranteed zero-mean subbands**: this is a consequence of the wise translation trick prior to wrapping explained in Section 5.4.2. Of course, this operation preserves isometry and ℓ_2 norms.

5.4.4 *LR-FCT denoising, a good tradeoff between efficiency and memory storage*

In this experiment, the denoising performance using the LR-FCT is compared to several other 3D multiscale transforms on various types of datasets (3D spatial data, hyperspectral images and videos). The noise is additive white Gaussian (AWGN) and simple hard thresholding is used. The video datasets included in this experiment are the standard videos *mobile*, *tempete*, and *coastguard* CIF sequences available at `www.cipr.rpi.edu`. For hyperspectral data, a dataset from the OMEGA spectrometer on Mars Express was used (`www.esa.int/marsexpress`) with 128 wavelength from $0.93\mu m$ to $2.73\mu m$. Beside LR-FCT, the other transforms involved in this comparative study are: the dual-tree complex wavelet transform [82, 83], the surfacelet transform [26] and the orthogonal (decimated) and translation-invariant (undecimated) wavelet transforms.

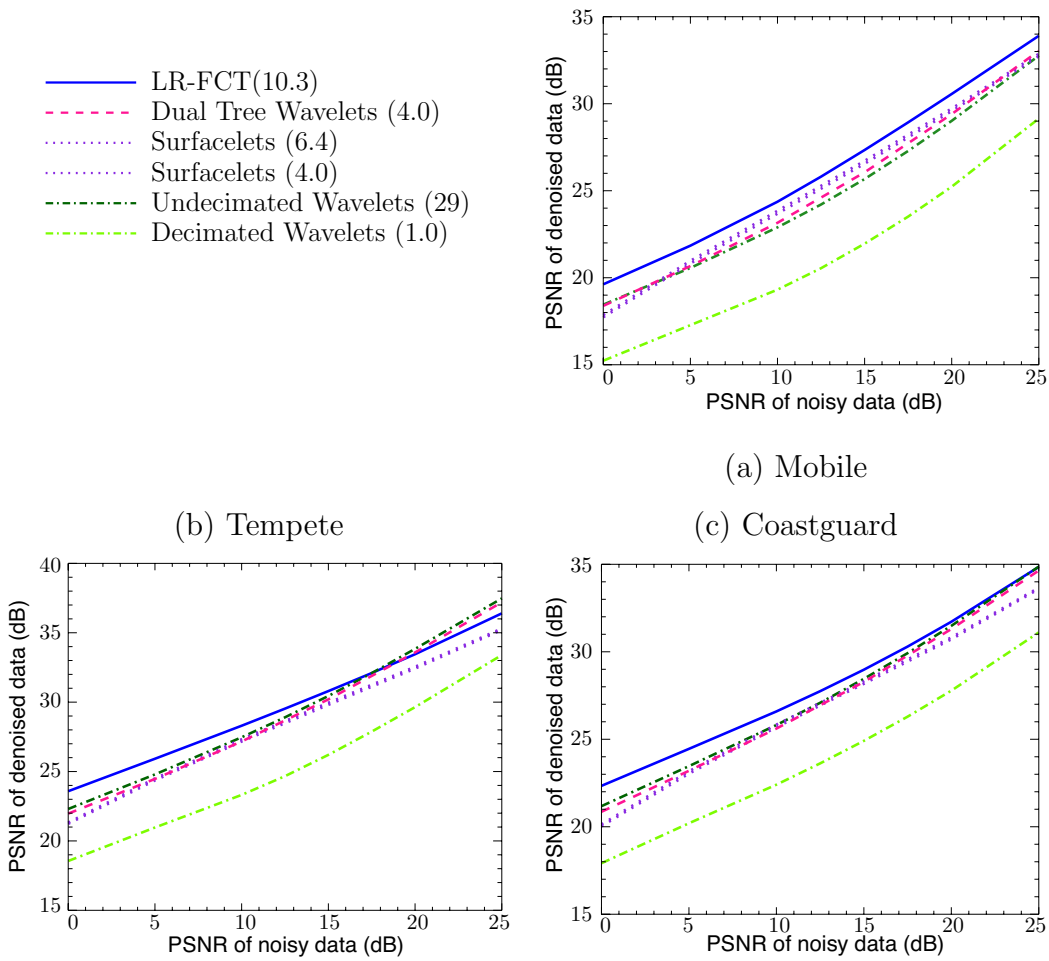


Fig. 27. Output PSNR as a function of the input PSNR for three video sequences. (a) *mobile*, (b) *tempete*, and (c) *coastguard* CIF sequence. The redundancy of each transform is indicated in parentheses on the legend.

Figures 27 and 28 show the output PSNR after denoising as a function of the input PSNR for each transform. Each point on each curve is the average output PSNR on ten noise realizations. The reader may have noticed that the wavelet results are much better here than those tabulated in [26]. The reason behind this is that unlike those authors, the Cohen-Daubechies-Fauveau 7/9 filterbank is used here, which is much better for denoising. Figure28 displays the results for the hyperspectral data from Mars Express (see caption for details). From these experiments, it can be clearly seen that the LR-FCT compares very favorably to the other multiscale geometrical 3D transforms, and is particularly better at the low PSNR regime. In a nutshell, it can be safely concluded that LR-FCT provides a very good compromise between denoising performance and memory/CPU requirements.

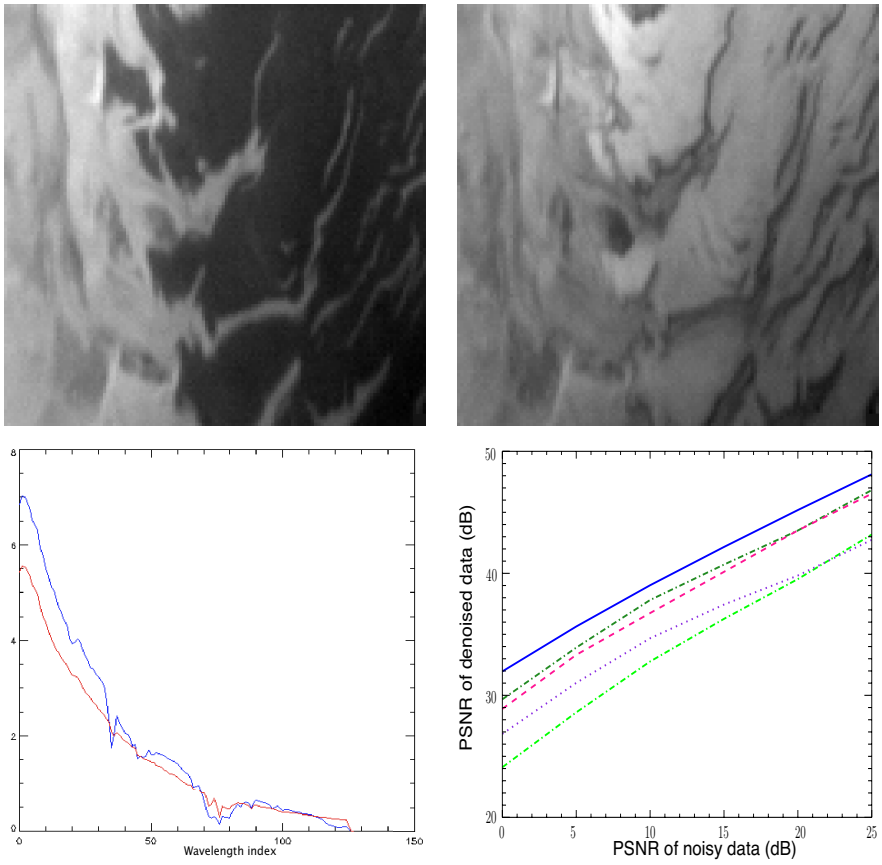


Fig. 28. Top row: Mars Express observations at two different wavelengths. Bottom-left: two spectra at two distinct pixels. Bottom-right: output PSNR as a function of the input PSNR for different transforms; see Figure 27 for legend of the curves.

Inpainting aims to restore missing data information based upon the still available (observed) cues from destroyed, occluded or deliberately masked subregions of the data. Inpainting has received considerable interest and excitement and has been attacked using diffusion and transport PDE/Variational principles, non-local exemplar region fill-in and sparsity-based regularization; see e.g. [6, 7] and references therein.

Let $f \in \mathbb{R}^N$ be a vectorized form of the sought after 3D data cube which is $\sqrt{N} \times \sqrt{N} \times \sqrt{N}$, and $\mathrm{M} \in \{0,1\}^{P \times N}$, $P < N$ be a binary rectangular matrix where each of its rows is zero except at the entry where a voxel is not missing. The observed (incomplete) data g is then the result of applying the lossy operator M to f:

$$g = \mathrm{M}f + \varepsilon \ .$$

where ε is some noise of finite variance σ^2 that may contaminate the observed values. Restoring f from g is an ill-posed which necessitates some form of regularization to reduce the space of candidate solutions. Here, we promote solutions that are sparse in some prescribed overcomplete dictionary of atoms $\Phi \in \mathrm{R}^{N \times L}$, $L \geq N$, meaning that $x := \Phi\alpha$ (a synthesis prior) can be sparsely represented to a high accuracy by a few number of atoms in Φ. Put formally, we are seeking to solve the following optimization problem :

$$\min_{\alpha \in \mathbb{R}^L} \|\alpha\|_0 \text{ s.t. } \|g - \mathrm{M}\Phi\alpha\|_2 \leq \epsilon(\sigma) \ , \tag{90}$$

where $\|\cdot\|_0$ is the ℓ_0 pseudo-norm that counts the number of nonzero entries of its argument, and $\epsilon(\sigma)$ is the constraint radius that depends on the noise variance. This is a very challenging NP-hard optimization problem, and one has to resort to alternative formulations or greedy algorithms to attempt to solve it. For instance, convex ℓ_1 relaxation could be used instead of the ℓ_0 penalty.

This application makes use of the algorithm devised in [6] which can be viewed as a stagewise hybridization of matching pursuit with block-coordinate relaxation. The adjective "stagewise" is because their algorithm exploits the fact that the dictionary is structured (union of transforms $\Phi = [\Phi_1, \cdots, \Phi_K]$) with associated fast analysis and synthesis operators Φ_k^{T} and Φ_k; see [6, 84] for details. For the reader convenience, Algorithm 10 recalls the main steps of this inpainting algorithm.

For the following experiment, this algorithm was used with a dictionary containing two transforms: the LR-FCT and the undecimated discrete wavelet transform (UDWT), in order to better take into account the morphological

Algorithm 10: Inpainting Algorithm.

Data: Observed data g and mask M.

Input: Dictionary $\Phi = [\Phi_1 \cdots \Phi_K]$, number of iterations T_{iter}, final threshold τ (e.g. 3).

begin

- Initial components $f_k^{(0)} = 0, k = 1, \cdots, K.$;
- Initial residual $r^{(0)} = g.$;
- Initial threshold: let $k^\star = \arg\max_k \|\Phi_k^{\mathrm{T}} g\|_\infty$, set $\lambda_0 = \max_{k \neq k^\star} \|\Phi_k^{\mathrm{T}} g\|_\infty.$;
- **for** $t = 1$ **to** T_{iter} **do**
  - **for** $k = 1$ **to** K **do**
    - Compute marginal residuals $r_k^{(t)} = r^{(t-1)} + f_k^{(t-1)}.$;
    - Update kth component coefficients by thresholding $\alpha_k^{(t)} = \mathrm{Thresh}_{\lambda_{t-1}} \left(\Phi_k^{\mathrm{T}} r_k^{(t)}\right).$;
    - Update kth component $f_k^{(t)} = \Phi_k \alpha_k^{(t)}.$;
  - Update the inpainted data $f^{(t)} = \sum_{k=1}^{K} f_k^{(t)}.$;
  - Update the residuals $r^{(t)} = g - \mathrm{M} f^{(t)}.$;
  - Update the threshold $\lambda_t = \lambda_0 - t\left(\lambda_0 - \tau\sigma\right)/T_{\text{iter}}.$;

Result: The estimate $f^{(T_{\text{iter}})}$ of f.

diversity of the features contained in the data. Figure 29 shows the inpainting result on a synthetic cerebral MRI volume available on BrainWeb [85] at `http://www.bic.mni.mcgill.ca/brainweb/` with two masks: 80% random missing voxels, and 10% missing z slices. We can see that even with 80% missing voxels, we can still see incredibly faint details in the restored anatomical structures such as in the gyri and the cerebellum.

6 Sparsity on the Sphere

Many wavelet transforms on the sphere have been proposed in the past years. Using the lifting scheme [86] 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 [87, 88, 89, 90]. These works have been extended to directional wavelet transforms [91, 92]. 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. [93] and [94] proposed the first redundant wavelet transform, based on the spherical harmonics transform, which presents an inverse transform. [29] proposed an invertible isotropic undecimated wavelet

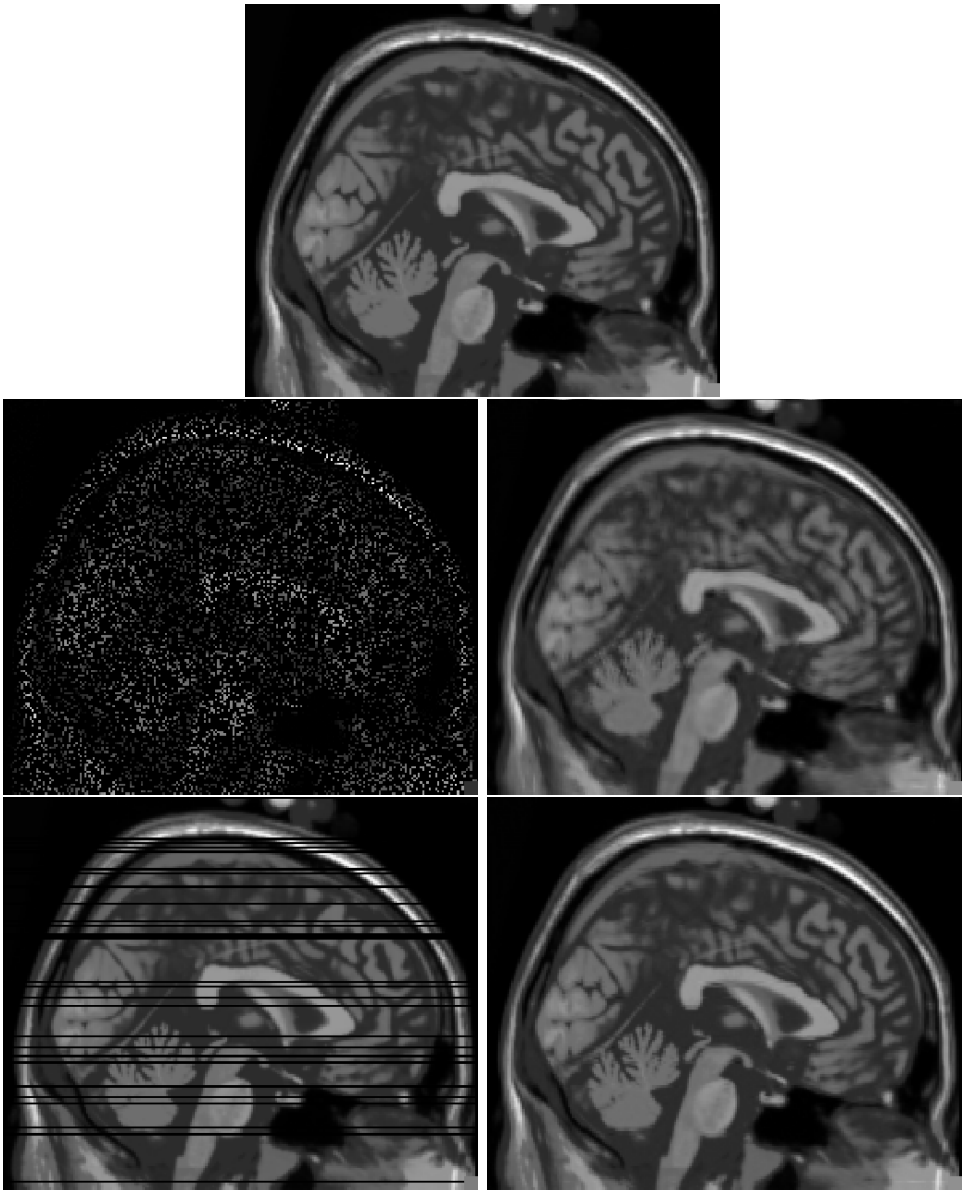


Fig. 29. Top : a sagittal $((y, z))$ slice of the original synthetic MRI volume from BrainWeb[85]. Left column : the data with missing areas, random 80% missing voxels, and 10% missing z slices. Right : inpainting results with a LF-FCT+UDWT dictionary.

transform (UWT) on the sphere, also based on spherical harmonics, which has the same property as the starlet transform, i.e. the sum of the wavelet scales reproduces the original image. A similar wavelet construction [30, 31, 32] used the so-called needlet filters. [33] 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 [34, 35, 36] or inpainting [37, 38].

Extensions to the sphere of 2D geometric multiscale decompositions such as the ridgelet transform and the curvelet transform were presented in [29].

The aim of this Section is to introduce data representation on the sphere and the Isotropic Undecimated Wavelet Transform on the Sphere (IUWTS). These two elements will be useful to build the full 3D wavelets on the ball in the next Section. We also present the extension of the IUWTS to 2D-1D data following much the same approach as in Section 2.3. Such a transform can be very useful for time varying or multichannel data on the sky.

In the last part of this section we illustrate the spherical 2D-1D transform on an application to the deconvolution of multichannel data.

6.1 Data representation on the sphere

6.1.1 Discrete data representation on the sphere

Various pixelization schemes for data on the sphere exist in the literature. These include the Equidistant Coordinate Partition (ECP), the Icosahedron method [95], the Quad Cube [96], IGLOO [97], HEALPix [98], Hierarchical Triangular Mesh (HTM) [99] or Gauss-Legendre Sky Pixelization (GLESP) [100]. 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 chapter, we use the HEALPix representation which has several useful properties.

The HEALPix representation (Hierarchical Equal Area isoLatitude Pixelization of a sphere) [98] [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 the poles and equator. Each face is subsequently divided into N_{side}^2 pixels following a quadrilateral multiscale tree structure (see Fig. 30). 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.

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

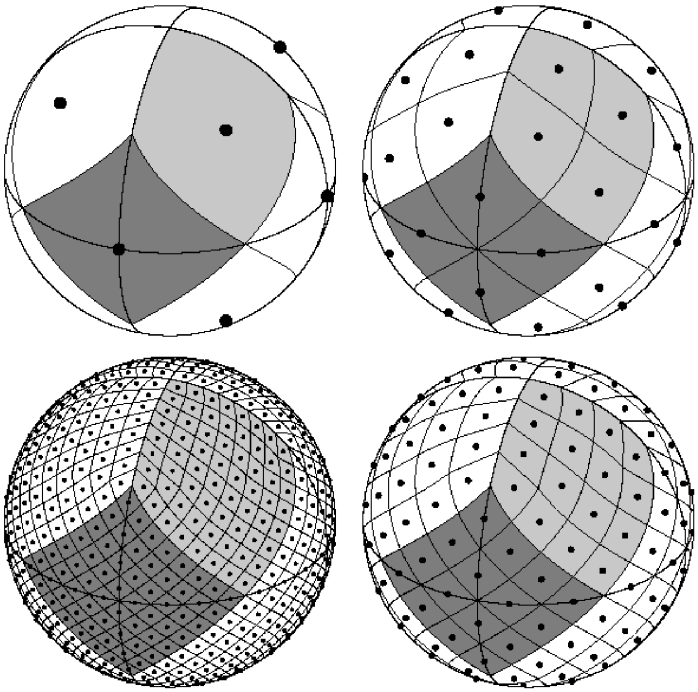


Fig. 30. The HEALPix sampling grid for four different resolutions.

6.1.2 Spherical Harmonics

The equivalent of the Fourier transform on the sphere is the spherical harmonics transform. Any function $f(\theta, \varphi) \in L_2(S^2)$ on the sphere S^2 in $\mathbb{R}^3$ can be decomposed into spherical harmonics:

$$f(\theta, \varphi) = \sum_{l=0}^{+\infty} \sum_{m=-l}^{l} f_{lm} Y_{lm}(\theta, \varphi), \tag{91}$$

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

$$Y_{lm}(\theta, \varphi) = \sqrt{\frac{2l+1}{4\pi} \frac{(l-|m|)!}{(l+|m|)!}} P_{lm}(\cos\varphi) e^{im\theta}, \tag{92}$$

This definition, ensures the normalization of the spherical harmonics so that $\int |Y_{lm}|^2 d\Omega = 1$. P_{lm} are the associated Legendre functions (or polynomials) defined by the following differential equation:

$$\frac{d}{dt}\left[(1-t^2)\frac{d}{dt}P_{lm}\right] + \left(l(l+1) - \frac{m^2}{1-t^2}\right)P_{lm} = 0. \tag{93}$$

These functions are related to the Legendre polynomials P_l by

$$P_{lm}(t) = (-1)^m (1-t^2)^{m/2} \frac{d^m}{dt^m} P_l(t), \tag{94}$$

where P_l is:

$$P_l(t) = \frac{1}{2^l l!} \frac{d^l}{dt^l} (t^2 - 1)^l. \tag{95}$$

An important property of the spherical harmonics is that they are orthonormal:

$$\int_0^{2\pi} \int_0^{\pi} Y_{lm}^*(\theta, \varphi) Y_{l'm'}(\theta, \phi) \sin(\theta) d\theta d\varphi = \delta_{ll'} \delta_{mm'} \ . \tag{96}$$

Since they form an orthonormal basis of $L_2(S^2)$, the spherical harmonics coefficients of f are uniquely determined by its projection onto $\{Y_{lm}\}_{lm}$ using the Hermitian product:

$$\forall l \in \mathbb{N}, \forall m \in [\![-l, l]\!] \quad f_{lm} = \int_0^{2\pi} \int_0^{\pi} Y_{lm}^*(\theta, \varphi) f(\theta, \varphi) \sin(\theta) d\theta d\varphi \ . \tag{97}$$

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

6.2 Isotropic Undecimated Wavelet Transform on the Sphere

Here an undecimated isotropic transform (UWTS) is described which is similar in many respects to the starlet transform (see Section 2.2), 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 $\phi^{l_c}(\theta, \varphi)$ with cut-off frequency l_c and azimuthal symmetry, meaning that ϕ^{l_c} does not depend on the azimuth φ. Hence the spherical harmonics coefficients $\phi_{lm}^{l_c}$ of ϕ^{l_c} vanish when $m \neq 0$ so that:

$$\phi^{l_c}(\theta, \varphi) = \phi^{l_c}(\theta) = \sum_{l=0}^{l_c} \phi_{l0}^{l_c} Y_{l0}(\theta, \varphi). \tag{98}$$

Then, convolving a function $f(\theta, \varphi) \in L_2(S^2)$ with ϕ^{l_c} is greatly simplified and the spherical harmonics coefficients of the resulting map c^0 are readily given by

$$c_{lm}^0 = \left[\phi^{l_c} * f\right]_{lm} = \sqrt{\frac{2l+1}{4\pi}} \phi_{l0}^{l_c} f_{lm}. \tag{99}$$

6.2.1 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:

$$\begin{aligned} c^0 &= \phi^{l_c} * f \\ c^1 &= \phi^{2^{-1}l_c} * f \\ &\dots \\ c^j &= \phi^{2^{-j}l_c} * f, \end{aligned} \tag{100}$$

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 its spherical harmonics coefficients as:

$$\begin{aligned} H^j_{lm} &= \sqrt{\frac{4\pi}{2l+1}} h^j_{lm} \\ &= \begin{cases} \frac{\phi^{2^{-(j+1)}l_c}_{lm}}{\phi^{2^{-j}l_c}_{lm}} & \text{if } l < \frac{l_c}{2^{j+1}} \quad \text{and} \quad m = 0, \\ 0 & \text{otherwise .} \end{cases} \end{aligned} \tag{101}$$

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

6.2.2 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:

$$\begin{aligned} G^j_{lm} &= \sqrt{\frac{4\pi}{2l+1}} g^j_{lm} \\ &= \begin{cases} \frac{\psi^{2^{-(j+1)}l_c}_{lm}}{\phi^{2^{-j}l_c}_{lm}} & \text{if } l < \frac{l_c}{2^{j+1}} \quad \text{and} \quad m = 0, \\ 1 & \text{if } l \geq \frac{l_c}{2^{j+1}} \quad \text{and} \quad m = 0, \\ 0 & \text{otherwise .} \end{cases} \end{aligned} \tag{102}$$

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 starlet transform algorithm, the wavelet coefficients can be defined as the difference between two consecutive resolutions, $w^{j+1}(\theta, \varphi) = c^j(\theta, \varphi) -$

$c^{j+1}(\theta, \varphi)$. This defines a zonal wavelet function ψ^{l_c} as:

$$\psi_{lm}^{2^{-j}l_c} = \phi_{lm}^{2^{-(j-1)}l_c} - \phi_{lm}^{2^{-j}l_c}. \tag{103}$$

The high pass filters g^j associated with this wavelet are expressed as:

$$\begin{aligned} G_{lm}^j &= \sqrt{\frac{4\pi}{2l+1}} g_{lm}^j \\ &= 1 - \sqrt{\frac{4\pi}{2l+1}} h_{lm}^j = 1 - H_{lm}^j. \end{aligned} \tag{104}$$

Obviously other wavelet functions could be used just as well.

6.2.3 Choice of the Scaling Function

Any function with a cut-off frequency is a possible candidate. We retained here a B-spline function of order 3 (as in Section 2.2):

$$\hat{\phi}_{lm}^{l_c} = \frac{3}{2} B_3 \left(\frac{2l}{l_c} \right) , \tag{105}$$

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

$$B_3(x) = \frac{1}{12} \left(|x-2|^3 - 4|x-1|^3 + 6|x|^3 - 4|x+1|^3 + |x+2|^3 \right). \tag{106}$$

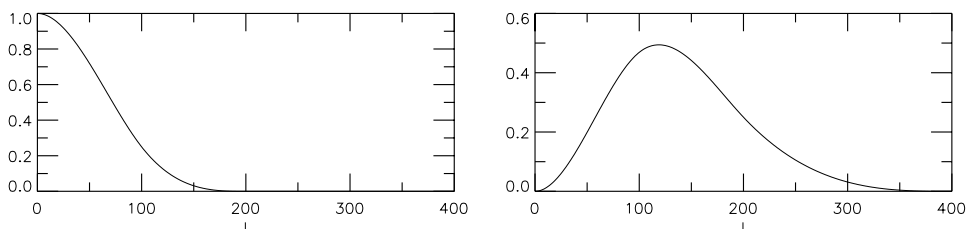


Fig. 31. On the left, spherical harmonics coefficients ϕ_{l0} of the the scaling function ϕ and, on the right, those of the wavelet function ψ.

In Fig. 31 the spherical harmonics coefficients of the scaling function derived from a B_3-spline, and those of the associated wavelet function (103), are plotted as a function of l. Other functions such as the needlet function [30] can be used as well.

The steps of the UWT on the sphere of a discrete image X sampled from f are summarized in Algorithm 11. If the wavelet function corresponds to the choice (103), Step 3 in this UWTS algorithm reduces to $w^{j+1} = c^j - c^{j+1}$.

Algorithm 11: The Undecimated Wavelet Transform on the Sphere.

Task: Compute the UWTS of a discrete X.
Parameters: Data samples X and number of wavelet scales J.
Initialization:
- $c^0 = X$.
- Compute the B_3-spline scaling function and derive ψ_{l0}, H_{l0} and G_{l0} numerically.
- Compute the corresponding spherical harmonics transform of c^0_{lm}.

for $j = 0$ ***to*** $J - 1$ **do**
(1) Compute the spherical harmonics transform of the scaling coefficients: $c^{j+1}_{lm} = c^j_{lm} H^j_{l0}$.
(2) Compute the inverse spherical harmonics transform of c^{j+1}_{lm} to get c^{j+1}.
(3) Compute the spherical harmonics transform of the wavelet coefficients: $w^{j+1}_{lm} = c^j_{lm} G^j_{l0}$.
(4) Compute the inverse spherical harmonics transform of w^{j+1}_{lm} to get w^{j+1}.

Output: $\mathcal{W} = \{w^1, w^2, \ldots, w^J, c^J\}$ the UWTS of X.

Fig. 32 shows the Mars topographic map (top left) [5] and its wavelet transform, using five scales (four wavelet scales + coarse scale). The sum of the five scales reproduces exactly the original image.

6.2.4 Inverse Transform

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:

$$c^0(\theta, \varphi) = c^J(\theta, \varphi) + \sum_{j=1}^{J} w^j(\theta, \varphi). \tag{107}$$

This reconstruction formula is the same as with the starlet algorithm.

But since the transform is redundant there is actually no unique way to reconstruct an image from its coefficients. Indeed, using the relations:

$$\begin{aligned} \hat{c}^{j+1}_{lm} &= H^j_{l0} c^j_{lm} \\ \hat{w}^{j+1}_{lm} &= G^j_{l0} c^j_{lm} \ , \end{aligned} \tag{108}$$

a least-squares estimate of c^j from c^{j+1} and w^{j+1} gives:

$$c^j_{lm} = c^{j+1}_{lm} \widetilde{H}^j_{l0} + w^{j+1}_{lm} \widetilde{G}^j_{l0} \ , \tag{109}$$

[5] 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.

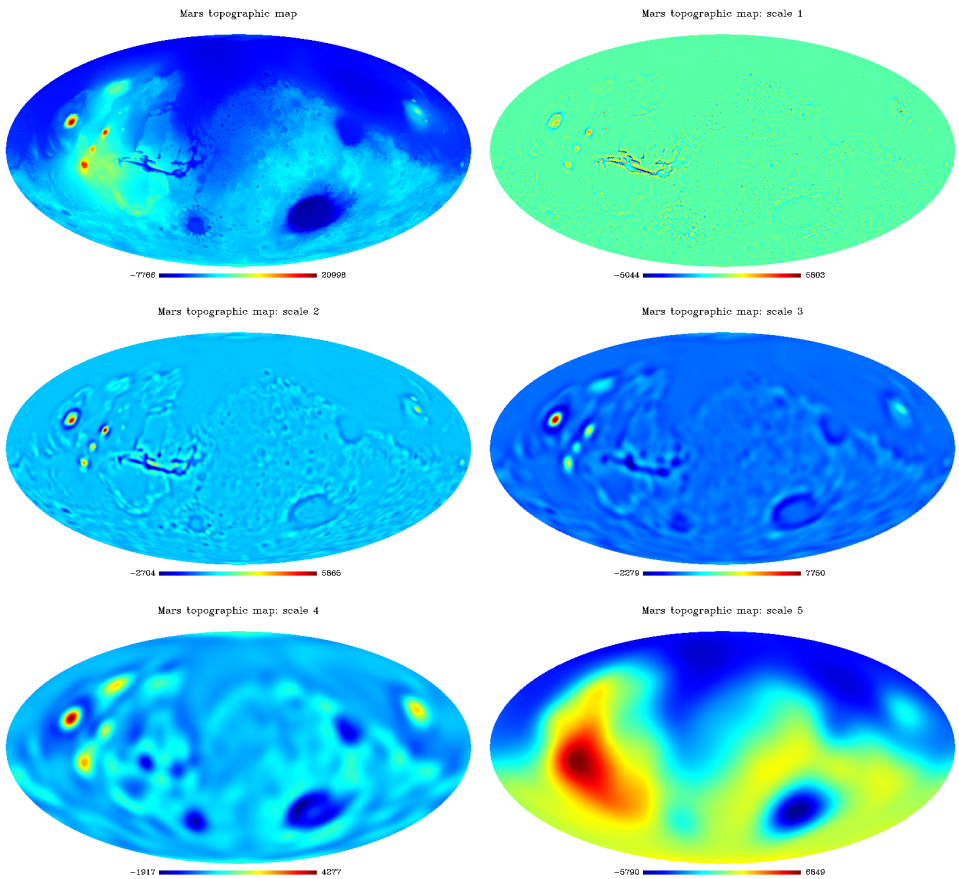


Fig. 32. Mars topographic map and its UWTS (four wavelet detail scales and the scaling (smooth) band).

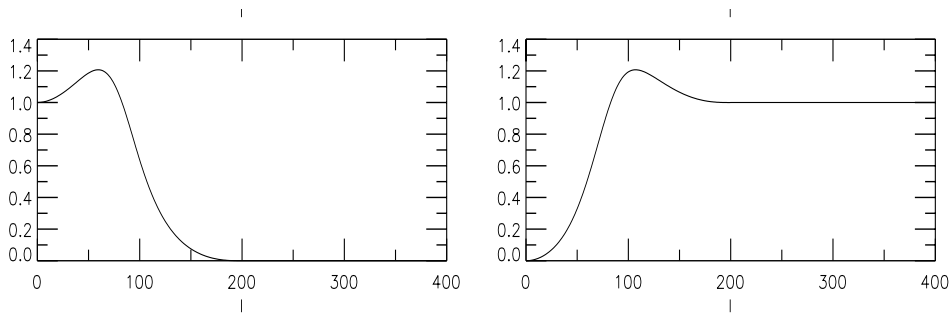


Fig. 33. On the left, spherical harmonics coefficients $\tilde{h}_{l0}$ the filter $\tilde{h}$, and on the right, those of the filter $\tilde{g}$.

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

$$\widetilde{H}^j_{l0} = \sqrt{\frac{4\pi}{2l+1}}\tilde{h}^j_{l0} = \frac{H^{*j}_{l0}}{\left|H^j_{l0}\right|^2 + \left|G^j_{l0}\right|^2},$$
$$\widetilde{G}^j_{l0} = \sqrt{\frac{4\pi}{2l+1}}\tilde{g}^j_{l0} = \frac{G^{*j}_{l0}}{\left|H^j_{l0}\right|^2 + \left|G^j_{l0}\right|^2}. \tag{110}$$

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 $\widetilde{H}$ and $\widetilde{G}$ are plotted in Fig. 33. The reconstruction algorithm is given in Algorithm 12.

Algorithm 12: 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 ψ_{l0}, H_{l0}, G_{l0}, $\widetilde{H}_{l0}$ and $\widetilde{G}_{l0}$ numerically.
- Compute the spherical harmonics transform of c^J to get c^J_{lm}.

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 w^{j+1}_{lm}.
(2) Multiply c^{j+1}_{lm} by $\widetilde{H}^j_{l0}$.
(3) Multiply w^{j+1}_{lm} by $\widetilde{G}^j_{l0}$.
(4) Get the spherical harmonics of $c^j_{lm} = c^{j+1}_{lm} + w^{j+1}_{lm}$.

Compute The inverse Spherical Harmonics transform of c^0_{lm}.
Output: c^0 is the inverse UWT on the sphere.

Fig. 34 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.

6.3 2D-1D Wavelet on the Sphere

Using the Isotropic Undecimated Wavelet Transform on the Sphere, one can extend the 2D-1D formalism [51] presented in Section 2.3 to spherical data with an additional dependency in either time or energy [39]. As before, since the 2D spatial dimension and the 1D time or energy dimension do not have the same physical meaning, it appears natural that the wavelet scale along the third dimension should not be connected to the spatial scale. Hence, the 2D-1D wavelet function is defined by tensor product of a 2D wavelet and a 1D wavelet:

$$\psi(k_\theta, k_\varphi, k_t) = \psi^{(\theta\phi)}(k_\theta, k_\varphi)\psi^{(t)}(k_t), \tag{111}$$

where $\psi^{(\theta\phi)}$ is the spherical spatial wavelet and $\psi^{(t)}$ the 1D wavelet along the third dimension. Considering only isotropic and dyadic spatial scales the discrete 2D-1D wavelet decomposition can be built by first taking an Isotropic Undecimated Wavelet transform for each k_t, followed by a 1D wavelet transform (e.g. 1D starlet) for each resulting spatial wavelet coefficient along the third dimension.

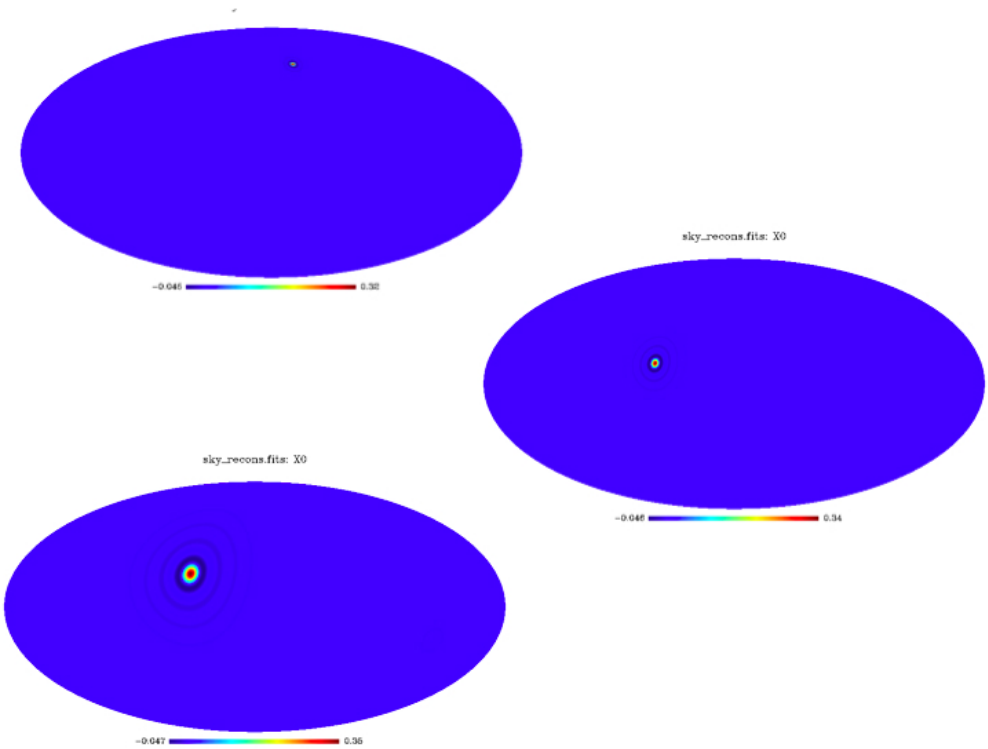


Fig. 34. Reconstruction from a single wavelet coefficient at different scales. Each map is obtained by setting all wavelet coefficients to zero but one, and by applying an inverse UWTS. Depending on the position and scale of the non-zero coefficient, the reconstructed map shows an isotropic feature at different scales and positions.

Hence for a given multichannel data set on the sphere $D[k_\theta, k_\varphi, k_t]$, applying first the IUWTS yields

$$\forall k_t, \quad D[\cdot,\cdot,k_t] = c_{J_1}[\cdot,\cdot,k_t] + \sum_{j_1=1}^{J1} w_{j_1}[\cdot,\cdot,k_t], \tag{112}$$

where J_1 is the number of spatial scales, a_{J_1} is the (spatial) approximation subband and $\{w_{j_1}\}_{j_1=1}^{J_1}$ are the (spatial) detail subbands. To lighten the notations in the sequel, we replace the two spatial indices by a single index k_r which corresponds to the pixel index. Expression (112) reads now

$$\forall k_t, \quad D[\cdot,k_t] = a_{J_1}[\cdot,k_t] + \sum_{j_1=1}^{J1} w_{j_1}[\cdot,k_t],. \tag{113}$$

Then, for each spatial location k_r and for each 2D wavelet scale j_1, a 1D wavelet transform can be applied along t on the spatial wavelet coefficients

$w_{j_1}[k_r, \cdot]$ such that

$$\forall k_t, \quad w_{j_1}[\cdot, k_t] = w_{j_1,J_2}[\cdot, k_t] + \sum_{j_2=1}^{J_2} w_{j_1,j_2}[\cdot, k_t], \tag{114}$$

where J_2 is the number of scales along t. The approximation spatial subband c_{J_1} is processed in a similar way, hence yielding

$$\forall k_t, \quad c_{J_1}[\cdot, k_t] = c_{J_1,J_2}[\cdot, k_t] + \sum_{j_2=1}^{J_2} w_{J_1,j_2}[\cdot, k_t] \ . \tag{115}$$

Inserting (114) and (115) into (113), we obtain the 2D-1D spherical undecimated wavelet representation of D:

$$D[k_r, k_t] = c_{J_1,J_2}[k_r, k_t] + \sum_{j_1=1}^{J_1} w_{j_1,J_2}[k_r, k_t] + \sum_{j_2=1}^{J_2} w_{J_1,j_2}[k_r, k_t] + \sum_{j_1=1}^{J_1} \sum_{j_2=1}^{J_2} w_{j_1,j_2}[k_r, k_t] \ . \tag{116}$$

Just as in Section 2.3, four kinds of coefficients can be distinguished in this expression:

- Detail-Detail coefficients ($j_1 \leqslant J_1$ and $j_2 \leqslant J_2$):

$$w_{j_1,j_2}[k_r, \cdot] = (\delta - \bar{h}_{1\mathrm{D}}) \star (h_{1\mathrm{D}}^{(j_2-1)} \star a_{j_1-1}[k_r, \cdot] - h_{1\mathrm{D}}^{(j_2-1)} \star a_{j_1}[k_r, \cdot]).$$

- Approximation-Detail coefficients ($j_1 = J_1$ and $j_2 \leqslant J_2$):

$$w_{J_1,j_2}[k_r, \cdot] = h_{1\mathrm{D}}^{(j_2-1)} \star c_{J_1}[k_r, \cdot] - h_{1\mathrm{D}}^{(j_2)} \star c_{J_1}[k_r, \cdot].$$

- Detail-Approximation coefficients ($j_1 \leqslant J_1$ and $j_2 = J_2$):

$$w_{j_1,J_2}[k_r, \cdot] = h_{1\mathrm{D}}^{(J_2)} \star c_{j_1-1}[k_r, \cdot] - h_{1\mathrm{D}}^{(J_2)} \star c_{j_1}[k_r, \cdot].$$

- Approximation-Approximation coefficients ($j_1 = J_1$ and $j_2 = J_2$):

$$c_{J_1,J_2}[k_r, \cdot] = h_{1\mathrm{D}}^{(J_2)} \star c_{J_1}[k_r, \cdot].$$

As this 2D-1D transform 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 2D-1D wavelet transform.

6.4 Application: Multichannel Poisson Deconvolution on the Sphere

In this section, we present an application of this 2D-1D spherical wavelet to the deconvolution of multichannel data on the sphere in the presence of

Poisson noise. This application [101] was developed in the context of the Fermi Gamma-Ray Space Telescope, which studies the high-energy gamma-ray sky through its main instrument, the Large Area Telescope (LAT).

As mentioned in the application of the Cartesian 2D-1D in Section 2.4 to the same LAT data, the detection of point sources is complicated by the Poisson noise inherent to the weakness of the fluxes of celestial gamma rays and by the instrument's point spread function (PSF). In particular, the PSF is strongly energy-dependent, it varies from about 3.5 at 100 MeV to less than 0.1 (68% containment) at 10 GeV. Owing to large-angle multiple scattering in the tracker, the PSF has broad tails, the 95%/68% containment ratio may be as large as 3.

Using a direct extension of the Cartesian 2D-1D MS-VST presented in Section 2.4 to Spherical data with an energy dependence it is possible to address the multichannel PSF deconvolution problem on the sphere in the presence of Poisson noise in a single general framework.

6.4.1 2D-1D MS-VST on the Sphere

The extension of the 2D-1D MS-VST to spherical data simply amounts to replacing the Cartesian 2D-1D transform by the spherical transform defined in the previous section. Again, four kind of coefficients can be identified:

- Detail-Detail coefficients ($j_1 \leqslant J_1$ and $j_2 \leqslant J_2$):

$$w_{j_1,j_2}[k_r,\cdot] = (\delta - \bar{h}_{1\mathrm{D}})\star \\ \left(\mathcal{A}_{j_1-1,j_2-1}\left(h_{1\mathrm{D}}^{(j_2-1)} \star c_{j_1-1}[k_r,\cdot]\right) - \mathcal{A}_{j_1,j_2-1}\left(h_{1\mathrm{D}}^{(j_2-1)} \star c_{j_1}[k_r,\cdot]\right)\right).$$

- Approximation-Detail coefficients ($j_1 = J_1$ and $j_2 \leqslant J_2$):

$$w_{J_1,j_2}[k_r,\cdot] = \mathcal{A}_{J_1,j_2-1}\left(h_{1\mathrm{D}}^{(j_2-1)} \star c_{J_1}[k_r,\cdot]\right) - \mathcal{A}_{J_1,j_2}\left(h_{1\mathrm{D}}^{(j_2)} \star c_{J_1}[k_r,\cdot]\right).$$

- Detail-Approximation coefficients ($j_1 \leqslant J_1$ and $j_2 = J_2$):

$$w_{j_1,J_2}[k_r,\cdot] = \mathcal{A}_{j_1-1,J_2}\left(h_{1\mathrm{D}}^{(J_2)} \star c_{j_1-1}[k_r,\cdot]\right) - \mathcal{A}_{j_1,J_2}\left(h_{1\mathrm{D}}^{(J_2)} \star c_{j_1}[k_r,\cdot]\right).$$

- Approximation-Approximation coefficients ($j_1 = J_1$ and $j_2 = J_2$):

$$c_{J_1,J_2}[k_r,\cdot] = h_{1\mathrm{D}}^{(J_2)} \star c_{J_1}[k_r,\cdot].$$

$\mathcal{A}_{j_1,j_2}$ is the non-linear square root VST introduced in [102] (see Section 2.4). In summary, all 2D-1D wavelet coefficients $\{w_{j_1,j_2}\}_{j_1\leqslant J_1, j_2\leqslant J_2}$ are now stabilized, and the noise on all these wavelet coefficients is zero-mean Gaussian with known variance that depends solely on h on the resolution levels (j_1, j_2). As before, these variances can be easily tabulated.

6.4.2 The multichannel deconvolution problem

Many problems in signal and image processing can be cast as inverting the linear system:

$$Y = \mathbf{H}X + \varepsilon \ , \tag{117}$$

where $X \in \mathcal{X}$ is the data to recover, $Y \in \mathcal{Y}$ is the degraded noisy observation, ε is an additive noise, and $\mathbf{H} : \mathcal{X} \to \mathcal{Y}$ is a bounded linear operator which is typically ill-behaved since it models an acquisition process that encounters loss of information. When $\mathbf{H}$ is the identity, it is just a denoising problem which can be treated with the previously described methods. Inverting (117) is usually an ill-posed problem. This means that there is no unique and stable solution.

In the present case, the objective is to remove the effect of the instrument's PSF. $\mathbf{H}$ is the convolution operator by a blurring kernel (i.e. PSF) whose consequence is that Y lacks the high frequency content of X. Furthermore, since the noise is Poisson, ε has a variance profile $\mathbf{H}X$. The problem at hand is then a deconvolution problem in the presence of Poisson noise. As the PSF is channel-dependent, the convolution observation model is

$$Y[\cdot, k_t] = \mathbf{H}_{k_t} X[\cdot, k_t] + \varepsilon[\cdot, k_t] \ ,$$

in each channel k_t, where $\mathbf{H}_{k_t}$ is the (spatial) convolution operator in channel k_t with known PSF. In the case of the LAT, the PSF width depends strongly on the energy, from 6.9 at 50 MeV to better than 0.1 at 10 GeV and above. Figure 35 shows the normalized profiles of the PSF for different energy bands.

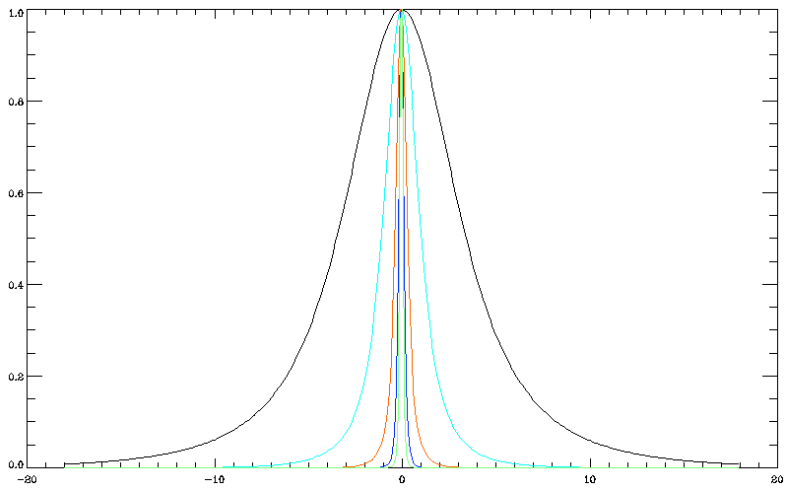


Fig. 35. Normalized profile of the PSF for different energy bands as a function of the angle in degree. ***Black:*** 50 MeV - 82 MeV. ***Cyan:*** 220 MeV - 360 MeV. ***Orange:*** 960 MeV - 1.6 GeV. ***Blue:*** 4.2 GeV - 6.9 GeV. ***Green:*** 19 GeV - 31 GeV.

This inversion can be performed using the well-known Richardson-Lucy algorithm with an additional regularization constraint from a multiresolution support [53]. Let $\mathbf{H}$ be the multichannel convolution operator, which acts on a 2D-1D multichannel spherical data set X by applying $\mathbf{H}_{k_t}$ on each channel $X[\cdot, k_t]$ independently [6]. The regularized multichannel Richardson-Lucy scheme proposed in [101] is

$$X^{(n+1)} = \mathbf{P}_+ \left(X^{(n)} \otimes \left(\mathbf{H}^T \left((\mathbf{H}X^{(n)} + \overline{R}^{(n)}) \oslash \mathbf{H}X^{(n)} \right) \right) \right) \ , \tag{118}$$

where $\otimes$ (resp. $\oslash$) stands for the element-wise multiplication (resp. division) between two vectors, $\mathbf{P}_+$ is the orthogonal projector onto the positive orthant and $\overline{R}^{(n)}$ is the regularized (significant) residual

$$\overline{R}^{(n)} = \mathcal{W}^{-1} M \mathcal{W} \left(Y - \mathbf{H}X^{(n)} \right) \ , \tag{119}$$

with $\mathcal{W}$ being the IUWTS and M being the multiresolution support defined similarly to (38) by selecting significant coefficient in the MS-VSTS of the data. Figure 36 shows the performance of the multichannel MS-VSTS de-

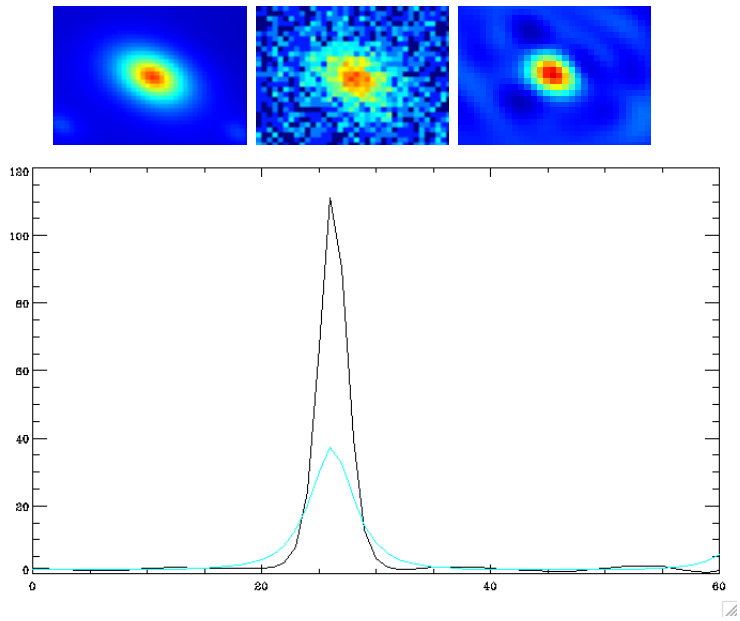


Fig. 36. Spectrum of a single gamma-ray point source recovered using the multichannel MS-VSTS deconvolution algorithm. ***Top:*** Single gamma-ray point source on simulated (blurred) Fermi data (energy band: 360 MeV - 589 MeV) (***left:*** simulated blurred source; ***middle:*** blurred noisy source; ***right:*** deconvolved source). ***Bottom:*** Spectrum profile of the center of the point source (***cyan:*** simulated spectrum; ***black:*** restored spectrum from the deconvolved source.

convolution algorithm on a single point source. The deconvolution not only

[6] If X were to be vectorized by stacking the channels in a long column vector, $\mathbf{H}$ would be a block-diagonal matrix whose blocks are the circulant matrices $\mathbf{H}_{k_t}$.

removes effectively the blur and recovers sharply localized point sources, but it also allows to restore the whole spectral information.

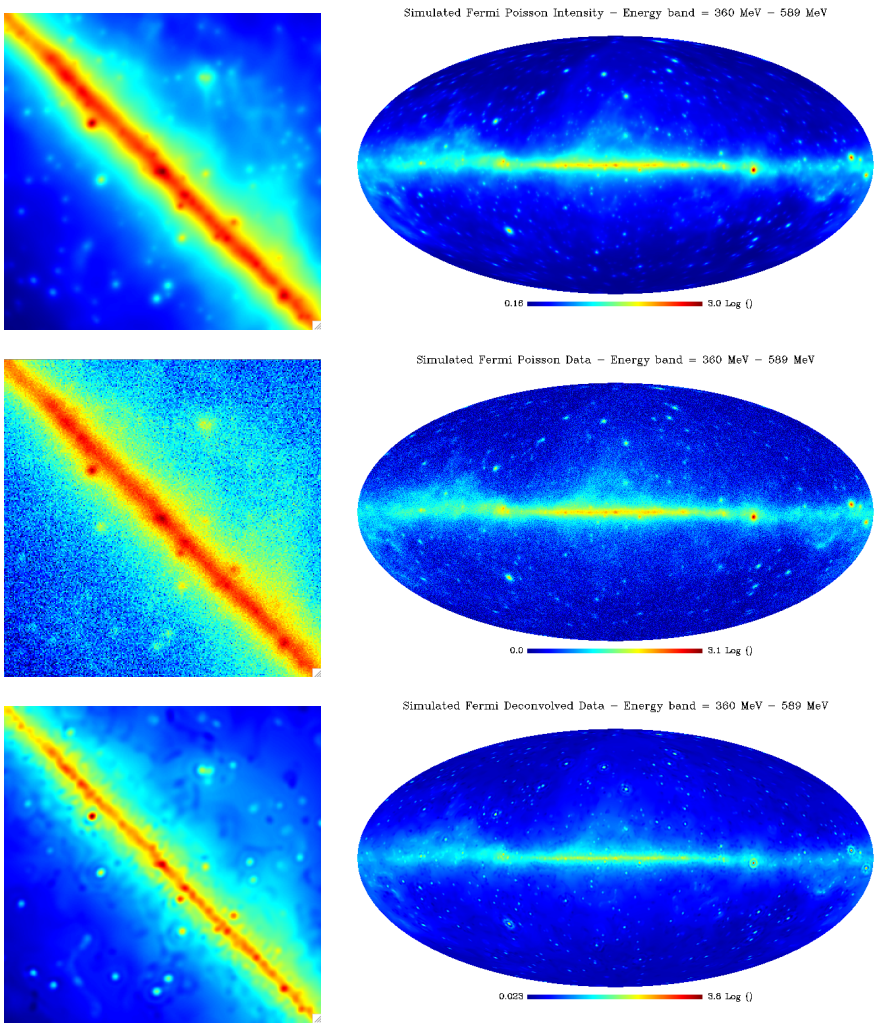


Fig. 37. Result of the deconvolution algorithm in the 360 MeV - 589 MeV energy band. The left images are single HEALPix faces covering the galactic plane. ***Top Left:*** Simulated Fermi Poisson intensity. ***Top Right:*** Simulated Fermi noisy data. ***Bottom:*** Fermi data deconvolved with multichannel MS-VSTS.

Figure 37 depicts the result of the multichannel MS-VSTS deconvolution algorithm in one energy band on the whole sky and on a single HEALPix face covering the galactic plane. The effect of the deconvolution is strikingly good. The MS-VSTS multichannel deconvolution algorithm manages to remove a large part of the blur introduced by the PSF.

7 3D Wavelets on the ball

In the previous sections, we have described multiresolution transforms for data provided in 3D Cartesian coordinates or on the 2D sphere. However, these transforms are not adapted to three-dimensional signals which are naturally expressed in spherical coordinates. Such signals arise for instance in astrophysics in the study of the 3D distribution of galaxies [103]-[104] for which we can have access to both angular position on the sky and distance along the line of sight. Recently two different wavelet transform for data in spherical coordinates (i.e. on the 3D ball) have been developed in [40] and [41]. These two transforms differ mainly in the harmonic expansion used to develop data on the ball.

The expansion introduced in [41] is based on exact sampling theorems in the angular domain based on [105] and in the radial domain based the orthogonality of Laguerre polynomials. The resulting Fourier-Laguerre transform allows for exact decomposition and reconstruction of band limited signals on the 3D ball and is used to implement a wavelet transform (named flaglets) with exact decomposition and reconstruction formulae. Due to the choice of independent basis for the radial and angular domains, flaglets probe independently angular and radial scales. However, separating angular and radial domains breaks the 3D translational invariance of the harmonic expansion. Indeed, depending on the radial position of an object of a given physical size, the apparent angular size will vary. Therefore, in the flaglet transform, the same object at different radial positions will be represented by wavelet coefficients of different angular scales.

In this section, we present the approach of [40] which is based on the natural harmonic expansion of data in spherical coordinate using the spherical Fourier-Bessel transform. Using this transform, equivalent to a Fourier transform in spherical coordinates, the link between angular and radial scales is preserved. The drawback of this transform however is that no exact sampling theorem exists in the radial domain [106]. Contrary to the Fourier-Laguerre transform, the spherical Fourier-Bessel Transform cannot be computed exactly for a discretely sampled band limited signal on the ball. To circumvent this issue, a discrete spherical Fourier-Bessel Transform was introduced in [40] which allows the evaluation of this transform to any desired accuracy. Using the spherical Fourier-Bessel transform presented in the following section, an Isotropic Undecimated Spherical 3D Wavelet Transform similar to the IUWT on the sphere (see section 6.2) will be derived in section 7.3. This wavelet transform is exact in the spherical Fourier-Bessel domain and wavelet coefficients can be recovered in the direct domain using the discrete spherical Fourier-Bessel transform that will be described in section 7.2.2.

7.1.1 The Spherical Fourier-Bessel Transform

In the same way that the natural expansion of a function on the sphere are the spherical harmonics, the natural expansion of a function on the ball is the spherical Fourier-Bessel transform. This transform consists in the projection of a function $f \in L_2(\mathbb{R}^3)$ onto a set of orthogonal functions $\Psi_{lmk}(r, \theta, \varphi)$, composed of Spherical Harmonics and Spherical Bessel functions:

$$\forall l \in \mathbb{N}, \forall m \in [\![-l, l]\!], \forall k \in \mathbb{R}^+, \quad \Psi_{lmk}(r, \theta, \varphi) = \sqrt{\frac{2}{\pi}} j_l(kr) Y_{lm}(\theta, \varphi) \ , \quad (120)$$

where Y_{lm} are the spherical harmonics introduced in Section 6.1.2 and j_l are spherical Bessel functions of the first kind. These functions can be expressed in terms of the ordinary Bessel functions of the first kind J_ν for all $l \in \mathbb{N}$ and all $r \in \mathbb{R}^+$:

$$j_l(r) = \sqrt{\frac{\pi}{2r}} J_{l+1/2}(r) \ , \quad (121)$$

where J_ν is defined for $z \in \mathbb{C}$ and $\nu \in \mathbb{R}$ as:

$$J_\nu(z) = \sum_{k=0}^{\infty} \frac{(-1)^k z^{\nu+2k}}{2^{\nu+2k} k! \Gamma(\nu + k + 1)} \ . \quad (122)$$

Just as the spherical harmonics verify the orthonormality relation (96), the spherical Bessel functions are orthogonal:

$$\forall k, k' \in \mathbb{R}^+, \quad \int_0^\infty j_l(kr) j_l(k'r) r^2 dr = \frac{\pi}{2k^2} \delta(k - k') \ . \quad (123)$$

Using the orthogonality relations of both Spherical Harmonics and Spherical Bessel functions, the orthogonality of the Ψ_{lmk} is easily derived:

$$\begin{aligned} \int \Psi^*_{lmk}(\mathbf{r}) \Psi_{l'm'k'}(\mathbf{r}) d\mathbf{r} &= \frac{2}{\pi} \int j_l(k'r) j_l(kr) r^2 dr \int_\Omega Y^*_{lm}(\theta, \varphi) Y_{l'm'}(\theta, \varphi) d\Omega \\ &= \frac{1}{k^2} \delta(k - k') \delta_{ll'} \delta_{mm'} \ . \end{aligned} \quad (124)$$

From this relation, the spherical Fourier-Bessel transform of $f \in L_2(\mathbb{R}^3)$ is

uniquely defined by the projection of f on the $\{\Psi_{lmk}\}$:

$$\begin{aligned}\tilde{f}_{lm}(k) = \langle f, \Psi_{lmk}\rangle &= \int \Psi^*_{lmk}(r,\theta,\varphi) f(r,\theta,\varphi) r^2 \sin(\theta) d\theta d\varphi dr \\ &= \int_0^{2\pi}\int_0^{\pi}\left[\sqrt{\frac{2}{\pi}}\int_0^{\infty} f(r,\theta,\varphi) j_l(kr) r^2 dr\right] Y^*_{lm}(\theta,\varphi)\sin(\theta) d\theta d\varphi \quad (125)\\ &= \sqrt{\frac{2}{\pi}}\int_0^{\infty}\left[\int_0^{2\pi}\int_0^{\pi} f(r,\theta,\varphi) Y^*_{lm}(\theta,\varphi)\sin(\theta) d\theta d\varphi\right] j_l(kr) r^2 dr\ . \quad (126)\end{aligned}$$

In this expression, one can recognize the commutative composition of two transforms: a spherical harmonics transform in the angular domain and a *spherical Bessel transform* (SBT) in the radial domain. We define the SBT and its inverse as:

$$\tilde{f}_l(k) = \sqrt{\frac{2}{\pi}}\int f(r) j_l(kr) r^2 dr \quad (127a)$$

$$f(r) = \sqrt{\frac{2}{\pi}}\int \tilde{f}_l(k) j_l(kr) k^2 dk\ . \quad (127b)$$

In the following, the notation $\tilde{f}_l$ denotes the SBT of order l of a 1D function f.

The inversion formula for the Spherical Fourier-Bessel Transform is as follows:

$$f(r,\theta,\varphi) = \sqrt{\frac{2}{\pi}}\sum_{l=0}^{\infty}\sum_{m=-l}^{l}\int \tilde{f}_{lm}(k) k^2 j_l(kr) dk Y_{lm}(\theta,\varphi)\ . \quad (128)$$

7.1.2 Convolution in the Spherical Fourier-Bessel domain

A key point to the spherical Fourier-Bessel transform is the existence of an expression for the real space convolution $h = f * g$ of two functions $f, g \in L_2(\mathbb{R}^3)$ which reduces to a very simple formula in the case of an isotropic function g. The convolution in the Spherical Fourier-Bessel domain can be expressed from the well known expression in the Fourier domain:

$$\begin{aligned}\hat{h}(k,\theta_k,\varphi_k) &= \mathbf{F}\{f * g\}(k,\theta_k,\varphi_k) \\ &= \sqrt{(2\pi)^3}\hat{f}(k,\theta_k,\varphi_k)\hat{g}(k,\theta_k,\varphi_k)\ ,\end{aligned} \quad (129)$$

using the following unitary convention for the Fourier Transform :

$$\hat{f}(\boldsymbol{k}) = \frac{1}{\sqrt{(2\pi)^3}}\int f(\boldsymbol{r}) e^{-i\boldsymbol{k}.\boldsymbol{r}} d\boldsymbol{r}\ , \qquad f(\boldsymbol{r}) = \frac{1}{\sqrt{(2\pi)^3}}\int \hat{f}(\boldsymbol{k}) e^{i\boldsymbol{k}.\boldsymbol{r}} d\boldsymbol{k}\ . \quad (130)$$

To relate Fourier and spherical Fourier-Bessel coefficients, one can use the expansion of the Fourier kernel in spherical coordinates:

$$e^{-i\boldsymbol{k}.\boldsymbol{r}} = 4\pi \sum_{l=0}^{\infty} \sum_{m=-l}^{l} (-i)^l j_l(kr) Y_{lm}^*(\theta_r, \varphi_r) Y_{lm}(\theta_k, \varphi_k) \ . \tag{131}$$

When injected in the definition of the Fourier transform, this expression directly leads to the following relation between Fourier and spherical Fourier-Bessel transforms:

$$\hat{f}(k, \theta_k, \varphi_k) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \left[(-i)^l \tilde{f}_{lm}(k)\right] Y_l^m(\theta_k, \varphi_k) \ . \tag{132}$$

It is worth noticing that the spherical Fourier-Bessel transform $\tilde{f}_{lm}(k)$ is merely a Spherical Harmonics transform applied on shells of radii k in Fourier space (up to a factor $(-i)^l$): $\hat{f}_{lm}(k) = (-i)^l \tilde{f}_{lm}(k)$.

This expression for the Fourier transform combined with the convolution equation (129) yields the convolution formula in spherical Fourier-Bessel domain (see Appendix A.2 [40] for the full derivation):

$$\begin{aligned} \tilde{h}_{lm}(k) = (i)^l \sqrt{(2\pi)^3} \sum_{l'=0}^{\infty} \sum_{m'=-l'}^{l'} (-i)^{l'} \tilde{f}_{l'm'}(k) \\ \times \sum_{l''=|l-l'|}^{l+l'} c^{l''}(l, m, l', m')(-i)^{l''} \tilde{g}_{l''m-m'}(k) \ , \end{aligned} \tag{133}$$

where $c^{l''}(l, m, l', m')$ are Slater integrals:

$$c^{l''}(l, m, l', m') = \iint Y_{lm}^*(\theta, \varphi) Y_{l'm'}(\theta, \varphi) Y_{l''}^{m-m'}(\theta, \varphi) d\Omega \ . \tag{134}$$

These integrals are only non-zero for $|l - l'| \leq l'' \leq l + l'$.

As already mentioned, this expression reduces to a simple form when g is isotropic. In this case, g has no angular dependence in the Fourier domain therefore $\hat{g}$ is constant on spherical shells and $\hat{g}_{lm}(k) = 0 = \tilde{g}_{lm}(k)$ for all $(l, m) \neq (0, 0)$. Then, knowing that $c^0(l, m, l, m) = 1/\sqrt{4\pi}$ equation (133) becomes:

$$\tilde{h}_{lm}(k) = \sqrt{2}\pi \tilde{g}_{00}(k) \tilde{f}_{lm}(k) \ . \tag{135}$$

This expression can therefore be used to express in the spherical Fourier-Bessel domain a convolution by any isotropic filter g.

The transform introduced so far yields a natural discretization in the angular domain thanks to the spherical harmonics, however in the radial domain, the spherical Bessel transform is purely continuous. In order to implement wavelets in the harmonic domain and to be able to compute wavelet coefficients in the direct domain, a discretization scheme for the spherical Bessel transform is required. The main difficulty comes from the lack of an exact quadrature formula for this radial transform and therefore the lack of an exact sampling theorem. To circumvent this issue, we propose an approximated discrete spherical Bessel transform for a radially limited signal, extension of the discrete Bessel Transform introduced in [106]. Although this discrete transform is not exact, it can be evaluated to any desired accuracy by increasing the number of sampling points. Combined with the HEALpix [98] sampling in the angular domain (see section 6.1.1) we build a sampling grid in spherical coordinates which allows for back and forth computation of the spherical Fourier-Bessel transform.

7.2.1 The 1D Discrete Spherical Bessel Transform

The transform described here is an extension to the spherical Bessel transform of the discrete Bessel transform from [106]. The discretization of the spherical Bessel transform uses the well known orthogonality property of the Spherical Bessel functions on the interval $[0, R]$. If f is a continuous function defined on $[0, R]$ which verifies the boundary condition $f(R) = 0$ then the spherical Bessel transform defined Eq. (127) can be expressed using Spherical Fourier-Bessel series:

$$\tilde{f}_l(k_{ln}) = \sqrt{\frac{2}{\pi}} \int_0^R f(r) j_l(k_{ln} r) r^2 dr \tag{136a}$$

$$f(r) = \sum_{n=1}^{\infty} \tilde{f}_l(k_{ln}) \rho_{ln} j_l(k_{ln} r) \ . \tag{136b}$$

In this expression, $k_{ln} = \frac{q_{ln}}{R}$ where q_{ln} is the nth zero of the Bessel function of the first kind of order l and the weights ρ_{ln} are defined as:

$$\rho_{ln} = \frac{\sqrt{2\pi} R^{-3}}{j_{l+1}^2(q_{ln})} \ . \tag{137}$$

Although this formulation provides a discretization of the inverse Spherical Bessel Transform and of the k spectrum, the direct transform is still continuous and another discretization step is necessary. Assuming that a boundary condition of the same kind can applied to $\tilde{f}_l(k)$ so that $\tilde{f}_l(K_l) = 0$, then by us-

ing the same result, the spherical Fourier-Bessel expansion of $\tilde{f}_l(k)$ is obtained by:

$$\tilde{\tilde{f}}_l(r_{ln}) = \sqrt{\frac{2}{\pi}} \int_0^K \tilde{f}_l(k) j_l(r_{ln}k) k^2 dk \tag{138a}$$

$$\tilde{f}_l(k) = \sum_{n=1}^{\infty} \tilde{\tilde{f}}_l(r_{ln}) \kappa_{ln} j_l(r_{ln}k) \ , \tag{138b}$$

where $r_{ln} = \frac{q_{ln}}{K_l}$ and where the weights ρ_{ln} are defined as:

$$\kappa_{ln} = \frac{\sqrt{2\pi} K_l^{-3}}{j_{l+1}^2(q_{ln})} \ . \tag{139}$$

The spherical Bessel transform being an involution, $\tilde{\tilde{f}} = f$ so that $\tilde{\tilde{f}}_l(r_{ln}) = f(r_{ln})$. Much like the previous set of equations had introduced a discrete k_{ln} grid, a discrete r_{ln} grid is obtained for the radial component. Since equations (136b) and (138b) can be used to compute f and $\tilde{f}_l$ for any value of r and k, they can in particular be used to compute $f(r_{ln})$ and $\tilde{f}_l(r_{l'n})$ where l' does not have to match l. The Spherical Bessel Transform and its inverse can then be expressed only in terms of series:

$$\tilde{f}_l(k_{l'n}) = \sum_{p=1}^{\infty} f(r_{lp}) \kappa_{lp} j_l(r_{lp} k_{l'n}) \tag{140a}$$

$$f(r_{l'n}) = \sum_{p=1}^{\infty} \tilde{f}_l(k_{lp}) \rho_{lp} j_l(r_{l'n} k_{lp}) \ . \tag{140b}$$

Thanks to this last set of equations one can compute the Spherical Bessel Transform and its inverse without the need of evaluating any integral. Furthermore only discrete values of f and $\hat{f}$ respectively sampled on r_{ln} and k_{ln} are required.

However, this expression of the direct and inverse spherical Bessel transform is only valid if f is band limited ($\tilde{f}_l(K_l) = 0$) and radially limited ($f(R) = 0$) at the same time. It is well known that these two conditions can never be verified at the same time. The same problem arises for the Fourier transform, a band limited signal necessarily has an infinite time support. In practice, by increasing the band limit K_l to any arbitrary value, one can recover an approximation of the exact transform to any required accuracy.
The second difficulty comes from the infinite sums over p in equations (140a) and (140b). In practical applications, for a given value of l only a limited number N of $\tilde{f}_l(k_{ln})$ and $f(r_{ln})$ coefficients can be stored so that $r_{lN} = R$ and $k_{lN} = K_l$. Since r_{ln} is defined by $r_{ln} = \frac{q_{ln}}{K_l}$, for $n = N$, R and K_l are bound by the following relation:

$$q_{lN} = K_l R \ . \tag{141}$$

Therefore, the value of K_l is fixed for a choice of N and R.
The main point is that any desired accuracy in the evaluation of the direct and inverse transform can be reached by increasing the number of points N and artificially increasing R above the actual radial limit of the signal.

The truncation of the direct and inverse series to N coefficients yields a convenient matrix formulation for the discrete spherical Bessel transform and its inverse. Defining a transform matrix $T^{ll'}$ as:

$$T^{ll'}_{pq} = \left(\frac{\sqrt{2\pi}}{j^2_{l+1}(q_{lq})} j_l(\frac{q_{l'p} q_{lq}}{q_{lN}}) \right)_{pq} . \tag{142}$$

The direct transform can be expressed as:

$$\begin{bmatrix} \tilde{f}_l(k_{l'1}) \\ \tilde{f}_l(k_{l'2}) \\ \vdots \\ \tilde{f}_l(k_{l'N}) \end{bmatrix} = \frac{1}{K_l^3} T^{ll'} \begin{bmatrix} f(r_{l1}) \\ f(r_{l2}) \\ \vdots \\ f(r_{lN}) \end{bmatrix} . \tag{143}$$

Reciprocally, the inverse the values of f can be computed on any $r_{l'n}$ grid from $\tilde{f}_l$ sampled on $k_{l'n}$ using the exact same matrix:

$$\begin{bmatrix} f(r_{l'1}) \\ f(r_{l'2}) \\ \vdots \\ f(r_{l'N}) \end{bmatrix} = \frac{1}{R^3} T^{ll'} \begin{bmatrix} \tilde{f}_l(k_{l1}) \\ \tilde{f}_l(k_{l2}) \\ \vdots \\ \tilde{f}_l(k_{lN}) \end{bmatrix} . \tag{144}$$

The Discrete Spherical Bessel Transform is defined by the set of equations (143) and (144).

Finally, it can be shown [40] that spherical Bessel transforms of different orders can be related through the following equation:

$$\tilde{f}_l(k_{ln}) = \sum_{m=1}^{\infty} \tilde{f}_{l_0}(k_{l_0 m}) \frac{2}{j^2_{l_0+1}(q_{l_0 m})} W^{l_0 l}_{nm} . \tag{145}$$

where the weights $W^{ll'}_{nm}$ are defined as:

$$W^{l_0 l}_{nm} = \int_0^1 j_{l_0}(q_{l_0 m} x) j_l(q_{ln} x) x^2 dx . \tag{146}$$

Therefore, the Spherical Bessel Transform of a given order can be expressed as the sum of the coefficients obtained for a different order of the transform,

with the appropriate weighting. It is also worth noticing that the weights $W_{nm}^{ll'}$ are independent of the problem and can be tabulated. Using this relationship between orders, it will be possible to convert the Spherical Bessel coefficients of order l_0 into coefficients of any other order l, which will prove useful for the implementation of the full discrete spherical Fourier-Bessel transform.

7.2.2 The 3D Discrete Spherical Fourier-Bessel Transform

As presented in section 7.1, the Spherical Fourier-Bessel Transform is the composition of a Spherical Harmonics Transform for the angular component and a Spherical Bessel Transform for the radial component. Since these two transforms can commute, they can be treated independently and by combining discrete algorithms for both transforms, one can build a Discrete Spherical Fourier-Bessel Transform. A convenient choice for the angular part of the transform is the HEALPix [98] pixelization scheme. The radial component can be discretized using the Discrete Spherical Bessel Transform algorithm presented in the previous section. The choice of these two algorithms introduces a discretization of the Fourier-Bessel coefficients as well as a pixelization of the 3D space in spherical coordinates.

The Spherical Fourier-Bessel coefficients $\tilde{f}_{lm}(k)$ are defined by Equation (125) for continuous values of k. Assuming a boundary condition on the density field f, the Discrete Spherical Bessel Transform can be used to discretize the values of k. The Discrete Spherical Fourier-Bessel coefficients are therefore defined as:

$$a_{lmn} = \tilde{f}_{lm}(k_{ln}) \ , \tag{147}$$

for $0 \leq l \leq L_{max}$, $-l \leq m \leq l$ and $1 \leq n \leq N_{max}$. These discrete coefficients are simply obtained by sampling the continuous coefficients on the k_{ln} grid introduced in the previous section.

To this discretized Fourier-Bessel spectrum corresponds a dual grid of the 3D space defined by combining the HEALPix pixelization scheme and the Discrete Spherical Bessel Transform.

In the angular domain, for a given value of r, the field $f(r, \theta, \varphi)$ can be sampled on a finite number of points using HEALPix. The radial component of the transform is conveniently performed using the Discrete Spherical Bessel Transform. Indeed, this algorithm introduces a radial grid compatible with the discretized k_{ln} spectrum. Although this radial grid r_{ln} depends on the order l of the Spherical Bessel Transform, it will be justified in the next section that only one grid $r_{l_0 n}$ is required to sample the field along the radial dimension. The value of l_0 is set to 0 because in this case the properties of the zeros of

the Bessel function ensure that r_{0n} will be regularly spaced between 0 and R:

$$r_{0n} = \frac{n}{N_{max}} R \ . \tag{148}$$

For given values of θ_i and φ_j, the field $f(r, \theta_i, \varphi_j)$ can now be sampled on discrete values of $r = r_{0n}$.

Combining angular and radial grids, the 3D spherical grid is defined as a set of N_{max} HEALPix maps equally spaced between 0 and R. An illustration of this grid is provided on Fig.38 where only one quarter of the space is represented for clarity.

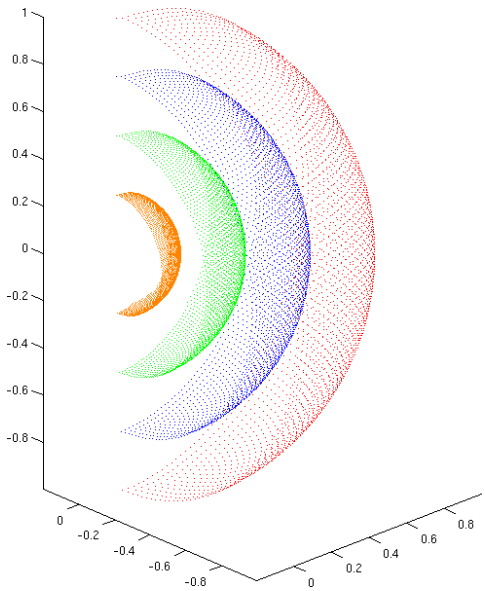


Fig. 38. Representation of the spherical 3D grid for the Discrete Spherical Fourier-Bessel Transform ($R = 1$ and $N_{max} = 4$)

Using this 3D grid it becomes possible to compute back and forth the Spherical Fourier-Bessel Transform between a density field and its Spherical Fourier-Bessel coefficients. Here, a detailed description of the algorithm for both the direct and inverse Discrete Spherical Fourier-Bessel Transform is provided below.

Inverse Transform Let a_{lmn} be the coefficients of the Spherical Fourier-Transform of the density field f. The reconstruction of f on the spherical 3D grid requires two steps:

1) First, from the a_{lmn}, the inverse Discrete Spherical Bessel Transform is computed for all l and m. This transform can be easily evaluated thanks to

a matrix product:

$$\begin{cases} \forall \ 0 \ \leq \ l \ \leq L_{max} \\ \forall \ -l \leq m \leq \ \ l \end{cases}, \quad \begin{bmatrix} f_{lm}(r_{l_0 1}) \\ f_{lm}(r_{l_0 2}) \\ \vdots \\ f_{lm}(r_{l_0 N_{max}}) \end{bmatrix} = \frac{T^{ll_0}}{R^3} \begin{bmatrix} a_{lm1} \\ a_{lm2} \\ \vdots \\ a_{lmN_{max}} \end{bmatrix}. \tag{149}$$

Here, it is worth noticing that the matrix T^{ll_0} allows the evaluation of the Spherical Bessel Transform of order l and provides the results on the grid of order l_0.

2) From the spherical harmonics coefficients $f_{lm}(r_{l_0 n})$ given at specific radial distances $r_{l_0 n}$ it is possible to compute the inverse Spherical Harmonics Transform. For each n between 1 and N_{max} the HEALPix inverse Spherical Harmonics Transform is performed on the set of coefficients $\{f_{lm}(r_{l_0 n})\}_{l,m}$. This yields N_{max} HEALPix maps which constitute the sampling of the reconstructed density field on the 3D spherical grid.

Direct Transform Given a density field f sampled on the spherical 3D grid, the Spherical Fourier-Bessel coefficients a_{lmn} are computed in three steps:

1) For each n between 1 and N_{max} the Spherical Harmonics Transform of the HEALPix map of radius $r_{l_0 n}$ is computed. This yields $f_{lm}(r_{l_0 n})$ coefficients.

2) The next step is to compute the Spherical Bessel Transform of order l_0 from the $f_{lm}(r_{l_0 n})$ coefficients for every (l, m). Again, this operation is a simple matrix product:

$$\begin{cases} \forall \ 0 \ \leq \ l \ \leq L_{max} \\ \forall \ -l \leq m \leq \ \ l \end{cases}, \quad \begin{bmatrix} \tilde{f}^{l_0}_{lm}(k_{l_0 1}) \\ \tilde{f}^{l_0}_{lm}(k_{l_0 2}) \\ \vdots \\ \tilde{f}^{l_0}_{lm}(k_{l_0 N_{max}}) \end{bmatrix} = \frac{T^{l_0 l_0}}{K^3} \begin{bmatrix} f_{lm}(r_{l_0 1}) \\ f_{lm}(r_{l_0 2}) \\ \vdots \\ f_{lm}(r_{l_0 N_{max}}) \end{bmatrix}. \tag{150}$$

This operation yields $\hat{f}^{l_0}_{lm}(k_{l_0 n})$ coefficients which are not yet Spherical Fourier-Bessel coefficients because the order of the Spherical Bessel transform l_0 does not match the order of the Spherical Harmonics coefficients l. An additional step is necessary.

3) The last step required to gain access to the Spherical Fourier-Bessel coefficients a_{lmn} is to convert the Spherical Bessel coefficients for order l_0 to the

correct order l that matches the Spherical Harmonics order. This is done by using relation (145):

$$\begin{cases} \forall \ 0 \ \leq \ l \ \leq L_{max} \\ \forall \ -l \leq m \leq \quad l \\ \forall \ 1 \ \leq \ n \ \leq N_{max} \end{cases} , \tilde{f}_{lm}(k_{ln}) = \sum_{p=1}^{N_{max}} \tilde{f}_{lm}^{l_0}(k_{l_0 p}) \frac{2W_{np}^{l_0 l}}{j_{l_0+1}^2(q_{lp})} \ , \tag{151}$$

where $W_{np}^{l_0 l}$ are defined by Eq. (146). This operation finally yields the $a_{lmn} = \tilde{f}_{lm}(k_{ln})$ coefficients.

7.3 Isotropic Undecimated Spherical 3D Wavelet Transform

The aim of this section is to transpose the ideas behind the Isotropic Undecimated Wavelet Transform on the Sphere introduced in Section 6.2 to the case of data in 3D spherical coordinates. Indeed, the isotropic wavelet transform can be fully defined using isotropic filters which are simple to express in the spherical Fourier-Bessel domain as seen in section 7.1.2. Furthermore, the issue of the practical evaluation of the direct and inverse spherical Fourier-Bessel transform was addressed in the previous section.

7.3.1 Wavelet decomposition

The Isotropic Undecimated Spherical 3D Wavelet Transform is based on a scaling function $\varphi^{k_c}(r, \theta_r, \varphi_r)$ with cut-off frequency k_c and spherical symmetry. The symmetry of this function is preserved in the Fourier space and therefore, its Spherical Fourier-Bessel Transform verifies $\tilde{\varphi}_{lm}^{k_c}(k) = 0$ as soon as $(l, m) \neq (0, 0)$. Furthermore, due to its cut-off frequency, the scaling function verifies $\tilde{\varphi}_{00}^{k_c}(k) = 0$ for all $k \geq k_c$. In other terms, the scaling function verifies:

$$\phi^{k_c}(r, \theta_r, \varphi_r) = \phi^{k_c}(r) = \sqrt{\frac{2}{\pi}} \int_0^{k_c} \tilde{\phi}_{00}^{k_c}(k) k^2 j_0(kr) dk Y_0^0(\theta_r, \varphi_r) \ . \tag{152}$$

Using relation (135) the convolution of the original data $f(r, \theta, \varphi)$ with ϕ^{k_c} becomes very simple:

$$\tilde{c}_{lm}^0(k) = [\widetilde{\phi_{k_c} * f}]_{lm}(k) = \sqrt{2}\pi \tilde{\phi}_{00}^{k_c}(k) \tilde{f}_{lm}(k) \ . \tag{153}$$

Thanks to this scaling function, it is possible to define a sequence of smoother approximations $c^j(r, \theta_r, \varphi_r)$ of a function $f(r, \theta_r, \varphi_r)$ on a dyadic resolution scale. Let $\phi^{2^{-j}k_c}$ be a rescaled version of ϕ^{k_c} with cut-off frequency $2^{-j}k_c$.

Then $c^j(r, \theta_r, \varphi_r)$ is obtained by convolving $f(r, \theta_r, \varphi_r)$ with $\phi^{2^{-j}k_c}$:

$$\begin{aligned} c^0 &= \phi^{k_c} * f \\ c^1 &= \phi^{2^{-1}k_c} * f \\ &\cdots \\ c^j &= \phi^{2^{-j}k_c} * f \ . \end{aligned}$$

Applying the Spherical Fourier-Bessel Transform to the last relation yields:

$$\tilde{c}^j_{lm}(k) = \sqrt{2}\pi \tilde{\phi}^{2^{-j}k_c}_{00}(k) \tilde{f}_{lm}(k) \ . \tag{154}$$

This leads to the following recurrence formula :

$$\forall k < \frac{k_c}{2^j}, \quad \tilde{c}^{j+1}_{lm}(k) = \frac{\tilde{\phi}^{2^{-(j+1)}k_c}_{00}(k)}{\tilde{\phi}^{2^{-j}k_c}_{00}(k)} \tilde{c}^j_{lm}(k) \ . \tag{155}$$

Just like for the starlet algorithm (see section 2.2), the wavelet coefficients $\{w^j\}$ can be defined as the difference between two consecutive resolutions:

$$w^{j+1}(r, \theta_r, \varphi_r) = c^j(r, \theta, \varphi) - c^{j+1}(r, \theta, \varphi) \ . \tag{156}$$

This choice for the wavelet coefficients is equivalent to the following definition for the wavelet function ψ^{k_c} :

$$\tilde{\psi}^{2^{-j}k_c}_{lm}(k) = \tilde{\phi}^{2^{-(j-1)}k_c}_{lm}(k) - \tilde{\phi}^{2^{-j}k_c}_{lm}(k) \ , \tag{157}$$

so that :

$$\begin{aligned} w^0 &= \psi^{k_c} * f \\ w^1 &= \psi^{2^{-1}k_c} * f \\ &\cdots \\ w^j &= \psi^{2^{-j}k_c} * f \ . \end{aligned}$$

By applying the Spherical Fourier-Bessel Transform to the definition of the wavelet coefficients and using the recurrence formula verified by the c^js yields:

$$\forall k < \frac{k_c}{2^j}, \quad \tilde{w}^{j+1}_{lm}(k) = \left(1 - \frac{\tilde{\phi}^{2^{-(j+1)}k_c}_{00}(k)}{\tilde{\phi}^{2^{-j}k_c}_{00}(k)}\right) \tilde{c}^j_{lm}(k) \ . \tag{158}$$

Equations (158) et (155) which define the wavelet decomposition are in fact equivalent to convolving the resolution at a given scale j with a low-pass and a high-pass filter in order to obtain respectively the resolution and the wavelet coefficients at scale $j + 1$.

The low-pass filter h^j can be defined for each scale j by :

$$\tilde{h}^j_{lm}(k) = \begin{cases} \frac{\tilde{\phi}_{00}^{2^{-(j+1)}k_c}(k)}{\tilde{\phi}_{00}^{2^{-j}k_c}(k)} & \text{if } k < \frac{k_c}{2^{j+1}} \text{ and } l = m = 0 \\ 0 & \text{otherwise .} \end{cases} \tag{159}$$

Then the approximation at scale $j+1$ is given by the convolution of scale j with h^j :

$$c^{j+1} = c^j * \frac{1}{\sqrt{2}\pi} h^j \ . \tag{160}$$

In the same way, a high pass filter g^j can be defined on each scale j by:

$$\tilde{g}^j_{lm}(k) = \begin{cases} \frac{\tilde{\psi}_{00}^{2^{-(j+1)}k_c}(k)}{\tilde{\phi}_{00}^{2^{-j}k_c}(k)} & \text{if } k < \frac{k_c}{2^{j+1}} \text{ and } l = m = 0 \\ 1 & \text{if } k \geq \frac{k_c}{2^{j+1}} \text{ and } l = m = 0 \\ 0 & \text{otherwise .} \end{cases} \tag{161}$$

Given the definition of ψ, g^j can also be expressed in the simple form :

$$\tilde{g}^j_{lm}(k) = 1 - \tilde{h}^j_{lm}(k) \ . \tag{162}$$

The wavelet coefficients at scale $j+1$ are obtained by convolving the resolution at scale j with g^j:

$$w^{j+1} = c^j * \frac{1}{\sqrt{2}\pi} g^j \ . \tag{163}$$

To sum-up, the two relations necessary to recursively define the wavelet transform are:

$$\begin{aligned} \tilde{c}^{j+1}_{lm}(k) &= \tilde{h}^j_{00}(k)\tilde{c}^j_{lm}(k) \\ \tilde{w}^{j+1}_{lm}(k) &= \tilde{g}^j_{00}(k)\tilde{c}^j_{lm}(k) \ . \end{aligned} \tag{164}$$

7.3.2 Choice of a scaling function

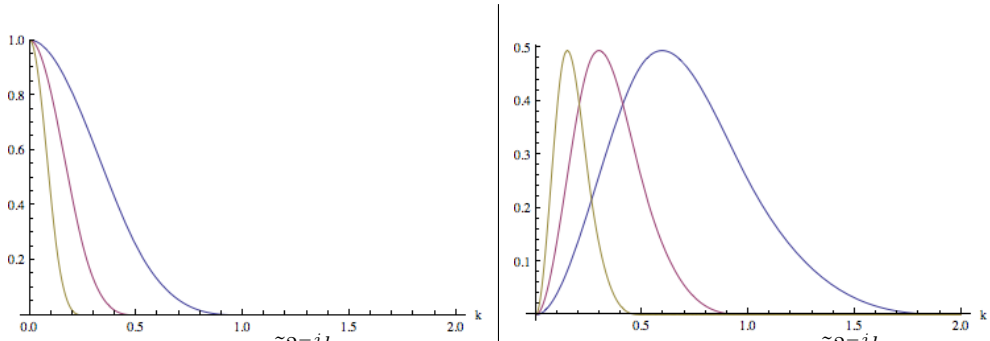
Any function with spherical symmetry and a cut-off frequency k_c would do as a scaling function but in this work we choose to use a B-spline function of order 3 to define our scaling function:

$$\tilde{\phi}^{k_c}_{lm}(k) = \frac{3}{2} B_3 \left(\frac{2k}{k_c} \right) \delta_{l0}\delta_{m0} \ . \tag{165}$$

where

$$B_3(x) = \frac{1}{12}\left(|x-2|^3 - 4|x-1|^3 + 6|x|^3 - 4|x+1|^3 + |x+2|^3\right) . \qquad (166)$$

The scaling function and its corresponding wavelet function are plotted in Spherical Fourier-Bessel space for different values of j in Fig. 39



(a) Scaling function $\tilde{\phi}_{00}^{2^{-j}k_c}(k)$ for $j = 0, 1, 2$

(b) Wavelet function $\tilde{\psi}_{00}^{2^{-j}k_c}(k)$ for $j = 0, 1, 2$

Fig. 39. Scaling function and Wavelet function for $k_c = 1$

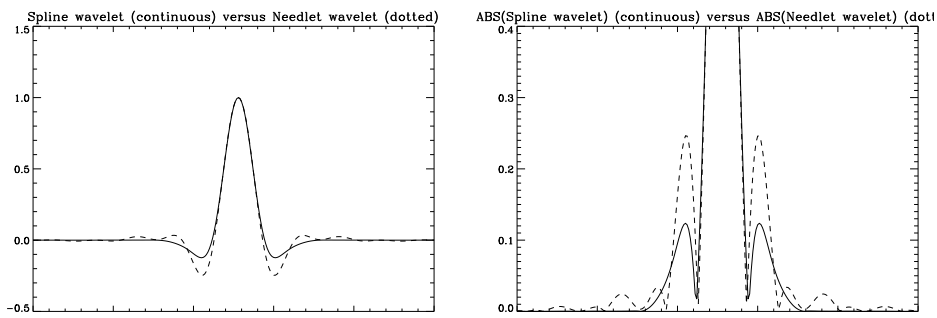


Fig. 40. Comparaison between spline, needlet, and Meyer wavelet functions on the sphere.

Other functions such as Meyer wavelets or the needlet function [30] can be used as well. Needlet wavelet functions have 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 Figure 40 two different wavelet functions. Figure 40 left shows 1D profile of the spline (continuous line) and the needlet wavelet function (dotted line) at a given scale. Figure 40 right shows the same function, 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. 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 needlet shoud be preferred to the spline wavelet.

The complete algorithm for the Isotropic Undecimated Spherical 3D Wavelet Transform is provided in Algorithm 13. This algorithm makes use the discrete Spherical Fourier-Bessel transform described in section 7.2.2. Using this transform, the spherical Fourier-Bessel coefficients are now sampled at discrete k_{ln} values and we note $\tilde{f}_{lm}(k_{ln}) = \tilde{f}_{lmn}$.

To illustrate this wavelet transform, a set of Spherical Fourier-Bessel Coefficients was extracted from a 3D density field using the Discrete Spherical Fourier-Bessel Transform described in the next section. The test density field was provided by a cosmological n-body simulation which was carried out by the Virgo Supercomputing Consortium using computers based at Computing Centre of the Max-Planck Society in Garching and at the Edinburgh Parallel Computing Centre. The data is publicly available at http://www.mpa-garching.mpg.de/Virgo/VLS.html.
The wavelet decomposition presented above can then be computed from the Spherical Fourier-Bessel coefficients of the test density field and yields the Spherical Fourier-Bessel coefficients of the various wavelet scales and smoothed away density. Using the inverse Discrete Spherical Fourier-Bessel Transform, the actual wavelet coefficients can be retrieved in the form of 3D density fields. These density fields are shown on Figure 41.

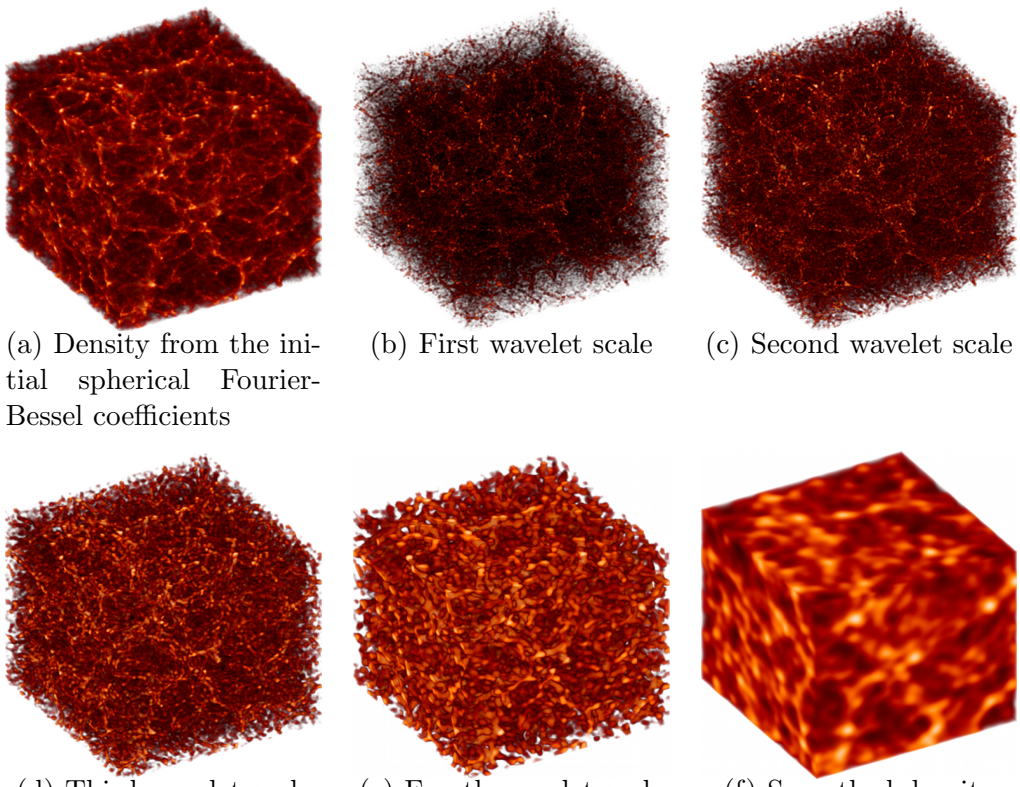
Algorithm 13: The Isotropic Undecimated Spherical 3D Wavelet Transform.

Task: Compute the Isotropic Undecimated Spherical 3D Wavelet Transform of a discrete X sampled on the spherical grid from section 7.2.2.
Parameters: Data samples X and number of wavelet scales J.
Initialization:
- $c^0 = X$.
- Compute the B_3-spline scaling function and derive $\tilde{\psi}_{00n}$, $\tilde{h}_{00n}$ and $\tilde{g}_{00n}$ numerically.
- Compute $\tilde{c}^0_{lmn}$ the discrete spherical Fourier-Bessel transform of c^0.

for $j = 0$ ***to*** $J - 1$ **do**
  (1) Compute the discrete spherical Fourier-Bessel transform of the scaling coefficients: $\tilde{c}^{j+1}_{lmn} = \tilde{c}^j_{lmn}\tilde{h}^j_{00n}$.
  (2) Compute the discrete spherical Fourier-Bessel transform of the wavelet coefficients: $w^{j+1}_{lmn} = c^j_{lmn}g^j_{00n}$.
  (3) Compute the inverse spherical harmonics transform of w^{j+1}_{lmn} to get w^{j+1}.

- Compute the inverse spherical harmonics transform of c^J_{lmn} to get c^J.

Output: $\mathcal{W} = \{w^1, w^2, \ldots, w^J, c^J\}$ the Isotropic Undecimated Spherical 3D Wavelet Transform of X.



(a) Density from the initial spherical Fourier-Bessel coefficients (b) First wavelet scale (c) Second wavelet scale

(d) Third wavelet scale (e) Fourth wavelet scale (f) Smoothed density

Fig. 41. Isotropic Undecimated Spherical 3D wavelet decomposition of a density field. Only a cube at the center of the spherical field is displayed.

7.3.3 Inverse Transform

Since the wavelet coefficients are defined as the difference between two resolutions, the reconstruction from the wavelet decomposition $\mathcal{W} = \{w^1, \ldots, w^J, c^J\}$ is straightforward and corresponds to the reconstruction formula of the *à trous* algorithm:

$$c^0(r, \theta_r, \varphi_r) = c^J(r, \theta_r, \varphi_r) + \sum_{j=1}^{J} w^j(r, \theta_r, \varphi_r) \ . \tag{167}$$

However, given the redundancy of the transform, the reconstruction is not unique. It is possible to take advantage of this redundancy to reconstruct c^j from c^{j+1} and w^{j+1} by using a least squares estimate.

From the recursive wavelet decomposition defined in (164), by respectively multiplying these equations by $\tilde{h}^{*j}_{lm}(k)$ and $\tilde{g}^{*j}_{lm}(k)$ and then added together, the following expression is obtained for the least squares estimate of c^j from c^{j+1} and w^{j+1}:

$$\tilde{c}^j_{lm}(k) = \tilde{c}^{j+1}_{lm}(k)\tilde{H}^j_{lm}(k) + \tilde{w}^{j+1}_{lm}\tilde{G}^j_{lm}(k) \ , \tag{168}$$

where $\tilde{H}^j$ and $\tilde{G}^j$ are defined as follows:

$$\tilde{H}^j_{lm}(k) = \frac{\tilde{h}^{*j}_{lm}(k)}{|\tilde{h}^j_{lm}(k)|^2 + |\tilde{g}^j_{lm}(k)|^2} \tag{169}$$

$$\tilde{G}^j_{lm}(k) = \frac{\tilde{g}^{*j}_{lm}(k)}{|\tilde{h}^j_{lm}(k)|^2 + |\tilde{g}^j_{lm}(k)|^2} \ . \tag{170}$$

Among the advantages of using this reconstruction formula instead of the raw sum over the wavelet coefficients is that there is no need to perform an inverse and then direct spherical Fourier-Bessel transform to reconstruct the coefficients of the original data. Indeed, both the wavelet decomposition and reconstruction procedures only require access to spherical Fourier-Bessel coefficients and there is no need to revert back to the direct space.

7.4 Application: Denoising of a ΛCDM simulation

In this section, we present a simple wavelet denoising application on a density field in spherical coordinates using the Isotropic Undecimated Spherical 3D Wavelet Transform of the previous section.

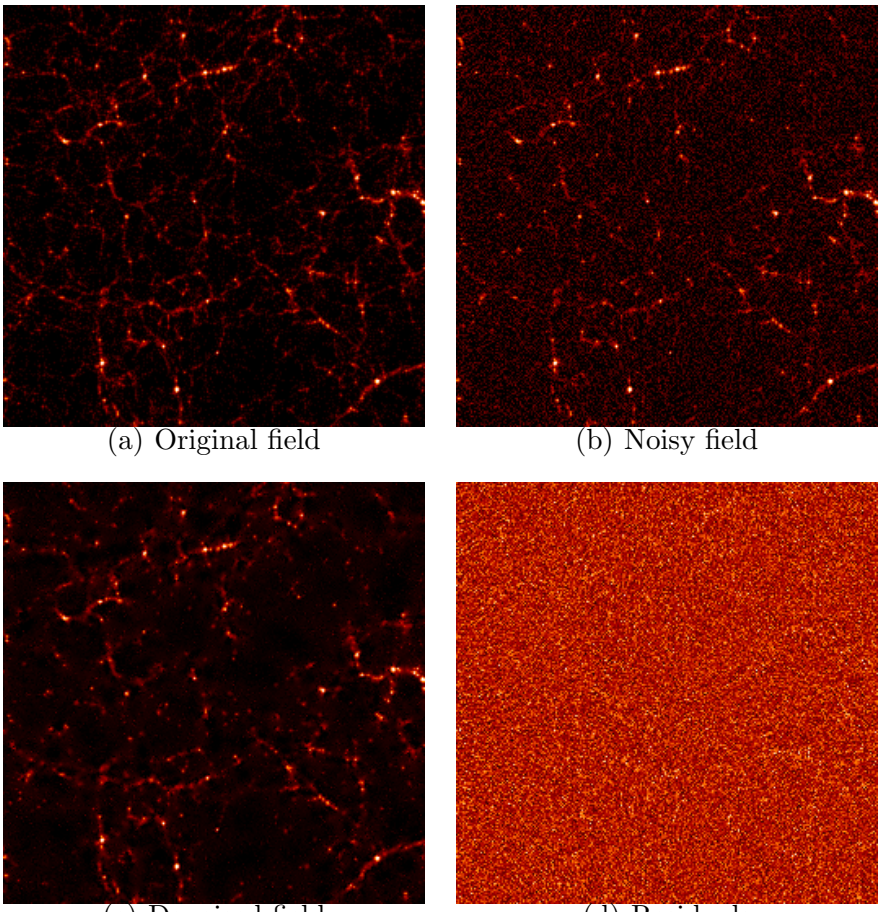
Denoising using sparse transforms can be performed very easily, by applying a simple thresholding on the coefficients. One can use a *soft* or *hard* thresholding according to whether we want more accuracy or less artifacts. The threshold level is usually taken as three times the noise standard deviation, such that for an additive gaussian noise, the thresholding operator kills all noise coefficients except a small percentage, keeping the big coefficients containing information. The threshold we use is often a simple $\kappa\sigma$, with $\kappa \in [3, 4]$, which corresponds respectively to 0.27% and $6.3 \cdot 10^{-5}$ false detections. Sometimes we use a higher κ for the finest scale [3]. Other methods exist, that estimate automatically the threshold to use in each band like the False Discovery Rate (see [69, 70]). The correlation between neighbor coefficients intra-band and/or inter-band may also be taken into account (see [71, 72]).

This experiment is performed on the same N-body simulation from the Virgo Consortium as the one presented in the previous section on Figure 41. The Virgo large box simulation [7] provides us with a Cartesian density cube. The SFB coefficients of the test density field are first computed by sampling the Virgo density field on the spherical 3D grid illustrated in Figure 38, for $n_{side} = 2048$, $l_{max} = 1023$ and $n_{max} = 512$. In order to perform the SFB decomposition, the observer is placed at the center of the box, and the SFB coefficients are

[7] a ΛCDM simulation at $z = 0$, which was calculated using 5123 particles for the following cosmology: $\Omega_m = 0.3$, $\Omega_\Lambda = 0.7$, $H_o = 70\text{kms}^{-1}\text{Mpc}^{-1}$,$\sigma_8 = 0.9$. The data cube provided is 479 h^1Mpc in length.

calculated out to $R = 479/2$ h^{-1}Mpc, setting the density field to zero outside of this spherical volume.

A Gaussian noise was then added to the SFB coefficients to produce a noisy density field. Figures 42(a) and 42(b) show the central portion of slices taken in the middle of respectively the original and noisy spherical density fields. The level of the noise is comparable to the amplitude of the faint filamentary structures that can be seen in the original density field on Figure 42(a). Using Hard Thresholding of the wavelet coefficients, the noisy field is filtered to yield the restored density displayed on Figure 42(c). The residuals after denoising are shown on Figure 42(d). The artificially added noise is successfully removed, without much loss to the large scale structure, though some of the smaller filamentary structures are removed. This however is to be expected given the isotropic nature of the wavelet transform used here, better suited to restore more isotropic features such as clusters.



(a) Original field (b) Noisy field

(c) Denoised field (d) Residuals

Fig. 42. Isotropic Undecimated Spherical 3D wavelet Hard thresholding applied to a test density field

Software

A number of free software are available for different transforms described in this chapter at `http://www.cosmostat.org/software.html`:

- MSVST-lab: Matlab code for Sparse representation-based image deconvolution with Poisson noise.
- Fast 3D-Curvelets: Matlab code for 3D Fast curvelets.
- 3DEX: a code for Fast Fourier-Bessel Decomposition of Spherical 3D Survey.
- MRS3D: 3D Spherical Wavelet Transform on the Sphere.
- MS-VSTS: Multi-Scale Variance Stabilizing Transform on the Sphere.

Other resources include:

- `http://www.flaglets.org`: For the flaglet wavelet transform on the ball.
- `http://www.curvelet.org`: For the Curvelab Matlab/C++ toolbox implementing the Fast Discrete Curvelet Transform.

Acknowledgments

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

References

[1] V. Chandrasekaran, M. Wakin, D. Baron, R. Baraniuk, Representation and compression of multidimensional piecewise functions using surflets, IEEE Transactions on Information Theory 55 (1) (2009) 374–400.

[2] J. Starck, D. Donoho, E. Candès, Very high quality image restoration by combining wavelets and curvelets, in: A. Laine, M. Unser, A. Aldroubi (Eds.), SPIE conference on Signal and Image Processing: Wavelet Applications in Signal and Image Processing IX, San Diego, 1-4 August, SPIE, 2001.

[3] J.-L. Starck, E. Candès, D. Donoho, The curvelet transform for image denoising, IEEE Transactions on Image Processing 11 (6) (2002) 670–684.

[4] G. Hennenfent, F. Herrmann, Seismic denoising with nonuniformly sampled curvelets, IEEE Computing in Science and Engineering 8 (3) (2006) 16–25.

[5] J.-L. Starck, F. Murtagh, E. Candès, D. Donoho, Gray and color image

contrast enhancement by the curvelet transform, IEEE Transactions on Image Processing 12 (6) (2003) 706–717.

[6] M. Elad, J.-L. Starck, P. Querre, D. Donoho, Simultaneous cartoon and texture image inpainting using morphological component analysis, Applied and Computational Harmonic Analysis 19 (2005) 340–358.

[7] M. Fadili, J.-L. Starck, F. Murtagh, Inpainting and zooming using sparse representations, The Computer Journal 52 (1) (2007) 64–79.

[8] J. Starck, M. Nguyen, F. Murtagh, Deconvolution based on the curvelet transform, in: International Conference on Image Processing, 2003, pp. II: 993–996.

[9] J. Starck, M. Nguyen, F. Murtagh, Wavelets and curvelets for image deconvolution: a combined approach, Signal Processing 83 (2003) 2279–2283.

[10] E. Candès, D. Donoho, New tight frames of curvelets and optimal representations of objects with C^2 singularities, Communications on Pure and Applied Mathematics 57 (2) (2003) 219–266.

[11] G. Peyré, S. Mallat, Discrete bandelets with geometric orthogonal filters, Proceedings of ICIP'05 1 (2005) 65–68.

[12] M. Do, M. Vetterli, The contourlet transform: an efficient directional multiresolution image representation, IEEE Transactions on Image Processing 14 (12) (2005) 2091–2106.

[13] D. Rusanovskyy, K. Egiazarian, Video Denoising Algorithm in Sliding 3D DCT Domain, Lecture Notes in Computer Science 37 (08) (2005) 618–625.

[14] I. Selesnick, K. Li, Video denoising using 2D and 3D dual-tree complex wavelet transforms, in: Proc. of SPIE conference on Wavelet Applications in Signal and Image Processing X, San Diego, USA, Aug. 2003.

[15] A. Dima, M. Scholz, K. Obermayer, Semiautomatic quality determination of 3D confocal microscope scans of neuronal cells denoised by 3D wavelet shrinkage, in: H. H. Szu (Ed.), Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, Vol. 3723, 1999, pp. 446–457.

[16] Z. Chen, R. Ning, Breast volume denoising and noise characterization by 3D wavelet transform, Computerized Medical Imaging and Graphics 28 (5) (2004) 235–246.

[17] D. Donoho, O. Levi, Fast X-Ray and Beamlet Transforms for Three-Dimensional Data, in: D. Rockmore, D. Healy (Eds.), Modern Signal Processing, 2002, pp. 79–116.

[18] J. Starck, V. Martinez, D. Donoho, O. Levi, P. Querre, E. Saar, Analysis of the spatial distribution of galaxies by multiscale methods, Eurasip Journal on Applied Signal Processing 15 (2005) 2455–2469.

[19] P. Carre, D. Helbert, E. Andres, 3D fast ridgelet transform, in: International Conference on Image Processing, Vol. 1, 2003, pp. 1021–1024.

[20] A. Woiselle, J. Starck, M. Fadili, 3d curvelet transforms and astronomical data restoration, ACHA 28 (2) (2010) 171–188.

[21] L. Ying, L. Demanet, E. Candès, 3D Discrete Curvelet Transform, in: Proceedings of Wavelets XI conference, San Diego, 2005.

[22] V. Chandrasekaran, M. Wakin, D. Baron, R. Baraniuk, Surflets : A sparse representation for multidimensional functions containing smooth discontinuities, Information Theory, 2004. ISIT 2004. Proceedings. International Symposium on.

[23] D. Donoho, Wedgelets: Nearly minimax estimation of edges, The Annals of Statistics 27 (3) (1999) 859–897.

[24] J. Romberg, M. Wakin, R. Baraniuk, Multiscale wedgelet image analysis: fast decompositions and modeling, in: in IEEE Int. Conf. on Image Proc. 2002, Vol. 3, 2002, pp. 585–588.

[25] Y. Lu, M. N. Do, 3-D directional filter banks and surfacelets, in: Proc. of SPIE Conference on Wavelet Applications in Signal and Image Processing XI, San Diego, USA, 2005.

[26] Y. Lu, M. Do, Multidimensional directional filter banks and surfacelets, IEEE Transactions on Image Processing 16 (4) (2007) 918–931.

[27] D. Labate, W.-Q. Lim, G. Kutyniok, G. Weiss, Sparse multidimensional representation using shearlets, in: Wavelets XI, Vol. 5914, SPIE, 2005, 254–262.

[28] P. Negi, D. Labate, 3-d discrete shearlet transform and video processing, Image Processing, IEEE Transactions on 21 (6) (2012) 2944–2954.

[29] J.-L. Starck, Y. Moudden, P. Abrial, M. Nguyen, Wavelets, ridgelets and curvelets on the sphere, Astronomy and Astrophysics 446, 1191–1204.

[30] D. Marinucci, D. Pietrobon, A. Balbi, P. Baldi, P. Cabella, G. Kerkyacharian, P. Natoli, D. Picard, N. Vittorio, Spherical needlets for cosmic microwave background data analysis, Monthly Notices of the Royal Astronomical Society 383, 539–545.

[31] G. Faÿ, F. Guilloux, Consistency of a needlet spectral estimator on the sphere, ArXiv preprint.

[32] G. Faÿ, F. Guilloux, M. Betoule, J.-F. Cardoso, J. Delabrouille, M. Le Jeune, CMB power spectrum estimation using wavelets, PRD 78 (8), 083013.

[33] Y. Wiaux, J. D. McEwen, P. Vandergheynst, O. Blanc, Exact reconstruction with directional wavelets on the sphere, Monthly Notices of the Royal Astronomical Society 388, 770–788.

[34] Y. Moudden, J.-F. Cardoso, J.-L. Starck, J. Delabrouille, Blind component separation in wavelet space: Application to CMB analysis, EURASIP Journal on Applied Signal Processing 15, 2437–2454.

[35] J. Bobin, Y. Moudden, J. L. Starck, M. Fadili, N. Aghanim, SZ and CMB reconstruction using generalized morphological component analysis, Statistical Methodology 5 (4), 307–317.

[36] J. Delabrouille, J. Cardoso, M. Le Jeune, M. Betoule, G. Fay, F. Guilloux, A full sky, low foreground, high resolution CMB map from WMAP, ArXiv preprint.

[37] P. Abrial, Y. Moudden, J. Starck, J. Bobin, M. Fadili, B. Afeyan,

M. Nguyen, Morphological component analysis and inpainting on the sphere: Application in physics and astrophysics, Journal of Fourier Analysis and Applications 13 (6), 729–748.

[38] P. Abrial, Y. Moudden, J. Starck, M. J. Fadili, J. Delabrouille, M. Nguyen, CMB data analysis and sparsity, Statistical Methodology 5 (4), 289-298.

[39] J. Schmitt, J. L. Starck, J. M. Casandjian, J. Fadili, I. Grenier, Poisson denoising on the sphere: application to the Fermi gamma ray space telescope, AA 517 (2010) A26+.

[40] F. Lanusse, A. Rassat, J.-L. Starck, Spherical 3D isotropic wavelets, AA 540 (2012) A92.

[41] B. Leistedt, J. D. McEwen, Exact Wavelets on the Ball, IEEE Transactions on Signal Processing 60 (2012) 6257–6269.

[42] S. Mallat, A theory for multiresolution signal decomposition: the wavelet representation, IEEE Transactions on Pattern Analysis and Machine Intelligence 11, 674–693.

[43] M. Smith, T. Barnwell, Exact reconstruction technique for tree structured subband coders, IEEE Transactions on Acoustics, Speech, and Signal Processing 34, 434–441.

[44] A. Cohen, I. Daubechies, J. Feauveau, Biorthogonal bases of compactly supported wavelets, Communications in Pure and Applied Mathematics 45, 485–560.

[45] M. Vetterli, Filter banks allowing perfect reconstruction, Signal Processing 10 (3), 219–244.

[46] J.-L. Starck, F. Murtagh, Image restoration with noise suppression using the wavelet transform, Astronomy and Astrophysics 288, 343–348.

[47] J.-L. Starck, F. Murtagh, Astronomical Image and Data Analysis, Springer, 2006, 2nd edn.

[48] M. Holschneider, R. Kronland-Martinet, J. Morlet, P. Tchamitchian, A real-time algorithm for signal analysis with the help of the wavelet transform, in: Wavelets: Time-Frequency Methods and Phase-Space, Springer-Verlag, 1989, 286-297.

[49] M. Shensa, Discrete wavelet transforms: Wedding the à trous and Mallat algorithms, IEEE Transactions on Signal Processing 40, 2464–2482.

[50] J.-L. Starck, F. Murtagh, M. Fadili, Sparse Image and Signal Processing, Cambridge University Press, 2010.

[51] J.-L. Starck, J. M. Fadili, S. Digel, B. Zhang, J. Chiang, Source detection using a 3D sparse representation: application to the Fermi gamma-ray space telescope, AA 504 (2009) 641–652.

[52] B. Zhang, M. J. Fadili, J.-L. Starck, Wavelets, ridgelets and curvelets for Poisson noise removal, IEEE Transactions on Image Processing 17 (7), 1093–1108.

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

[54] D. Donoho, For most large underdetermined systems of linear equations, the minimal ℓ_1 solution is also the sparsest solution, Communications on Pure and Applied Mathematics 59 (7), 907–934.

[55] I. Yamada, 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, S. Reich (Eds.), Inherently Parallel Algorithms in Feasibility and Optimization and Their Applications, Elsevier, 2001.

[56] E. Candès, D. Donoho, Ridgelets: the key to high dimensional intermittency?, Philosophical Transactions of the Royal Society of London A 357, 2495–2509.

[57] D. Donoho, X. Huo, Lecture Notes in Computational Science and Engineering, Springer, 2001, Ch. Beamlets and Multiscale Image Analysis.

[58] P. C. L. Zhi-Oei Liang, Principles of Magnetic Resonance Imaging, IEEE Press, 2000.

[59] E. Candès, Harmonic analysis of neural netwoks, Appl. Comput. Harmon. Anal. 6 (1999) 197–218.

[60] D. Donoho, O. Levi, Fast X-Ray and Beamlet Transforms for Three-Dimensional Data, in: D. Rockmore, D. Healy (Eds.), Modern Signal Processing, 2002, 79–116.

[61] D. Donoho, O. Levi, J.-L. Starck, V. Martínez, Multiscale geometric analysis for 3-D catalogues, in: J.-L. Starck, F. Murtagh (Eds.), SPIE conference on Astronomical Telescopes and Instrumentation: Astronomical Data Analysis II, Waikoloa, Hawaii, 22-28 August, Vol. 4847, SPIE, 2002.

[62] J.-L. Starck, V. Martinez, D. Donoho, O. Levi, P. Querre, E. Saar, Analysis of the spatial distribution of galaxies by multiscale methods, EURASIP Journal on Applied Signal Processing 15, 2455–2469.

[63] P. J. E. Peebles, The large-scale structure of the universe, 1980.

[64] G. Kauffmann, J. M. Colberg, A. Diaferio, S. D. M. White, Clustering of galaxies in a hierarchical universe - I. Methods and results at z=0, MNRAS 303 (1999) 188–206.

[65] E. Candès, D. Donoho, Recovering edges in ill-posed inverse problems: Optimality of curvelet frames, Annals of Statistics 30, 784–842.

[66] D. L. Donoho, X. Huo, Beamlets and multiscale image analysis, Multiscale and Multiresolution Methods, Lecture Notes in Computational Science and Engineering 20 (Springer, NY, USA, 2001) 149–196.

[67] J. Starck, A. Bijaoui, B. Lopez, C. Perrier, Image reconstruction by the wavelet transform applied to aperture synthesis, Astronomy and Astrophysics 283 (1999) 349–360.

[68] E. Candès, D. Donoho, Ridgelets: the key to high dimensional intermittency, Philosophical Transactions of the Royal Society of London A 357 (1999) 2495–2509.

[69] Y. Benjamini, Y. Hochberg, Controlling the false discovery rate: a practical and powerful approach to multiple testing, Journal of the Royal

Statistical Society. Series B (Methodological) 57 (1) (1995) 289–300.
[70] C. J. Miller, C. Genovese, R. C. Nichol, L. Wasserman, A. Connolly, D. Reichart, A. Hopkins, J. Schneider, A. Moore, Controlling the false-discovery rate in astrophysical data analysis, The Astronomical Journal 122 (6) (2001) 3492–3505.
[71] L. Sendur, I. W. Selesnick, Bivariate shrinkage with local variance estimation, IEEE Signal Processing Letters 9 (12) (2002) 438–441.
[72] L. Sendur, I. W. Selesnick, Bivariate shrinkage functions for wavelet-based denoising exploiting interscale dependency, IEEE Trans. on Signal Processing 50 (11) (2002) 2744–2756.
[73] E. Candès, D. Donoho, New tight frames of curvelets and optimal representations of objects with piecewise C^2 singularities, Communications on Pure and Applied Mathematics 57 (2), 219–266.
[74] E. Candès, L. Demanet, D. Donoho, L. Ying, Fast discrete curvelet transforms, Multiscale Modeling and Simulation 5 (3), 861–899.
[75] E. Candès, L. Demanet, D. Donoho, L. Ying, Fast discrete curvelet transforms, SIAM Multiscale Model. Simul. 5 (3) (2006) 861–899.
[76] J. Ma, M. Hussaini, Three-dimensional curvelets for coherent vortex analysis of turbulence, Appl. Phys. Letters 91 (184101).
[77] F. Herrmann, G. Hennenfent, Non-parametric seismic data recovery with curvelet frames, Geophysical Journal International 173 (1) (2008) 233–248.
[78] A. Woiselle, J. Starck, M. Fadili, 3d data denoising and inpainting with the fast curvelet transform, in press (2011).
[79] S. Mallat, A wavelet tour of signal processing, Academic Press, 1998.
[80] A. Woiselle, J. Starck, M. Fadili, 3D Data Denoising and Inpainting with the Fast Curvelet transform, Journal of Mathematical Imaging and Vision (DOI:10.1007/s10851-010-0231-5).
[81] L. Demanet, Curvelets, wave atoms, and wave equations, Ph.D. thesis, California Institute of Technology (May 2006).
[82] N. Kingsbury, Complex wavelets for shift invariant analysis and filtering of signals, Applied and Computational Harmonic Analysis 10 (3) (2001) 234–253.
[83] I. Selesnick, The double-density dual-tree DWT, IEEE Transactions on Image Processing 52 (5) (2004) 1304–1314.
[84] J.-L. Starck, F. Murtagh, M. Fadili, Sparse Signal and Image Processing: Wavelets, Curvelets and Morphological Diversity, Cambridge University Press, Cambridge, UK, 2010.
[85] K. Remi, A. Evans, G. Pike, MRI simulation-based evaluation of image-processing and classification methods, IEEE Transactions on Medical Imaging 18 (11) (1999) 1085.
[86] P. Schröder, W. Sweldens, Spherical wavelets: Efficiently representing functions on the sphere, SIGGRAPH 95, Computer Graphics Proceedings161–172.
[87] J.-P. Antoine, The 2-D wavelet transform, physical applications and gen-

eralizations, in: J. van den Berg (Ed.), Wavelets in Physics, Cambridge University Press, 1999, 23–76.

[88] L. Tenorio, A. H. Jaffe, S. Hanany, C. H. Lineweaver, Applications of wavelets to the analysis of Cosmic Microwave Background maps, Monthly Notices of the Royal Astronomical Society 310, 823–834.

[89] L. Cayón, J. L. Sanz, E. Martínez-González, A. J. Banday, F. Argüeso, J. E. Gallegos, K. M. Górski, G. Hinshaw, 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.

[90] M. Holschneider, Wavelet analysis on the sphere, Journal of Mathematical Physics 37 (8), 4156–4165.

[91] J. Antoine, L. Demanet, L. Jacques, P. Vandergheynst, Wavelets on the sphere: Implementation and approximation, Applied and Computational Harmonic Analysis 13, 177–200.

[92] J. D. McEwen, M. P. Hobson, D. J. Mortlock, A. N. Lasenby, Fast directional continuous spherical wavelet transform algorithms, ArXiv preprint.

[93] W. Freeden, U. Windheuser, Combined spherical harmonics and wavelet expansion – a future concept in Earth's gravitational potential determination, Applied and Computational Harmonic Analysis 4, 1–37.

[94] W. Freeden, F. Schneider, Regularization wavelets and multiresolution, Inverse Problems 14, 225–243.

[95] M. Tegmark, An icosahedron-based method for pixelizing the celestial sphere, Astrophysical Journal Letters 470, l81–L84.

[96] R. A. White, S. W. Stemwedel, The quadrilateralized spherical cube and quad-tree for all sky data, in: D. M. Worrall, C. Biemesderfer, J. Barnes (Eds.), Astronomical Data Analysis Software and Systems I, Vol. 25 of Astronomical Society of the Pacific Conference Series, 1992, 379–381.

[97] R. G. Crittenden, Igloo pixelations of the sky, Astrophysical Letters and Communications 37, 377–382.

[98] K. M. Górski, E. Hivon, A. J. Banday, B. D. Wandelt, F. K. Hansen, M. Reinecke, M. Bartelmann, HEALPix: A framework for high-resolution discretization and fast analysis of data distributed on the sphere, Astrophysical Journal 622, 759–771.

[99] P. Z. Kunszt, A. S. Szalay, A. R. Thakar, The hierarchical triangular mesh, in: A. J. Banday, S. Zaroubi, M. Bartelmann (Eds.), Mining the Sky, 2001, 631–637.

[100] A. G. Doroshkevich, P. D. Naselsky, O. V. Verkhodanov, D. I. Novikov, V. I. Turchaninov, I. D. Novikov, P. R. Christensen, L.-Y. Chiang, Gauss-Legendre sky pixelization (GLESP) scheme for CMB maps, International Journal of Modern Physics D 14 (2), 275–290.

[101] J. Schmitt, J. L. Starck, J. M. Casandjian, J. Fadili, I. Grenier, Multichannel Poisson denoising and deconvolution on the sphere: application to the Fermi Gamma-ray Space Telescope, AA 546 (2012) A114.

[102] B. Zhang, M. Fadili, J.-L. Starck, Fast Poisson noise removal by biorthogonal Haar domain hypothesis testing, Statistical Methodology 5 (4), 387-396.
[103] A. F. Heavens, A. N. Taylor, A spherical harmonic analysis of redshift space, mnras 275 (1995) 483–497.
[104] A. Rassat, A. Refregier, 3D spherical analysis of baryon acoustic oscillations, AA 540 (2012) A115.
[105] J. D. McEwen, Y. Wiaux, A Novel Sampling Theorem on the Sphere, IEEE Transactions on Signal Processing 59 (2011) 5876–5887.
[106] D. Lemoine, The discrete Bessel transform algorithm, The Journal of Chemical Physics 101 (1994) 3936–3944.