A&A 621, A73 (2019)
https://doi.org/10.1051/0004-6361/201834041

Astronomy & Astrophysics

Learning sparse representations on the sphere

F. Sureau[1], F. Voigtlaender[2,3], M. Wust[2], J.-L. Starck[1], and G. Kutyniok[2]

[1] Laboratoire AIM, CEA, CNRS, Université Paris-Saclay, Université Paris Diderot, Sorbonne Paris Cité, 91191 Gif-sur-Yvette, France
e-mail: florent.sureau@cea.fr
[2] Institut für Mathematik, Technische Universität Berlin, 10623 Berlin, Germany
[3] Lehrstuhl für Wissenschaftliches Rechnen, Katholische Universität Eichstätt-Ingolstadt, Ostenstraße 26, 85072 Eichstätt, Germany

Received 8 August 2018 / Accepted 27 October 2018

ABSTRACT

Many representation systems on the sphere have been proposed in the past, such as spherical harmonics, wavelets, or curvelets. Each of these data representations is designed to extract a specific set of features, and choosing the best fixed representation system for a given scientific application is challenging. In this paper, we show that one can directly learn a representation system from given data on the sphere. We propose two new adaptive approaches: the first is a (potentially multiscale) patch-based dictionary learning approach, and the second consists in selecting a representation from among a parametrized family of representations, the α-shearlets. We investigate their relative performance to represent and denoise complex structures on different astrophysical data sets on the sphere.

Key words. methods: data analysis – methods: statistical – methods: numerical

1. Introduction

Wavelets on the sphere (Starck et al. 2015) are now standard tools in astronomy and have been widely used for purposes such as *Fermi* Large Area Telescope data analysis (Schmitt et al. 2010; McDermott et al. 2016), the recovery of the cosmic microwave background (CMB) intensity and polarized CMB maps (Bobin et al. 2015, 2016), string detection (McEwen et al. 2017), point source removal in CMB data (Sureau et al. 2014), the detection of CMB anomalies (Naidoo et al. 2017; Rassat et al. 2014), or stellar turbulent convection studies (Bessolaz & Brun 2011). While wavelets are well suited for representing isotropic components in an image, they are far from optimal for analyzing anisotropic features such as filamentary structures. This has motivated in the past the construction of so-called multiscale geometric decompositions such as ridgelets, curvelets (Candès & Donoho 2004; Starck et al. 2003), bandelets (Le Pennec & Mallat 2005), or shearlets (Labate et al. 2005b). Extensions to the sphere of ridgelets and curvelets were already presented in Starck et al. (2006), Chan et al. (2017) and McEwen (2015), and also for spherical vector field data sets in Starck et al. (2009) and Leistedt et al. (2017).

For a given data set, we therefore have the choice between many fixed representation spaces (such as pixel domain, harmonics, wavelets, ridgelets, curvelets), which are also called dictionaries. A dictionary is a set of functions, named atoms, and the data can be represented as a linear combination of these atoms. The dictionary can be seen as a kind of prior (Beckouche et al. 2013), and the best representation is the one leading to the most compact representation, one in which the maximum of information is contained in few coefficients. For the previously mentioned fixed dictionaries, there exist fast operators for decomposing the data into the dictionary, and fast operators for reconstructing the image from its coefficients in the dictionary (Starck et al. 2015).

In some cases, it is not clear which dictionary is the best, or even if the existing dictionaries are good enough for a given scientific application. Therefore, new strategies were devised in the Euclidean setting to construct adaptive representations. Among them, sparse dictionary learning (DL) techniques (Engan et al. 1999; Aharon et al. 2006) have been proposed to design a dictionary directly from the data, in such a way that the data can be sparsely represented in that dictionary. DL has been used in astronomy for image denoising (Beckouche et al. 2013), stellar spectral classification (Díaz-Hernández et al. 2014), and morphological galaxy classification (Díaz-Hernández et al. 2016).

An alternative approach for adaptively choosing a dictionary is to start with a large parametrized family of dictionaries, and then to choose the parameter(s), either based on simulations or directly from the data. An example of such a parametrized family of dictionaries is the family of α-shearlets (Labate et al. 2005; Grohs et al. 2016; Voigtlaender & Pein 2017).

In this paper, we propose to extend to the sphere both adaptive representation methods, DL and α-shearlets, and we compare the performance of the two approaches. More precisely, we are concerned with adaptive sparsifying representation systems for data defined on the sphere. In Sect. 2, we present our approach for performing DL on the sphere, while Sect. 3 is devoted to our extension of the α-shearlet transform to data defined on the sphere. We present the scenarios for our comparison of the two approaches in Sect. 4; the results of this comparison are presented in Sect. 5. Finally, we conclude the paper in Sect. 6. The necessary background related to α-shearlets in the Euclidean setting is covered in Appendix A.

2. Dictionary learning on the sphere

Dictionary learning techniques were proposed in the early 2000s (Olshausen & Field 1996; Engan et al. 1999; Aharon et al. 2006) to build adapted linear representations that yield sparse decompositions of the signals of interest. Contrary to fixed dictionaries, in dictionary learning the atoms are estimated from the data (or a proxy, such as simulations or exemplars of the data),

and can therefore model more complex geometrical content, which could ultimately result in sparser (and typically redundant) representations. DL techniques have proved their efficiency in many inverse problems in restoration, classification, and texture modeling (see, e.g., Elad & Aharon 2006; Mairal et al. 2008a, 2009; Peyré 2009; Zhang & Li 2010) with improved performance compared to fixed representations (see Beckouche et al. 2013 for denoising astrophysical data). A wide variety of dictionary learning techniques have been proposed to process multivariate data (Mairal et al. 2008a,b); to construct multiscale (Mairal et al. 2008b), translation-invariant (Jost et al. 2006; Aharon & Elad 2008), or hierarchical representations (Jenatton et al. 2011); to estimate coupled dictionaries (Rubinstein & Elad 2014); or to build analysis priors (Rubinstein et al. 2013). Also, online algorithms for dictionary learning have been considered (Mairal et al. 2010).

While fixed structured representations typically have fast direct and inverse transforms, dictionary learning techniques become computationally intractable even for signals of moderate size. Based on the observation that natural images exhibit nonlocal self-similarities, this computational problem is typically overcome by performing dictionary learning on patches extracted from the images that one wants to model. In this section we focus on this patch-based dictionary learning approach, and extend it for signals living on the sphere.

2.1. Sparse representation with patch-based dictionary learning

Given an $n \times n = N$ image represented as a vector $\mathbf{X} \in \mathbb{R}^N$, we consider square overlapping patches $\mathbf{x}_{ij}$ in $\mathbb{R}^Q$, with $Q = q \times q$, where q is typically small; in fact, in the present work we will always have $q \leq 12$. Formally,

$$\mathbf{x}_{ij} = \mathbf{R}_{ij}\mathbf{X}, \tag{1}$$

where the matrix $\mathbf{R}_{ij} \in \mathbb{R}^{Q \times N}$ extracts a patch with its upper left corner at position (i, j).

From a training set $\mathcal{T}$ of such patches $\left\{\mathbf{x}_{ij}\right\}_{(i,j)\in\mathcal{T}}$, a dictionary with M atoms $\mathbf{D} \in \mathbb{R}^{Q \times M}$ is then learned such that the codes $\mathbf{\Lambda} = \left\{\boldsymbol{\lambda}_{ij}\right\}_{(i,j)\in\mathcal{T}}$ satisfying $\mathbf{x}_{ij} = \mathbf{D}\boldsymbol{\lambda}_{ij}$ are sparse. To perform the training, one typically considers the following inverse problem, or one of its variants:

$$\underset{\mathbf{D}\in\mathcal{D},\mathbf{\Lambda}\in C}{\arg\min} \sum_{(i,j)\in\mathcal{T}} \|\mathbf{x}_{ij} - \mathbf{D}\boldsymbol{\lambda}_{ij}\|_2^2 + \mu \cdot \|\boldsymbol{\lambda}_{ij}\|_0, \tag{2}$$

where $\mathcal{D}$ (respectively C) is a non-empty convex set enforcing some constraints on the dictionary $\mathbf{D}$ (respectively the codes $\mathbf{\Lambda}$), and $\mu \cdot \|\boldsymbol{\lambda}_{ij}\|_0$ is the weighted ℓ_0 pseudo-norm, which enforces sparsity of the codes. To remove the scale indeterminacy in such a minimization problem – that is, if $(\mathbf{D}, \mathbf{\Lambda})$ is a solution, then so is $(\alpha\mathbf{D}, \alpha^{-1}\mathbf{\Lambda})$, at least if $\alpha\mathbf{D} \in \mathcal{D}$ and $\alpha^{-1}\mathbf{\Lambda} \in C$ – the set $\mathcal{D}$ typically enforces each atom (column) of the dictionary to belong to a unit ℓ_2 ball, while C can enforce constraints in the code (e.g., non-negativity in non-negative matrix factorization). More details can be found in Starck et al. (2015).

2.2. Extension of patch-based dictionary learning to the sphere

To extend patch-based dictionary learning to data defined on the sphere, we first need to specify how to construct patches on the sphere. We do so by introducing local charts on the sphere. Specifically, in this work we propose to consider the HEALPix framework (Górski et al. 1999, 2005), widely used in astronomy, to construct these charts.

2.2.1. Defining patches on the sphere

HEALPix partitions the sphere into equal area pixels with curvilinear boundaries, defined hierarchically from a set of twelve base quadrilaterals (see Fig. 1). These twelve base elements (or faces) form an atlas of the sphere, and are further partitioned dyadically to obtain finer discretization levels. Consequently, each of the twelve faces is typically considered as a chart with HEALPix pixel positions mapped onto a square grid in $[0, 1] \times [0, 1]$.

Using these charts to perform usual Euclidean patch-based dictionary learning is straightforward, and would have the main advantage of applying dictionary learning directly onto the pixel values, without requiring any interpolation. This comes, however, with two drawbacks: first, this approach introduces boundary issues even when using overlapping patches on each face; second, sampling on the sphere leads to patches with local characteristics (e.g., the pixel shape varies along the latitude in HEALPix). The first of these two problems can be overcome by creating the patches based on local neighbors, as defined by HEALPix. Because of the regularity of the HEALPix sampling scheme, all pixels have eight neighbors, except for eight pixels on the sphere that are located at the vertices in between equatorial and polar faces, which only have seven neighbors. The second problem, however, implies that the same signal defined continuously on the sphere, but centered at different patch centers, will likely lead to different patches being extracted (e.g., for a patch in the equatorial region or in the polar caps). We do not take this effect into account, so that these patches may have a different sparse decomposition or different approximation errors. HEALPix is also not suited to efficiently represent band limited signals, since only approximated quadrature rules are then available to compute spherical harmonic coefficients (Doroshkevich et al. 2005).

Provided some care is taken on defining the respective position of each neighbor to a central pixel across the sphere, overlapping patches can be created – even in between the twelve HEALPix faces – without any interpolation, except at the patches crossing the specific points on the HEALPix grid, which only have seven neighbors. Interpolation strategies to compensate for these "missing" neighbors can be envisioned; but in this work we choose not to interpolate, which implies that for a few pixels around these points, we do not construct all overlapping patches. The final covering of the map is illustrated in Fig. 2, also including patches randomly selected on the sphere. Once these patches are extracted, classical dictionary learning techniques can be used to learn a sparse adapted representation.

2.2.2. Learning a multiscale representation on the sphere

Our proposed approach for dictionary learning on the sphere can be extended to capture multiscale information as proposed in Ophir et al. (2011), namely, by learning a dictionary from patches extracted from a multiscale decomposition of the data. At lower scales, capturing meaningful information would require an increase in the patch size, and would ultimately lead to a computational burden impossible to handle. To capture this information without increasing the patch size, the decomposition is subsampled.

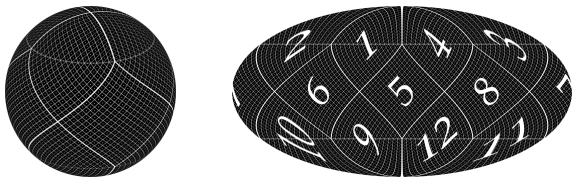

Fig. 1. HEALPix grid (visualizing $N_{\rm side} = 16$) in orthographic projection on the *left* and Mollweide projection on the *right*. Faint lines indicate the circles of latitude $\theta = \cos^{-1}(\pm\frac{2}{3})$. The right image also introduces the numbering of the faces, used in the following illustrations.

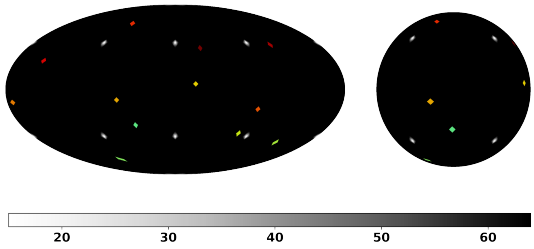

Fig. 2. Example of our covering of the sphere with overlapping patches based on HEALPix neighborhoods for $N_{\rm side} = 128$ and patch width $q = 8$ (note that in our numerical experiments, $N_{\rm side} = 2048$ and the patch width is either $q = 8$ or $q = 12$). Several randomly selected patches on the sphere are also represented in color. The plotted value in gray indicates the number of overlapping patches including each pixel. Because the patch width is usually small with respect to the number of pixels per face, the number of overlapping patches varies in small regions around the pixels that only have seven neighbors.

Table 1. Parameters used for learning the multiscale dictionary for thermal dust data.

Scale	$\ell_{\rm max}$	$N_{\rm side}$	NPatch	q	$M^{(s)}$	$K^{(s)}$	$N_{\rm it}$
3	n.a.	2048	200k	12	256	10	100
2	1024	512	50k	12	256	20	100
1	512	256	25k	12	256	30	100

Notes. For each Starlet scale, the maximal multipole $\ell_{\rm max}$, the $N_{\rm side}$ parameter, the number of patches, their width q, the number of atoms $M^{(s)}$, the maximal sparsity $K^{(s)}$, and the number of iterations $N_{\rm it}$ are displayed.

In this work, we use the Starlet decomposition for data on the sphere (Starck et al. 2006), with one dictionary learned per wavelet scale. Since all scales except the last one are band-limited, subsampling can be performed without loosing information by adapting the $N_{\rm side}$ parameter to the maximal multipole at the level considered (typically dyadically decreasing, as illustrated in Table 1).

The resulting minimization problem for the multiscale dictionary learning problem reads

$$\underset{\substack{\{\mathbf{D}^{(s)}\}_{s=1..S}\in\mathcal{D},\\ \{\mathbf{\Lambda}^{(s)}\}_{s=1..S}\in\mathcal{C}}}{\arg\min} \sum_{s=1}^{S}\sum_{(i,j)\in\mathcal{T}^{(s)}} \|\mathbf{R}_{ij}\mathcal{W}^{(s)}\mathbf{X}-\mathbf{D}^{(s)}\boldsymbol{\lambda}_{ij}^{(s)}\|_2^2+\mu^{(s)}\cdot\|\boldsymbol{\lambda}_{ij}^{(s)}\|_0, \quad (3)$$

where $\mathbf{X}$ is the signal on the sphere, $\mathcal{W}^{(s)}$ extracts the scale s of the wavelet transform on the sphere according to the $N_{\rm side}$ chosen for that scale, $\mathbf{R}_{ij}$ is now extracting patches according to neighbors on the sphere for the patch indexed by (i, j) at scale s in training set $\mathcal{T}^{(s)}$, and S is the total number of wavelet scales. For each scale $s = 1,\dots,S$, a dictionary $\mathbf{D}^{(s)}$ is therefore learned, giving coefficients $\boldsymbol{\lambda}_{ij}^{(s)}$ collected in $\mathbf{\Lambda}^{(s)}$; the hyperparameter $\mu^{(s)}$ is also allowed to change with the scale. Because the cost function is separable per scale, the minimization problem Eq. (3) is equivalent to solving S dictionary learning sub-problems associated to each wavelet scale.

2.3. Our algorithm for patch-based dictionary learning on the sphere

In the training phase, the joint nonconvex problems described in Eqs. (2)–(3) are typically handled by alternating sparse coding steps and dictionary update steps. Here, a sparse coding step means that one minimizes Eq. (2) (resp. Eq. (3)) with respect to $\mathbf{\Lambda}$ (resp. $\mathbf{\Lambda}^{(s)}$), with a fixed previously estimated dictionary. Similarly, a dictionary update step means that one minimizes Eq. (2) (resp. Eq. (3)) with respect to $\mathbf{D}$ (resp. $\mathbf{D}^{(s)}$), with the fixed, previously estimated codes. Standard algorithms were proposed for both sub-problems. In this work, we will use the classical dictionary learning technique K-SVD (Aharon et al. 2006) with Orthogonal Matching Pursuit (OMP; Mallat & Zhang 1993; Pati & Krishnaprasad 1993) as a sparse coder. For denoising applications, the sparse coding step will encompass both a maximal sparsity level, and an approximation threshold based on the ℓ_2 norm of the residual, similar to the approach in Elad & Aharon (2006). This approach resulted in adapted sparse representations, while not being sensitive to small fluctuations below the targeted level of approximation, and in practice led to faster algorithms. The resulting multiscale dictionary learning algorithm is described in Algorithm 1, from which its variant without the multiscale transform can be obtained for $S = 1$ and $\mathcal{W}^{(1)} = Id$.

Algorithm 1 Multiscale Dictionary Learning on the Sphere

Initialization: For each scale $s = 1,\dots,S$, choose the number of atoms $M^{(s)}$, a maximal sparsity degree $K^{(s)}$, a maximal approximation error $\epsilon^{(s)}$. Initialize the dictionary. Choose the number of iterations N_{it}.
Patch Extraction: For each scale s, extract randomly patches $\left\{\mathbf{R}_{ij}\mathcal{W}^{(s)}\mathbf{X}\right\}_{(i,j)\in\mathcal{T}^{(s)}}$ on the sphere. Subtract from each patch its mean value.
for $s = 1$ to S **do** {**Subproblem for scale** s}
  for $n = 0$ to N_{it} **do** {**Main Learning Loop**}
    for $(i, j) \in \mathcal{T}^{(s)}$ **do** {**Sparse Coding**}
      Compute the sparse code $\boldsymbol{\lambda}_{ij}^{(s)}$ using OMP with stopping criterion $\|\mathbf{R}_{ij}\mathcal{W}^{(s)}\mathbf{X} - \mathbf{D}^{(s)}\boldsymbol{\lambda}_{ij}^{(s)}\|_2 < \epsilon^{(s)}$ or $\|\boldsymbol{\lambda}_{ij}^{(s)}\|_0 > K^{(s)}$
    end for
    Update $\mathbf{D}^{(s)}$ using K-SVD (Aharon et al. 2006) {**Dictionary Update**}
  end for
end for
return $\left\{\mathbf{D}^{(s)}\right\}_{s=1..S}$

The first critical choice for this dictionary learning technique is to adapt the patch size q to capture information at the scale of the patch without impacting too much the computational burden of the algorithm (q is at most 12 in this work). The maximal sparsity degree $K^{(s)}$ and the number of atoms $M^{(s)}$ should be selected so that the dictionary leads to small approximation errors, while being able to capture the important features with only a few atoms, in particular for denoising applications. The parameter $\epsilon^{(s)}$ is the noise level expected in the denoising application at the

considered wavelet scale, and the number of iterations is in practice chosen to be sufficiently large so that the average approximation error does not change with iterations. Because this problem is non-convex, it is crucial to initialize the algorithm with a meaningful dictionary; in our case, the initial dictionary is chosen to be an overcomplete discrete cosine transform (DCT) dictionary as in Elad & Aharon (2006).

3. Extension of α-shearlets to the sphere

3.1. Euclidean α-shearlets

Adaptive dictionaries can also be derived from a parametrized family of representations such as the α-shearlets that generalizes wavelets and shearlets and are indexed by the anisotropy parameter $\alpha \in [0, 1]$. To each parameter α corresponds a dictionary characterized by:

- atoms with a "shape" governed by height $\approx$ width$^{\alpha}$ (see Fig. A.2);
- a directional selectivity: on scale j, an α-shearlet system can distinguish about $2^{(1-\alpha)j}$ different directions (see Fig. A.3);
- a specific frequency support for the atoms (see Fig. A.3).

A key result (Voigtlaender & Pein 2017) is that α-shearlets are almost optimal for the approximation of so-called C^{β}-cartoon-like functions, a model class for natural images. More precisely, the N-term α-shearlet approximation error (that is, the smallest approximation error that can be obtained using a linear combination of N α-shearlets) for a C^{β}-cartoon-like function is decreasing at (almost) the best rate that any dictionary can reach for the class of such functions. For this to hold, the anisotropy parameter α needs to be adapted to the regularity β, that is, one needs to choose $\alpha = 1/\beta$. For more details on this, we refer to Appendix A.

In general, given a certain data set, or a certain data model, different types of α-shearlet systems will be better adapted to the given data than other α'-shearlet systems. Thus, having such a versatile, parametrized family of representation systems is valuable to adapt to a variety of signals to recover.

3.2. Defining α-shearlet transforms on the sphere

In order to define the α-shearlet transform on the sphere, similarly to what was discussed for the dictionary learning approach, we need to define the charts on which the Euclidean α-shearlet transform will be applied. HEALPix faces are again an obvious candidate since these base resolution pixels can be interpreted as squares composed of N_{side} by N_{side} equally spaced pixels, although their shape is contorted in different ways on the sphere (see Fig. 1).

We could map the sphere to these twelve square faces and then take the α-shearlet transform on every one of them individually. However, as for dictionary learning, this approach to the processing of HEALPix data (e.g., for the task of denoising) is deemed to introduce boundary artifacts for this partition of the sphere. An example of such artifacts can be seen in the upper-left part of Fig. 18 shown in Sect. 5. Besides, contrary to patch-based dictionary learning where the patch size remains typically small compared to a face size, the increasing size of the α-shearlet atoms when going to lower scales can introduce large border effects.

In the following two subsections, we discuss two approaches for handling this problem. Similarly to dictionary learning in Sect. 2.2.1, we do not take into account the variation of the pixel shapes along the sphere when extending α-shearlets to the sphere.

3.2.1. The rotation-based approach

The first strategy to alleviate the block artifacts was proposed for curvelets in Starck et al. (2006). This approach relies on considering overlapping charts that are obtained by considering HEALPix faces after resampling the sphere through a small number of rotations. More precisely, for a given Euclidean α-shearlet system, a HEALPix face f, and a rotation $\mathbf{r}$, the redundant coefficients are obtained by

$$\boldsymbol{\lambda}_{\alpha,\mathbf{r},f} = \mathcal{S}_{\alpha}\left(\mathbf{H}_f \mathcal{R}_{\mathbf{r}}\left(\mathbf{X}\right)\right), \tag{4}$$

where $\mathcal{R}_{\mathbf{r}}$ computes the resampled map by a rotation $\mathbf{r}$ of the sphere, $\mathbf{H}_f$ is a matrix extracting the pixels that belong to the HEALPix face f, and $\mathcal{S}_{\alpha}$ computes the Euclidean α-shearlet transform on this face. In practice, a bilinear interpolation is performed by the HEALPix rotation routines that are used for the resampling.

The reconstruction is performed using a partition of unity on the sphere (see Fig. 3), which is obtained from weights that are smoothly decaying from 1 in a central region of the faces to 0 at their borders and therefore mitigating border effects. Formally, the reconstruction reads

$$\widetilde{\mathbf{X}} = \mathbf{N} \sum_{\mathbf{r}} \sum_{f=1}^{12} \mathcal{R}_{-\mathbf{r}}(\mathbf{H}_f^T \mathbf{M} \mathcal{T}_{\alpha}(\boldsymbol{\lambda}_{\alpha,\mathbf{r},f})), \tag{5}$$

where $\mathcal{R}_{-\mathbf{r}}$ resamples the sphere with the inverse rotation matrix, $\mathcal{T}_{\alpha}$ computes the inverse α-shearlet transform, $\mathbf{M}$ applies weights, and the normalization matrix $\mathbf{N}$ is simply a pointwise multiplication with weights chosen such that $\mathbf{N} \sum_{\mathbf{r},f} \mathcal{R}_{-\mathbf{r}}\left(\mathbf{H}_f^T \mathbf{M} \mathbf{1}\right) = \mathbf{1}$ where $\mathbf{1}$ is a vector with all entries equal to 1. An example of the weights and normalization maps used to construct this partition of unity is illustrated in Fig. 3.

Since the rotations $\mathcal{R}_{\mathbf{r}}$ and $\mathcal{R}_{-\mathbf{r}}$ are implemented using interpolation, it is not true exactly that $\mathcal{R}_{-\mathbf{r}}\mathcal{R}_{\mathbf{r}}\mathbf{X} = \mathbf{X}$. Therefore, even if the coefficients $\boldsymbol{\lambda}_{\alpha,\mathbf{r},f}$ are obtained through Eq. (4), the reconstruction in Eq. (5) will only satisfy $\widetilde{\mathbf{X}} \approx \mathbf{X}$, not $\widetilde{\mathbf{X}} = \mathbf{X}$. However, the error introduced by the inexact inverse rotation is often negligible, at least for sufficiently smooth signals; Sect. 5.2.4 offers further comment on this.

3.2.2. The "patchwork" approach

The "patchwork" approach is another strategy to eliminate artifacts that would arise if one naively used the disjoint HEALPix faces. Contrary to the rotation-based technique, where an interpolation is performed during the resampling, the patchwork approach is based on extending the HEALPix faces using parts of the surrounding faces so as to avoid interpolation. Similar to the rotation-based approach, the six resulting extended faces (see Fig. 4) form a redundant covering of the sphere, which is beneficial for avoiding boundary artifacts. Once these six extended faces are computed, the α-shearlet transform and all further processing are performed on these faces. Of course, for the reconstruction, the last step consists in combining the redundant faces to get back a proper HEALPix map.

Formally, the decomposition can be described as

$$\boldsymbol{\lambda}_{\alpha,f} = \mathcal{S}_{\alpha}\left(\mathcal{P}_f\left(\mathbf{X}\right)\right), \tag{6}$$

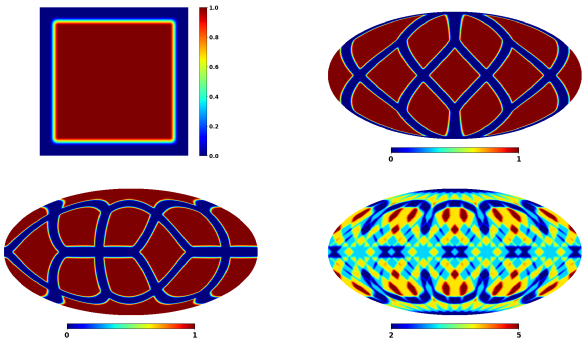

Fig. 3. Partition of unity for the rotation-based reconstruction. The weights smoothly decaying toward the border are presented in the *top left panel* and are copied to each HEALPix face in the *top right panel*. In the *bottom left panel*, resampling was first performed using a rotation and bilinear interpolation, and the image shows the weights that would be applied in the original reference coordinates. The resulting covering of the sphere using five rotations is illustrated in the *bottom right panel*.

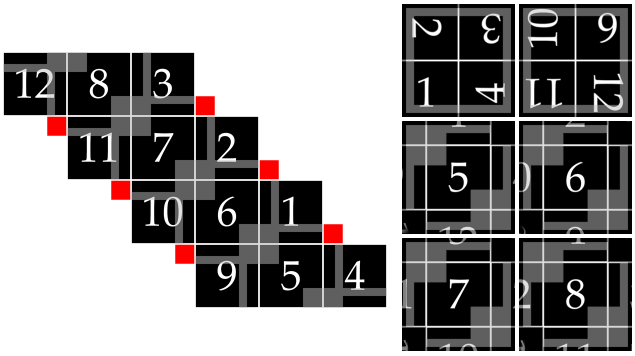

Fig. 4. *Left panel*: twelve squares corresponding to the faces of the HEALPix framework (see Fig. 1) arranged as a net in the plane. The areas that are covered by multiple of the extended faces – the transition zones – are displayed in gray. The areas where pixels are "missing" are displayed in red. *Right panel*: six extended faces produced by the patchwork procedure. The two polar faces form the top row, followed by the four equatorial faces below. The shaded area around the transition zone of each composite face indicates the margin, which is later discarded.

where $\mathcal{P}_f$ is now the operator that extracts the extended face f from the HEALPix map $\mathbf{X}$. Similarly, the reconstruction reads

$$\widetilde{\mathbf{X}} = \mathcal{M}\left[\left(\mathcal{T}_\alpha\left(\boldsymbol{\lambda}_{\alpha,f}\right)\right)_{f=1,\dots,6}\right], \tag{7}$$

where $\mathcal{M}$ is the operator that reconstructs a HEALPix map from data on the six extended faces.

The rest of this section explains how precisely the extended faces are obtained from the original HEALPix faces, and conversely how a HEALPix map can be obtained from data on these six extended faces. For an accompanying visual explanation of the procedure, the reader should consult Figs. 1, 4, and 5.

Each of the six extended faces consists of an inner square with HEALPix pixels that are unique to this extended face, and a border zone with HEALPix pixels that appear in several of the extended faces. The border itself is again subdivided into an outer margin that is disregarded after the reconstruction step so that the artifacts at the boundary are cut off (not mapped to the sphere), and an inner part that forms a transition zone, where the values of neighboring faces are blended together to prevent visible discontinuities between them.

Instead of extending all twelve original faces, we combine them to six bigger composite faces and extend those. This reduces the number of additional pixels that have to be processed (when using a border of the same size), at the cost of increased memory requirements. The first two composite faces cover the bulk of the north and south polar regions, and particularly the poles itself. Since the four faces of each polar region meet at the poles, we can arrange those four faces to form a square around the pole. It only remains to clip this area to the requested size. Although there is much freedom to set the extent of the individual composite faces, we prefer all squares to be of equal size, so that they can be processed without distinction. The remaining four composite faces are obtained by expanding the equatorial faces. An expansion of the equatorial faces by $\frac{N_\mathrm{side}}{4}$ in each direction results in areas of width $\frac{3N_\mathrm{side}}{2}$, which each contain a fourth of every surrounding polar face. By removing those parts from the polar areas, constructed earlier, those are truncated to the same width (see Fig. 5). Thus, we get six areas of equal size that cover the sphere. Chosen this way, there is still no overlap between the polar and equatorial composite faces; therefore

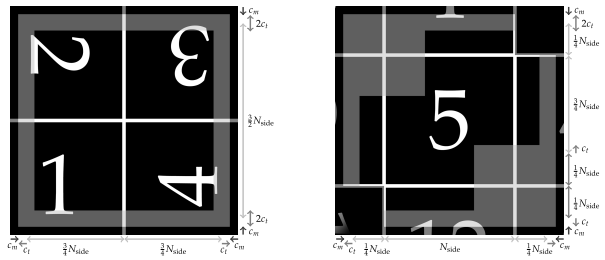

Fig. 5. Detailed view of two of the six extended faces. The dark outer boundary with width c_m is the margin that is discarded after the reconstruction step, and the two dark squares in the corners of the equatorial face on the right are treated likewise. The remaining part of the extended faces has a gray outer boundary of width $2c_t$. In conjunction with the gray squares in the corners of the equatorial face, this boundary forms the transition zone that contains the values shared with the neighboring extended faces.

we extend each face further by half the requested width of the transition zone. We chose an extension of width $\frac{N_\mathrm{side}}{16}$ (that is c_t in Fig. 5). Since each face enters its neighbors territory by that amount, this results in a transition zone of width $\frac{N_\mathrm{side}}{8}$ between each face. Additionally each face is extended by a margin (that is c_m in Fig. 5) to avoid border artifacts. Here, a margin of width $\frac{N_\mathrm{side}}{16}$ was chosen.

However, to extend the equatorial faces, we have to address the problem that there are eight vertexes where two faces of a polar region meet a face of the equatorial region (located on the circles of latitude $\theta = \cos^{-1}(\pm 2/3)$, depicted in Fig. 1). By arranging the twelve faces as a net in the plane – as illustrated in Fig. 4 – it becomes clear that there are gaps between the polar faces, where no values exist; these areas are marked in red in Fig. 4. We need to fill those gaps in order to obtain rectangular extended faces, to which we can apply the α-shearlet transform. In the end, these parts will be cut away and disregarded like the outer margin of the extension, so the filled-in values will not actually be used for the reconstruction. Nevertheless, we need to be careful, since otherwise we might introduce additional artifacts like the ones at the boundary.

For the sake of simplicity, we will describe the situation at the edge between faces 1 and 2 (see Figs. 1, 4, and 6), which

is exemplary for all gaps. From the perspective of face 2, the missing square is expected to feature a rotated copy of face 1, while conversely face 1 expects a rotated copy of face 2. To fabricate a weighted blending of those anticipated values, we divide the empty square, interpreted as $[0,1]^2$, along the lines $2x = y$, $x = y$, and $x = 2y$, into quarters, as demonstrated in Fig. 6. On both outer quarters the full weight is assigned to the face which the adjoining face expects, while the two middle quarters serve to produce a smooth transition. All weights are normalized in such a way that every pixel is a convex combination of the pixels of the two faces; that is, the weights are non-negative and their sum is one at each pixel.

With this process, we fill the vertex regions with values. We do not actually need to fill the whole square, but only the corner needed for the expansion (the red part in Fig. 4). Having done this, we can piece the equatorial faces together from the various parts of the six surrounding faces and two filler squares. Figure 4 shows the resulting extended faces on the right.

We have now described the operators $\mathcal{P}_f$ appearing in Eq. (6), which assign to a given HEALPix map $\mathbf{X}$ the six extended faces $\mathcal{P}_1(\mathbf{X}), \ldots, \mathcal{P}_6(\mathbf{X})$. On these rectangular faces, we can then apply the usual α-shearlet transform, and do any further processing that is desired (for instance, we can denoise the six extended faces by thresholding the α-shearlet coefficients).

After the processing is done on the six extended faces, the outer margin and filler values are disregarded and the remnant is separated along the boundaries of the original faces. From these pieces, the original faces are put back together. While doing so, all pixels that were part of a transition zone are weighted, similarly to above, as a convex combination of the pixels of the (up to four) involved extended faces.

Since we use only the values provided by the HEALPix grid, and instead of interpolating between pixels use convex combinations of pixel values in the transition zones, the patchwork procedure is invertible, with Eq. (7) describing a left inverse to the "patchwork α-shearlet coefficient operator" described in Eq. (6). Thus, the patchwork-based α-shearlets form a frame. We emphasize, however, that the reconstruction procedure described in Eq. (7) is not necessarily identical to the one induced by the canonical dual frame of the patchwork-based α-shearlet frame.

4. Experiments

To evaluate α-shearlets and dictionary learning, we have selected two different simulated data sets on the sphere:

- Thermal dust map: a full sky thermal dust map from the *Planck* Sky Model (100 GHz map) (Planck Collaboration XII 2016), obtained through the *Planck* Legacy Archive[1].
- Horizon full sky maps: a series of full sky maps from the Horizon N-body simulations describing the dark matter halo distribution between redshift 0 and 1 (Teyssier et al. 2009)[2].

While in the former scenario, the signal is smooth and expected to be best represented by multiscale transforms, in the latter the signal is more discontinuous and geometrically composed of filamentary structures joining clusters, with density changing with redshift. These two simulations are therefore illustrative of different scenarios where such adaptive transforms would be useful.

[1] http://pla.esac.esa.int/pla/#maps
[2] See http://www.projet-horizon.fr

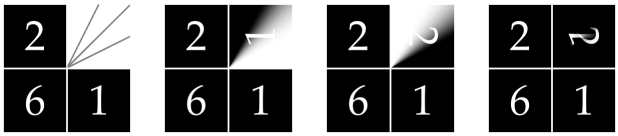

Fig. 6. "Missing" square between faces 1 and 2 is divided into four triangles of equal size, separated by the lines $2x = y$, $x = y$, and $x = 2y$, as seen on the *left*. The two images in the middle reveal how the rotated faces 1 and 2 are separately weighted along those segments. The data of face 1 have full weight (black) on the outer triangle adjacent to face 2, and no weight (white) on the other outer triangle, while the data of face 2 are treated conversely. A smooth transition is provided by the weights on the triangles in between. The sum of the weighted faces is used to fill the gap, as demonstrated in the right-most illustration.

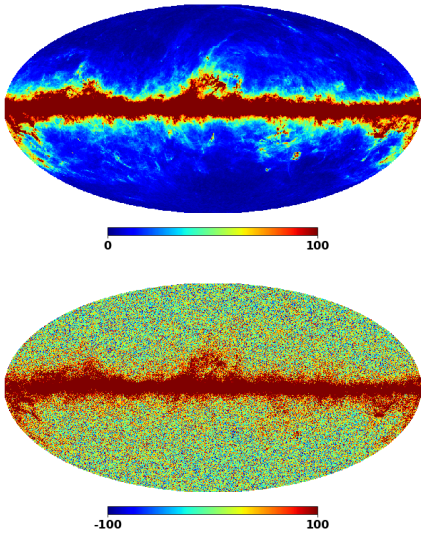

Fig. 7. Thermal dust simulation map (at 100 GHZ) without (*top panel*) and with the additive white Gaussian noise added (*bottom panel*), for evaluation of the methods. The colorscale has been stretched to illustrate the challenge of recovering structures at intermediate latitude. Units are in μK.

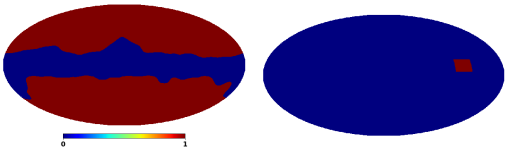

Fig. 8. *Left panel*: galactic mask used for thermal dust quantitative evaluation, covering 70% of the sky. *Right panel*: region close to galactic plane where methods are inspected.

To evaluate the respective performance of DL and α-shearlets for denoising, we have added to the thermal dust map an additive white Gaussian noise with standard deviation 45 μK, which corresponds to the expected level of CMB at such frequency. The resulting map can be seen in Fig. 7.

The galactic mask used for quantitative comparisons to separate regions of high dust amplitude from regions with lower values at higher galactic latitude is displayed in Fig. 8, along with the location of a region close to the galactic plane where the differences between the methods could be better visualized.

For the dark matter halo distribution, we select the first slice of the data cube, and adjust the white noise level to 5, so that filamentary structures are of a similar amplitude to the

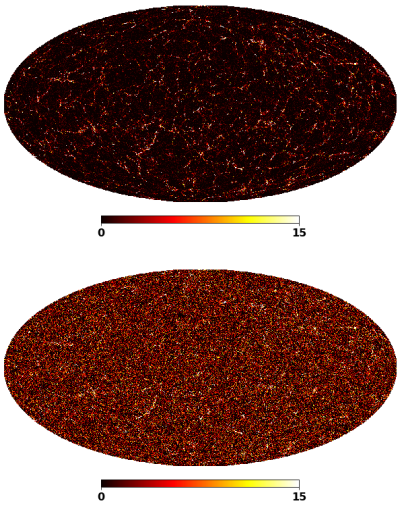

Fig. 9. Dark matter halo distribution for the first slice, without (*top panel*) and with the additive white Gaussian noise added (*bottom panel*), for evaluation of the methods. The colorscale has been stretched to visualize filamentary structures.

noise, as can be observed in Fig. 9. This noise does not correspond to something realistic in our actual experiments, but our goal here is only to evaluate how different adaptive representations behave when extracting features embedded in Gaussian noise.

In the following two subsections, we outline the precise choice of the hyperparameters that we used for the α-shearlets and for the dictionary learning based denoising, respectively.

4.1. Parameters for α-shearlets

For the two α-shearlet approaches, we used 11 values of α, sampled uniformly with a density of 0.1 ranging from 0 to 1. We used four scales of decomposition, using either the rotation-based approach (Eq. (4)), or the patchwork approach (Eq. (6)). For the actual denoising, we performed a hard thresholding of the α-shearlet coefficients. For this, we used different detection thresholds on different scales. To be precise, we used a 4σ detection threshold for the last scale with a lower signal to noise ratio, and a detection threshold of 3σ for the other scales; for the coarse scale, however, we did not do any thresholding. The reconstruction was then performed using either Eq. (5) or (7).

For the rotation-based approach, five rotations were selected as a balance between having "more uniform" weights and the computational burden of this approach. The weight maps were built using a margin and transition (smooth trigonometric variation in between 0 and 1) of size $\frac{N_{\text{side}}}{16}$.

For the patchwork approach, we set the size of both the utilized extension and the margin to $\frac{N_{\text{side}}}{16}$, which results in increasing the number of pixels that have to be processed by about half (53.1%). A little less than half of the added pixels are used for the sake of redundancy, and the rest is disregarded.

4.2. Dictionary learning parameters

For the thermal dust data where the information is present at several scales, we chose the multiscale dictionary learning technique. Three wavelet scales of the Starlet transform on the sphere (Starck et al. 2006) were first computed from the input simulated dust map without noise. To avoid artifacts for a non band-limited signal, the finest wavelet scale has not been directly computed through its spherical harmonic decomposition. We followed Algorithm 1 for the learning procedure, with the parameters listed in Table 1. The patch size, the number of atoms, and the maximal sparsity were selected experimentally by choosing values that lead to the lowest average approximation error during the training phase.

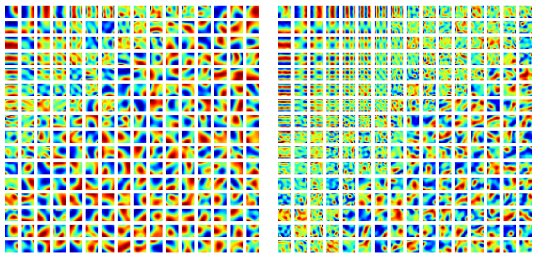

Fig. 10. Atoms learned in the multiscale dictionary learning approach: on the *left*, scale 3, on the *right*, scale 2. The dictionaries have departed from the original redundant DCT dictionary and have learned specific features related to the scale. Due to the change of the N_{side} parameter with the scale, the actual distance between two adjacent pixels has increased, and the atoms for scale 2 are indeed smoother than those for scale 3.

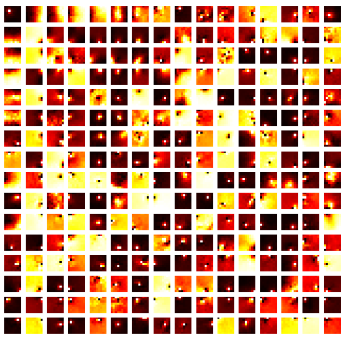

Fig. 11. Atoms learned in the dictionary learning approach, applied to the dark matter halo distribution data. The dictionary elements are composed of point-like structures and edges.

An example of a dictionary learned for this adaptive multiscale representation of thermal dust is shown in Fig. 10. The dictionaries have captured at various scales both directional and more isotropic structures.

In the second scenario, because information is localized in space, the dictionary was learned directly on patches extracted from the first slice describing the dark matter halo distribution, from a training set of 200 000 patches of size 8×8. As in the previous experiment, a stopping criterion was set for the approximation error (which should be less than the targeted level of noise), and a maximal sparsity of 7 was set for OMP. K-SVD was then run for 100 iterations. The learned dictionary is presented in Fig. 11. The atoms essentially contain high frequency information in this case, in contrast to the previously learned distribution on thermal dust.

Once these dictionary are learned, the sparse decomposition step with this representation is used for denoising. The same parameters as above were used for the sparse coding, except for the targeted approximation error, which was set to a value that would not be exceeded by a patch of pure noise with a probability of 0.9545.

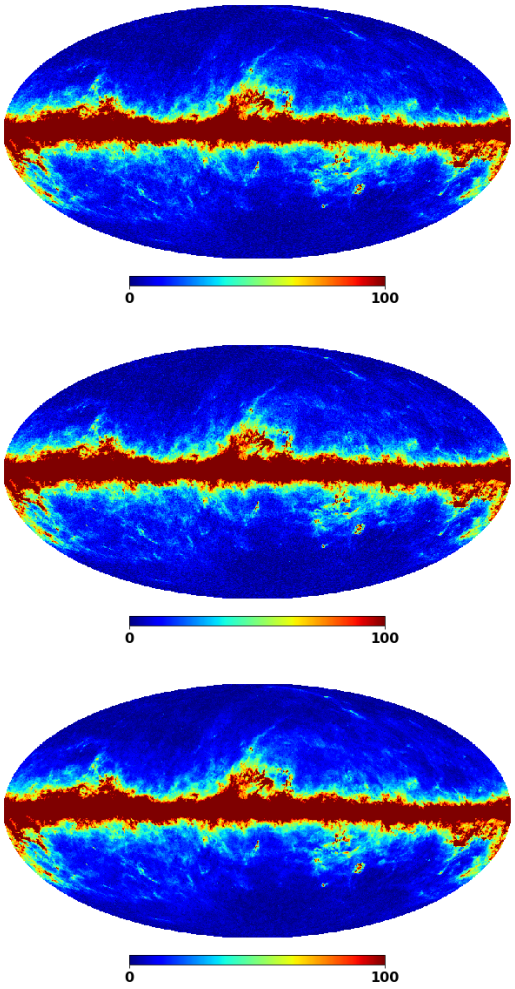

Fig. 12. Denoised thermal dust maps for all three approaches. *Top* and *middle panels*: α-shearlet denoising with rotation-based (*top*) or patchwork (*middle*) approach, both for $\alpha = 0.6$. *Bottom panel*: representation learned with dictionary learning. Units are in μK.

5. Results

5.1. Denoising experiments

We tested our adaptive approaches to denoise the data in the two denoising scenarios presented in the previous section, using the parameters described in Sects. 4.1 and 4.2. For the thermal dust simulation, the full sky denoised maps using the three approaches are displayed in Fig. 12, with a zoom to a region close to the galactic plane in Fig. 13, to visually inspect the differences between methods. Residuals on the full sphere are also shown in Fig. 14, and the performance of the different approaches is quantitatively evaluated in Table 2 in the full sky as well as in regions defined by the galactic mask.

Similarly, for the dark matter halo distribution, the full sky denoised maps are displayed in Fig. 15 and the residuals are presented in Fig. 16. To better inspect the recovery of the filamentary structures as well as the core regions, a zoom-in was also performed for this dataset; this is shown in Fig. 17. Finally, the results are quantitatively evaluated in Table 3.

To inspect the impact of the anisotropy parameter on the recovery of geometrical structures in the different redshift slices, we also computed for the patchwork approach the non-linear approximation curves that display the evolution of the RMSE as a function of given thresholds. This allows for a more comprehensive view of the best α for different density level thresholds. These non-linear approximation curves are illustrated in linear and log scale in Figs. 19 and 20, respectively.

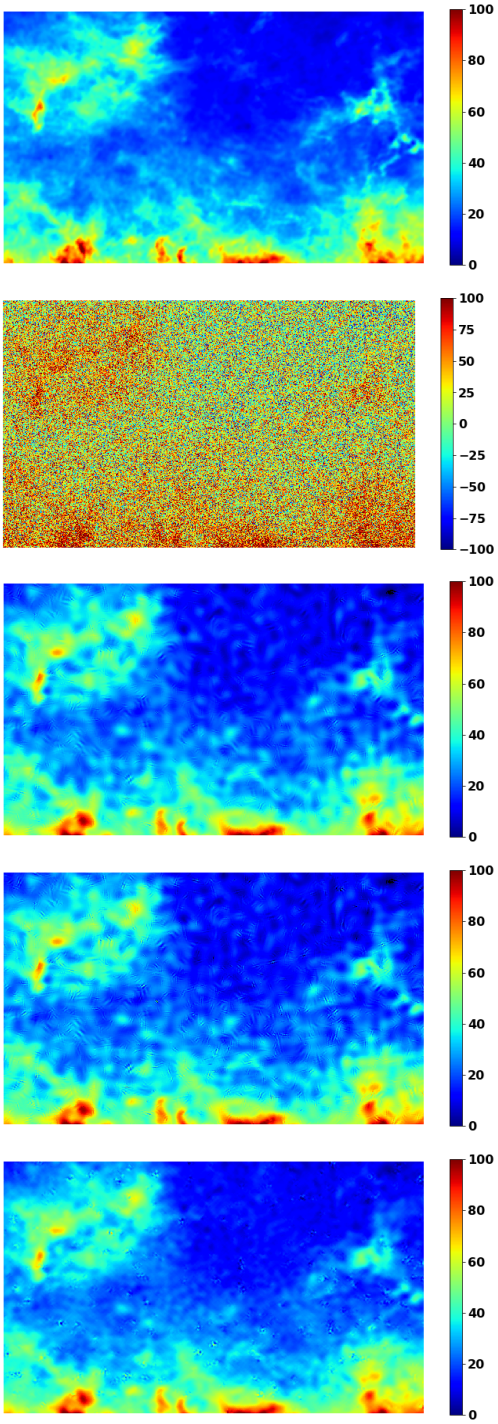

Fig. 13. Zoom on a region close to the galactic plane to visualize the respective denoising performance of the methods. *From top to bottom panels*: input map, noisy map (with own colorscale), rotation-based approach with $\alpha = 0.6$, patchwork approach with $\alpha = 0.6$, sparse representation learned from data. All units are in μK.

5.2. Discussion

In the following, we discuss several questions concerning the results; in particular, we analyze the relative performance of our different approaches to sparsifying representations on the sphere.

Table 2. Statistics on the recovery of spherical thermal dust maps with the proposed approaches.

Method		Bias			RMSE			MAD		
		All	Out	Gal.	All	Out	Gal.	All	Out	Gal.
Rotation	$\alpha = 0$	0.008	0.005	0.016	4.266	3.028	6.270	3.020	2.392	4.490
	$\alpha = 0.1$	0.008	0.005	0.016	4.264	3.025	6.268	3.018	2.389	4.488
	$\alpha = 0.2$	0.008	0.005	0.016	4.261	3.022	6.264	3.016	2.387	4.485
	$\alpha = 0.3$	0.008	0.005	0.016	4.256	3.019	6.257	3.012	2.384	4.480
	$\alpha = 0.4$	0.008	0.005	0.016	4.256	3.021	6.255	3.012	2.384	4.480
	$\alpha = 0.5$	0.008	0.005	0.016	4.258	3.024	6.257	3.012	2.384	4.481
	$\alpha = 0.6$	0.008	0.005	0.016	4.252	3.017	***6.252***	3.008	2.380	***4.477***
	$\alpha = 0.7$	0.008	0.005	0.016	4.256	3.020	6.256	3.010	2.381	4.480
	$\alpha = 0.8$	0.008	0.005	0.016	4.257	3.019	6.261	3.010	2.380	4.483
	$\alpha = 0.9$	0.008	0.005	0.016	4.260	3.019	6.266	3.011	2.380	4.486
	$\alpha = 1$	0.008	0.005	0.016	4.267	3.027	6.273	3.012	2.380	4.489
Patchwork	$\alpha = 0$	0.008	0.006	0.014	4.507	3.383	6.409	3.252	2.657	4.643
	$\alpha = 0.1$	0.008	0.006	0.014	4.502	3.376	6.404	3.246	2.650	4.638
	$\alpha = 0.2$	0.008	0.006	0.014	4.499	3.375	6.398	3.243	2.648	4.634
	$\alpha = 0.3$	0.008	0.006	0.014	4.488	3.364	6.386	3.231	2.636	4.642
	$\alpha = 0.4$	0.008	0.006	0.014	4.492	3.373	6.385	3.235	2.641	4.624
	$\alpha = 0.5$	0.008	0.006	0.014	4.497	3.379	6.388	3.232	2.637	4.623
	$\alpha = 0.6$	0.008	0.006	0.014	4.485	3.366	6.377	3.223	2.628	4.615
	$\alpha = 0.7$	0.008	0.006	0.014	4.497	3.382	6.385	3.230	2.635	4.621
	$\alpha = 0.8$	0.008	0.006	0.014	4.502	3.388	6.390	3.234	2.639	4.626
	$\alpha = 0.9$	0.008	0.006	0.014	4.509	3.395	6.398	3.239	2.644	4.632
	$\alpha = 1$	0.008	0.006	0.014	4.527	3.416	6.413	3.233	2.634	4.633
Dict. Learn.		0.008	0.006	0.014	**4.034**	**2.343**	6.440	**2.570**	**1.750**	4.487

Notes. Bias, root mean square error (RMSE), and mean absolute deviation (MAD) are presented, for the overall map (All), the region not in the mask (Out), and the galactic region (Gal.) defined by the mask of Fig. 8. The best results are in bold, the best results among α-shearlets are underlined. Units are in μK.

Table 3. Statistics on the recovery of dark matter halo distribution with the proposed approaches.

Method		Bias	RMSE	MAD
Rotation	$\alpha = 0$	0.0002	3.09	0.83
	$\alpha = 0.1$	0.0002	3.05	0.81
	$\alpha = 0.2$	0.0002	3.02	0.80
	$\alpha = 0.3$	0.0002	3.00	0.80
	$\alpha = 0.4$	0.0002	2.97	0.79
	$\alpha = 0.5$	0.0002	2.95	0.78
	$\alpha = 0.6$	0.0002	2.94	0.78
	$\alpha = 0.7$	0.0002	2.92	0.77
	$\alpha = 0.8$	0.0002	2.92	0.77
	$\alpha = 0.9$	0.0002	2.91	0.77
	$\alpha = 1$	0.0002	2.90	0.77
Patchwork	$\alpha = 0$	0.0002	1.64	0.86
	$\alpha = 0.1$	0.0002	1.58	0.84
	$\alpha = 0.2$	0.0002	1.53	0.82
	$\alpha = 0.3$	0.0002	1.49	0.81
	$\alpha = 0.4$	0.0002	1.45	0.80
	$\alpha = 0.5$	0.0002	1.43	0.79
	$\alpha = 0.6$	0.0002	1.39	0.78
	$\alpha = 0.7$	0.0002	1.37	0.78
	$\alpha = 0.8$	0.0002	1.35	0.77
	$\alpha = 0.9$	0.0002	1.34	0.77
	$\alpha = 1$	0.0002	1.35	0.77
Dict. Learn.		0.0002	**1.32**	**0.72**

Notes. Bias, root mean square error (RMSE), and mean absolute deviation (MAD) are presented. The best results for RMSE and MAD are in bold, the best results among α-shearlets are underlined.

5.2.1. Block artifacts

The first challenge in extending the representation from the Euclidean framework to data defined on the sphere was to avoid the border effects due to considering disjoint charts processed independently. Figure 18 illustrates that all our proposed redundant representations, based on different overlapping charts, are free of these block artifacts when denoising the thermal dust map. A similar result is obtained for denoising the dark matter maps.

5.2.2. Visual inspection

Qualitatively, Figs. 13 and 17 illustrate the different shapes captured by α-shearlets and dictionary learning atoms. In particular, for the thermal dust maps, the noise appears as curvelet-like structures for the α-shearlet approaches, while for the dictionary learning approach, the noise appears both as isotropic and as directional structures.

For the first slice of the dark matter halo distribution simulations, the dictionary learning approach visually seems to best recover the structures in the data, in particular the filamentary structures and the compact cores.

5.2.3. Which approach is best?

This is confirmed quantitatively in Tables 2 and 3 where the dictionary learning approach outperforms both α-shearlet techniques in the denoising of thermal dust (with a multiscale approach) and dark matter halo distribution. For thermal dust, when looking at specific regions (region inside or outside the galactic mask), the rotation-based approach gives, however, the lowest residuals in the galactic

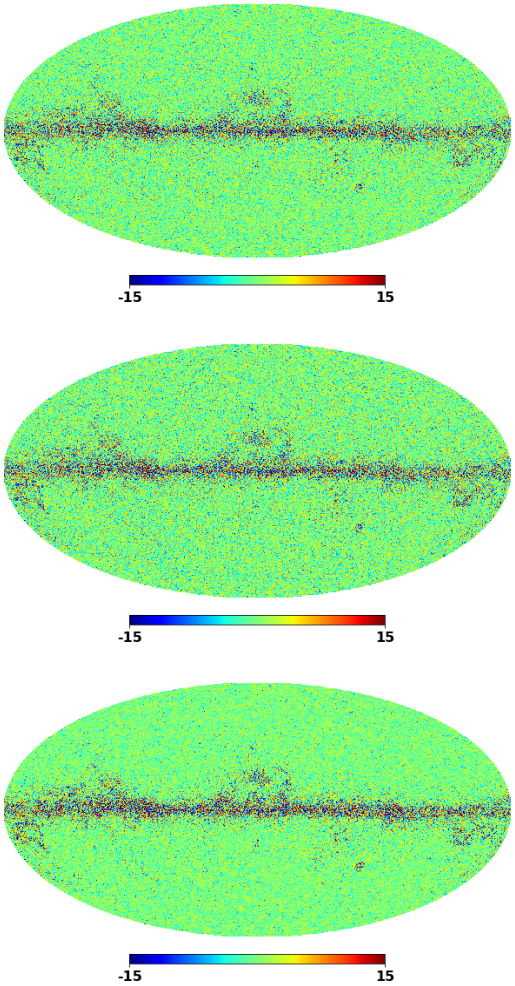

Fig. 14. Residuals for the maps displayed in Fig. 12. Units are in μK.

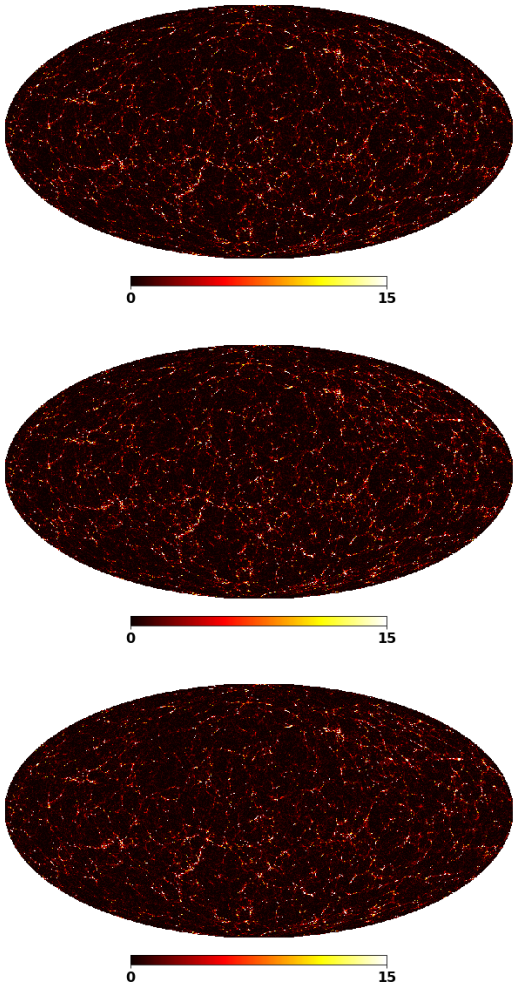

Fig. 15. Denoised dark matter maps for all three approaches. *Top* and *middle panels*: α-shearlet denoising with rotation-based (*top*) or patchwork (*middle*) approach, both with $\alpha = 1$. *Bottom panel*: representation learned with dictionary learning.

region, while using the learned representation gave the best results outside this region. This can be explained by the wide diversity of amplitudes in the galactic plane, not captured in our training set of 200 000 patches for the first wavelet scale, which corresponds only to 0.4% of the total number of patches over the full sky. Improving performance for dictionary learning in the galactic region would require us either to train the dictionary with a larger training set so that it encompasses more patches from the galactic center, or to sample more densely the galactic region than higher galactic latitudes in this training set.

5.2.4. Is the rotation-based or the patchwork approach preferable?

The rotation-based approach outperforms the patchwork approach in the thermal dust denoising scenario, but conversely the patchwork approach outperforms the rotation-based technique in the dark matter halo distribution scenario. The last result is due to the bilinear interpolation performed when resampling the sphere with rotations, which leads to severe approximation errors when the signal varies greatly at the scale of a few pixels.

5.2.5. What is the best α-value?

Tables 2 and 3 show that for α-shearlets in the denoising of thermal dust, $\alpha = 0.6$ (system close to the curvelets) gives the best performance, while for the dark matter halo distribution scenario, $\alpha = 1.0$ (system close to the wavelets) gave the best performance.

However, the second scenario displays a diversity of structures with both high density cores and numerous less dense filaments, with distribution changing in different slices of data corresponding to different redshifts. It would therefore be reductive to investigate a single noise level scenario to set a best α for one of these slices of the data.

We therefore computed for the patchwork approach the non-linear approximation curves for the different slices in redshift. These non-linear approximation curves are illustrated in linear and log scale in Figs. 19 and 20, respectively. These curves illustrate that for large threshold values, corresponding to selected dense core regions, the $\alpha = 0.9$-shearlet system is most suitable. For slice 600 and 605 (higher redshift), when decreasing the threshold, there is a transition from $\alpha = 0.9$ to $\alpha = 0$ (very elongated shearlets) for the best α value. This can be understood as including more and more filamentary structures when the threshold decreases.

For lower redshift slices on the other hand, the best values are obtained more consistently across thresholds for $\alpha = 0.9$ or $\alpha = 1$ because more core structures and less filaments are visible in the data. Overall, this illustrates how adaptive to diverse structures in the data the α-shearlets can be. Furthermore, it

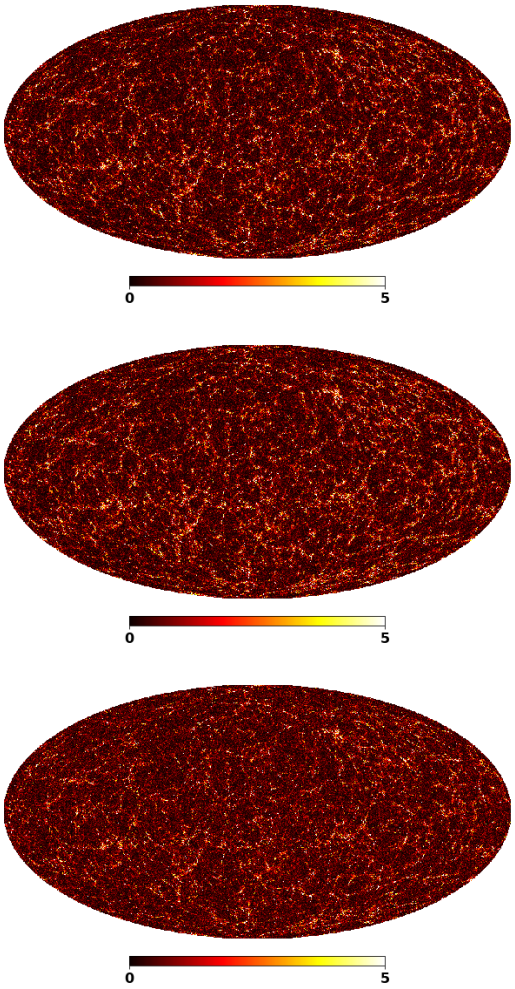

Fig. 16. Amplitude of the residuals for all three approaches, for the dark matter map scenario. *Top* and *middle panels*: α-shearlet denoising with rotation-based (*top*) or patchwork (*middle*) approach, both with $\alpha = 1$. *Bottom panel*: representation learned with dictionary learning.

shows that the anisotropy parameter α can be used to characterize different types of structure present in the data.

5.3. Computing requirements

All codes were run on the same cluster so that we can assess the relative computing time requirements for the three approaches. For the rotation-based approach, on the current python implementation using pyFFTW[3] and also based on a parallelized transform using six cores, denoising a $N_{\rm side} = 2048$ map using five rotations and four scales of decomposition takes about 35 min for $\alpha = 1$ and 1 h for $\alpha = 0$ (the most redundant transform). The time needed to perform the rotation-based approach scales linearly with the number of rotations. In comparison, denoising with the patchwork approach a $N_{\rm side} = 2048$ map using four scales of decomposition (with the same parallelization of the transform as for the rotation-based approach) takes about 9 min for $\alpha = 1$ and 20 min for $\alpha = 0$.

For the multiscale dictionary learning algorithm, computing time for the learning phase ranged from about 2.5 h for scale 3 to about 3.5 h for scale 1, when using our C++ code with four cores for the sparse coding. This increase is due to the low value for $\epsilon^{(1)}$ and large value for the maximal sparsity $K^{(1)}$, even though the training set is smaller than for scale 3. Learning these dictionaries can be performed in parallel, which was done in practice. For the dark matter scenario, the learning took about 65 min. Once the dictionary was learned, sparse coding of all patches took typically from 15 min (scale 1) to about 22 min (scale 3) for the thermal dust map, and 9 min for the dark matter halo distribution, using 24 cores. Overall, the two α-shearlet approaches are therefore easier to set up, with less parameters to optimize that depend directly on the data, and result in faster denoising than the dictionary learning based approach.

[3] https://pypi.org/project/pyFFTW/

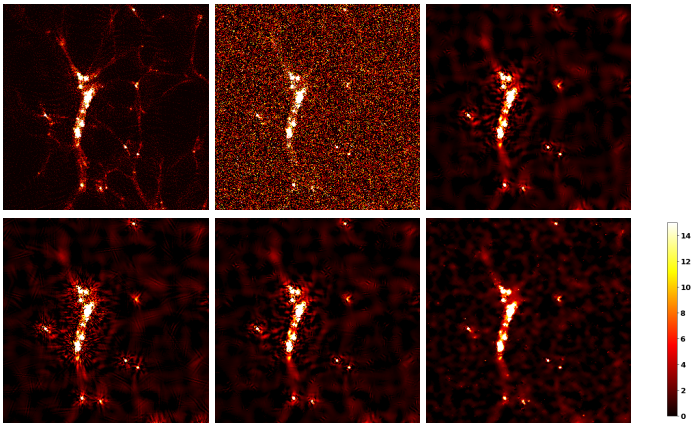

Fig. 17. Dark matter map amplitudes for all three approaches in a zoomed region. *From top to bottom* and *left to right panels*: original map, noisy map, rotation-based approach with $\alpha = 1$, patchwork approach with $\alpha = 0$, patchwork approach with $\alpha = 1$, representation learned from data.

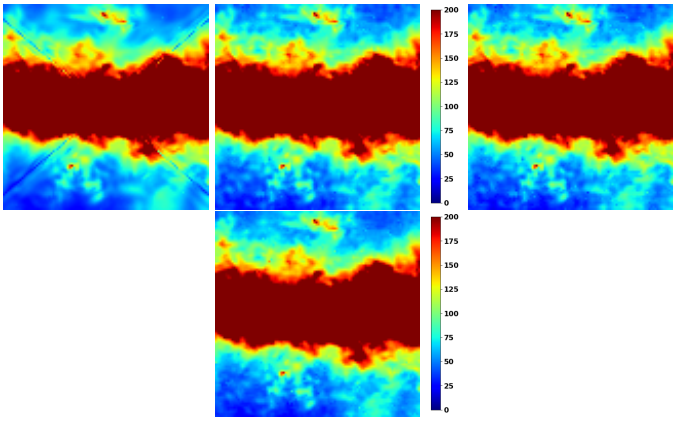

Fig. 18. Cartesian projection of the denoised thermal dust maps centered at the intersection of four faces. *From left to right* and *top to bottom panels*: denoising each face independently using α-shearlets with $\alpha = 1$, restoration via the rotation-based approach, patchwork approach with $\alpha = 1$, dictionary learning with patch width of 12. The colorscale has been stretched to visualize the artifacts seen as a cross-shape discontinuity at the boundaries of the four HEALPix faces in the *upper left panel*. All of our proposed approaches are free from these artifacts. Units are in μK.

6. Conclusions

We have proposed two new types of adaptive representations on the sphere: a patch-based dictionary learning approach and choosing among a parametrized family of representations, the α-shearlets. To extend these constructs from the Euclidean

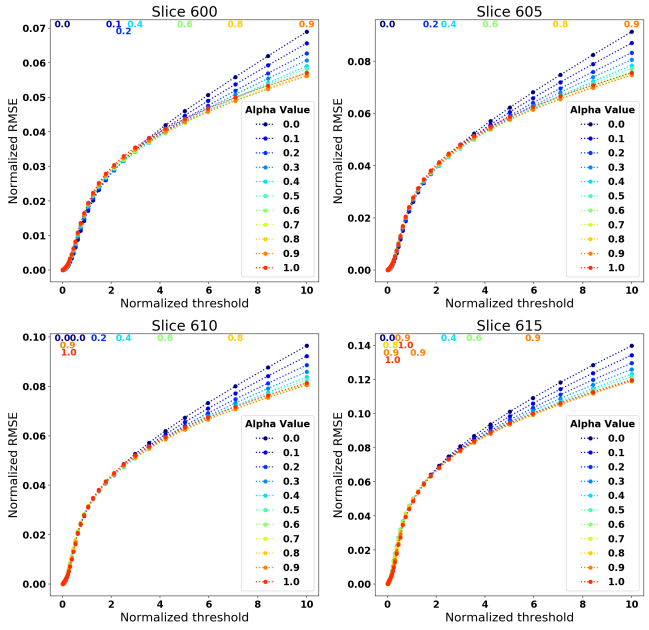

Fig. 19. Normalized non-linear approximation curves for four different slices of the dark matter distribution. For each threshold, the α value corresponding to the lowest approximation error is displayed on the top.

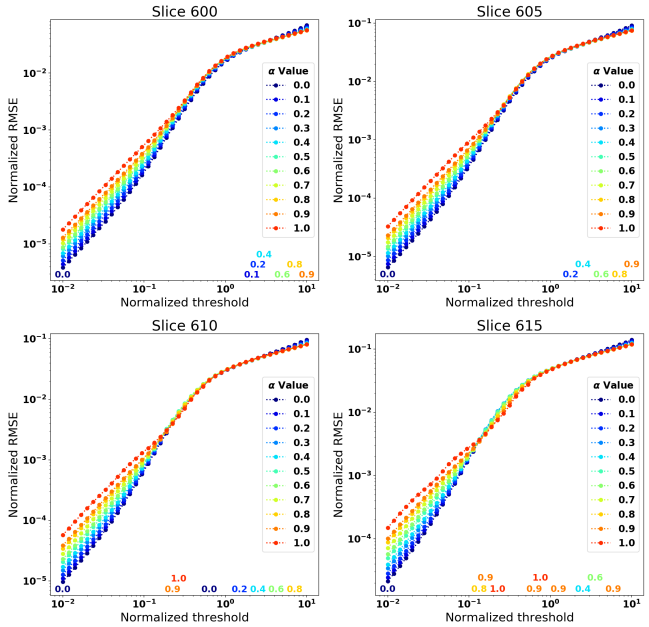

Fig. 20. Normalized non-linear log-approximation curves for four different slices of the dark matter distribution. For each threshold, the α value corresponding to the lowest approximation error is displayed on the bottom.

setting to data defined on the sphere, we proposed to use overlapping charts based on the HEALPix framework. For the dictionary learning technique, a possible multiscale extension was presented by learning dictionaries on each scale after performing a subsampled wavelet decomposition on the sphere. For the α-shearlets, we proposed two approaches to construct the charts: resampling the sphere according to various rotations associated with a partition of unity not sensitive to border effects, or constructing six overlapping charts based on composite extended HEALPix faces.

We evaluated all three approaches by conducting denoising experiments on thermal dust maps, and dark matter maps. Our main findings are as follows:

- thanks to the use of overlapping charts, all of our proposed approaches are free of the block artifacts that typically appear if one naively uses the disjoint HEALPix faces for doing denoising;
- in both scenarios investigated, the dictionary learning approach gave the best performance by providing atoms adapted to the structure present in the images, for a given noise level;
- the performance of the dictionary learning approach depends on setting several hyper-parameters that depend on the signal observed (multiscale or not), and on the training set. This approach therefore requires more computing and tuning time than the other approaches;
- which of the two α-shearlet approaches performed better depended on the chosen scenario; the rotation-based approach involves interpolation, which is detrimental to capturing signals that vary significantly on the scale of just a few pixels, but it achieved better results for the thermal dust simulations;
- for different values of the anisotropy parameter α, the α-shearlet system is adapted to different structures (filaments, dense cores) present in the dark matter halo distribution simulation.

The respective performance of these approaches depends on the criteria used: the dictionary learning approach provided the best denoising results in both scenarios, but has a higher number of parameters to set and requires more computing time; among the α-shearlets, the rotation-based approach is best for smooth signals, but the converse is true for signals with significant variation on the scale of a few pixels. The three proposed approaches can therefore be used to process data living on the sphere, and choosing the "best" approach will depend on the scenario considered as well as the computing resources available.

Reproducible research. In the spirit of reproducible research, we make public our codes for sparse representation systems on the sphere on the common repository[4]. The dictionary learning and alpha-shearlet codes on the sphere are associated with tutorial jupyter notebooks illustrating how to use them for denoising.

Acknowledgements. This work is funded by the DEDALE project, contract no. 665044, within the H2020 Framework Program of the European Commission. The authors thank the Horizon collaboration for making their simulations available.

References

Aharon, M., & Elad, M. 2008, SIAM J. Imaging Sci., 1, 228
Aharon, M., Elad, M., & Bruckstein, A. 2006, Int. Trans. Sig. Proc., 54, 4311
Beckouche, S., Starck, J. L., & Fadili, J. 2013, A&A, 556, A132
Bessolaz, N., & Brun, A. S. 2011, ApJ, 728, 115
Bobin, J., Sureau, F., & Starck, J.-L. 2015, A&A, 583, A92
Bobin, J., Sureau, F., & Starck, J.-L. 2016, A&A, 591, A50
Candès, E., & Donoho, D. 2004, Comm. Pure Appl. Math., 57, 219
Candès, E., Demanet, L., Donoho, D., & Ying, L. 2006, Multiscale Model. Simul., 5, 861
Chan, J. Y. H., Leistedt, B., Kitching, T. D., & McEwen, J. D. 2017, IEEE Trans. Signal Process., 65, 5
Christensen, O. 2016, in An Introduction to Frames and Riesz Bases, 2nd edn. (Cham: Birkhäuser/Springer), Appl. Numer. Harmonic Anal., XXV

[4] github.com/florentsureau/ARES

Daubechies, I. 1992, in Ten Lectures on Wavelets (Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM)), CBMS-NSF Regional Conf. Ser. Appl. Math., 61, XX
Díaz-Hernández, R., Peregrina-Barreto, H., Altamirano-Robles, L., González-Bernal, J. A., & Ortiz-Esquivel, A. E. 2014, Exp. Astron., 38, 193
Díaz-Hernández, R., Ortiz-Esquivel, A., Peregrina-Barreto, H., Altamirano-Robles, L., & González-Bernal, J. 2016, Exp. Astron., 41, 409
Doroshkevich, A. G., Naselsky, P. D., Verkhodanov, O. V., et al. 2005, Int. J. Mod. Phys. D, 14, 275
Elad, M., & Aharon, M. 2006, IEEE Trans. Image Process., 15, 3736
Engan, K., Aase, S. O., & Husoy, J. H. 1999, in ICASSP 1999 Proceedings, IEEE, 5, 2443
Górski, K. M., Hivon, E., & Wandelt, B. D. 1999, Proc. MPA/ESO Conf., Evol. Large Scale, Struct, 37
Górski, K. M., Hivon, E., Banday, A. J., et al. 2005, ApJ, 662, 759
Grohs, P., Keiper, S., Kutyniok, G., & Schäfer, M. 2016, Appl. Comput. Harmon. Anal., 41, 297
Guo, K., & Labate, D. 2007, SIAM J. Math. Anal., 39, 298
Guo, K., Kutyniok, G., & Labate, D. 2006, Wavelets and splines: Athens 2005 (Brentwood, TN: Nashboro Press), Mod. Methods Math., 189
Jenatton, R., Mairal, J., Obozinski, G., & Bach, F. 2011, J. Mach. Learn. Res., 12, 2297
Jost, P., Vandergheynst, P., Lesage, S., & Gribonval, R. 2006, ICASSP 2006 Proceedings, IEEE, 5, V
Kutyniok, G., & Lim, W. 2011, J. Approx. Theor., 163, 1564
Kutyniok, G., & Labate, D. 2012, Applied and Numerical Harmonic Analysis (New York: Birkhäuser/Springer), XX
Labate, D., Lim, W., Kutyniok, G., & Weiss, G. 2005, in Optics and Photonics 2005, Int. Soc. Opt. Photonics, 59140U
Labate, D., Lim, W.-Q., Kutyniok, G., & Weiss, G. 2005b, Wavelets XI, Vol. 5914 (SPIE), 254
Le Pennec, E., & Mallat, S. 2005, IEEE Trans. Image Process., 14, 423
Leistedt, B., McEwen, J. D., Büttner, M., & Peiris, H. V. 2017, MNRAS, 466, 3728
Mairal, J., Elad, M., & Sapiro, G. 2008a, IEEE Trans. Image Process., 17, 53
Mairal, J., Sapiro, G., & Elad, M. 2008b, Multiscale Model. Simul., 7, 214
Mairal, J., Ponce, J., Sapiro, G., Zisserman, A., & Bach, F. R. 2009, Adv. Neural Inf. Process. Syst., 1033
Mairal, J., Bach, F., Ponce, J., & Sapiro, G. 2010, J. Mach. Learn. Res., 11, 19
Mallat, S., & Zhang, Z. 1993, IEEE Trans. Signal Process., 41, 3397
McDermott, S. D., Fox, P. J., Cholis, I., & Lee, S. K. 2016, JCAP, 7, 045
McEwen, J. D. 2015, IEEE Trans. Sig. Proc., submitted, [arXiv:1510.01595]
McEwen, J. D., Feeney, S. M., Peiris, H. V., et al. 2017, MNRAS, 472, 4081
Naidoo, K., Benoit-Lévy, A., & Lahav, O. 2017, MNRAS, 472, L65
Olshausen, B., & Field, D. 1996, Vision Res., 37, 3311
Ophir, B., Lustig, M., & Elad, M. 2011, IEEE Sel. Sign. Process. Top., 5
Pati, Y. C., & Krishnaprasad, P. S. 1993, IEEE Trans. Neural Networks, 4, 73
Peyré, G. 2009, J. Math. Imaging Vision, 34, 17
Planck Collaboration XII. 2016, A&A, 594, A12
Rassat, A., Starck, J.-L., Paykari, P., Sureau, F., & Bobin, J. 2014, JCAP, 8, 006
Rubinstein, R., & Elad, M. 2014, IEEE Trans. Signal Process., 62, 5962
Rubinstein, R., Peleg, T., & Elad, M. 2013, IEEE Trans. Signal Process., 61, 661
Schmitt, J., Starck, J. L., Casandjian, J. M., Fadili, J., & Grenier, I. 2010, A&A, 517, A26
Starck, J.-L., Candès, E., & Donoho, D. 2003, A&A, 398, 785
Starck, J.-L., Moudden, Y., Abrial, P., & Nguyen, M. 2006, A&A, 446, 1191
Starck, J.-L., Moudden, Y., & Bobin, J. 2009, A&A, 497, 931
Starck, J.-L., Murtagh, F., & Fadili, M. J. 2015, Sparse Image and Signal Processing: Wavelets and Related Geometric Multiscale Analysis (Cambridge University Press)
Sureau, F. C., Starck, J.-L., Bobin, J., Paykari, P., & Rassat, A. 2014, A&A, 566, A100
Teyssier, R., Pires, S., Prunet, S., et al. 2009, A&A, 497, 335
van der Walt, S., Colbert, S. C., & Varoquaux, G. 2011, Comput. Sci. Eng., 13, 22
Van Rossum, G., & Drake, Jr., F. L. 1995, Python Tutorial (The Netherlands: Centrum voor Wiskunde en Informatica Amsterdam)
Voigtlaender, F., & Pein, A. 2017, ArXiv e-prints [arXiv:1702.03559v1]
Woiselle, A. 2010, PhD Thesis Paris 7
Woiselle, A., Starck, J.-L., & Fadili, J. 2011, J. Math. Imaging Vision, 39, 121
Zhang, Q., & Li, B. 2010, in 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (IEEE), 2691

Appendix A: Review of Euclidean α-shearlets

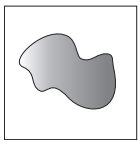

Fig. A.1. Example of a cartoon-like function. Such a function f is smooth, apart from a jump discontinuity along a curve γ. Even though f might be discontinuous along γ, the boundary curve γ itself is required to be smooth.

The α-shearlet family of representations generalizes wavelets and shearlets. Like shearlets – originally introduced in Labate et al. (2005) and Guo et al. (2006) – they are a directionally sensitive multiscale system in $\mathbb{R}^2$ improving upon wavelets when it comes to handling data that is governed by directional features like edges. They are characterized by an anisotropy parameter $\alpha \in [0,1]$, and were designed to yield optimally sparse representations for the class of C^β-cartoon-like functions (Kutyniok & Labate 2012; Kutyniok & Lim 2011; Guo & Labate 2007; Voigtlaender & Pein 2017), a model class for natural images (Candès & Donoho 2004) as illustrated in Fig. A.1.

In the remainder of this section, we briefly explain our motivation for choosing α-shearlet systems, discuss the most important mathematical properties of α-shearlet systems, and then comment on the implementation that we used.

A.1. Motivation

Before giving a formal definition of (α)-shearlet systems, it is instructive to roughly compare the operations used for their construction to the ones used for defining wavelet systems (Daubechies 1992). We recall (see, e.g., Daubechies 1992) that for a scaling function $\phi \in L^2(\mathbb{R}^d)$ and a mother wavelet $\psi \in L^2(\mathbb{R}^d)$, the associated (discrete) wavelet system with sampling density $\delta > 0$ is given by

$$\mathcal{W}(\phi,\psi;\delta) := (\phi(\bullet - \delta k))_{k\in\mathbb{Z}^d} \cup \left(2^{dj/2}\cdot\psi(2^j\bullet - \delta k)\right)_{j\in\mathbb{N}_0,k\in\mathbb{Z}^d}.$$

In other words, the wavelet system consists of all translates of the scaling function ϕ along the lattice $\delta\mathbb{Z}^d$, together with certain translates of the isotropically dilated scaling functions $\psi_j := 2^{dj/2}\psi(2^j\bullet)$. Here, the wavelet ψ_j on the jth scale is translated along the lattice $\delta\cdot 2^{-j}\mathbb{Z}^d$, which is adapted to the "size" of ψ_j.

It is crucial to note that even in dimension $d > 1$, wavelets use the isotropic dilations $x \mapsto 2^j x$, which treat all directions in the same way. Therefore, wavelet systems are not optimally suited for representing functions governed by features with different directions. Admittedly, instead of using a single mother wavelet ψ, it is common to employ wavelet systems that use finitely many mother wavelets $\psi^{(1)},\dots,\psi^{(N)}$; usually these are obtained by choosing each $\psi^{(j)}$ as a certain tensor product of one-dimensional scaling functions and mother wavelets. However, such a modified wavelet system is again only able to distinguish a fixed number of directions, independent of the scale j, and therefore does not allow a satisfactory directional sensitivity.

To overcome this problem, shearlets (like curvelets) use the parabolic dilation matrices $D_j^{(1/2)} := \begin{pmatrix} 2^j & 0 \\ 0 & 2^{j/2}\end{pmatrix}$. More generally, α-shearlets employ the α-parabolic dilation matrices

$$D_j^{(\alpha)} := \begin{pmatrix} 2^j & 0 \\ 0 & 2^{\alpha j}\end{pmatrix} \quad \text{for} \quad j\in\mathbb{N}_0.$$

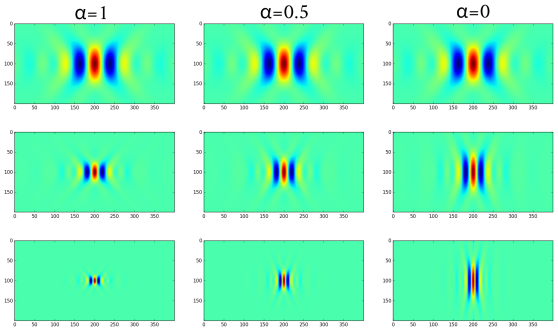

Fig. A.2. Effect of dilating a "prototype function" ψ (shown at the *top* of each column) with the matrices $D_j^{(\alpha)}$ to obtain $\psi(D_j^{(\alpha)}\bullet)$, for different values of the scale j (going from $j = 0$ (*top panels*) to $j = 2$ (*bottom panels*)) and of the "anisotropy parameter" $\alpha \in [0,1]$.

As shown in Fig. A.2, dilating a function ψ with these matrices $D_j^{(\alpha)}$ produces functions $\psi_j^{(\alpha)} = \psi(D_j^{(\alpha)}\bullet)$ that are more elongated along the x_2-axis than along the x_1-axis, where the anisotropy is more pronounced for larger values of α or j. The support of the dilated function satisfies $2^{-j\alpha} \approx \text{height} \approx \text{width}^\alpha$.

It is apparent from Fig. A.2 that for $\alpha < 1$ and large $j \in \mathbb{N}_0$, the functions $\psi_j^{(\alpha)}$ have a distinguished direction. More precisely, if (as in the figure) ψ oscillates along the x_1-axis, then $\psi_j^{(\alpha)}$ is similar to a sharp jump along the x_2-axis. Since we want our dictionary to be able to represent jumps along arbitrary directions, we have to allow for some way of changing the direction of the elements $\psi_j^{(\alpha)}$. The most intuitive way for achieving this is to use rotations, as was done in the construction of (second generation) curvelets (Candès & Donoho 2004). However, later on it was noted in Labate et al. (2005) and Guo et al. (2006) that from an implementation point of view, rotations have the disadvantage that they do not leave the digital grid $\mathbb{Z}^2$ invariant. Therefore, instead of rotations, (α)-shearlets use the shearing matrices

$$S_x := \begin{pmatrix} 1 & 0 \\ x & 1\end{pmatrix}$$

to adjust the direction of the functions $\psi_j^{(\alpha)}$. However the shearing matrices S_x, $x \in (-\infty,\infty)$ can never cause an effect similar to a rotation with angle θ for $|\theta| > 90°$. Therefore, for the definition of a cone-adapted shearlet system, one only uses shearings corresponding to rotations with angle $|\theta| \le 45°$, and one then uses a modified mother shearlet $\psi^\natural$ to cover the remaining directions.

Collecting all previously described constructs, the cone-adapted α-shearlet system with sampling density $\delta > 0$, associated to a low-pass filter $\varphi \in L^2(\mathbb{R}^2)$ and mother shearlet $\psi \in L^2(\mathbb{R}^2)$, is defined as

$$\mathrm{SH}_\alpha(\varphi,\psi;\delta) := (\varphi(\bullet - \delta k))_{k\in\mathbb{Z}^2} \cup \left(2^{(1+\alpha)j/2}\,\psi(R^\iota D_j^{(\alpha)} S_\ell \bullet - \delta k)\right)_{(j,\ell,\iota)\in I,k\in\mathbb{Z}^2}, \quad \text{(A.1)}$$

with $R := \begin{pmatrix} 0 & 1 \\ 1 & 0\end{pmatrix}$, and

$$I := I^{(\alpha)} := \left\{(j,\ell,\iota) \in \mathbb{N}_0\times\mathbb{Z}\times\{0,1\} \,:\, |\ell| \le \lceil 2^{j(1-\alpha)}\rceil\right\}.$$

For brevity, let us set $\psi^{(\alpha)}_{j,\ell,\iota} := 2^{(1+\alpha)j/2}\,\psi\left(R^\iota\, D^{(\alpha)}_j\, S_\ell\, \bullet\right)$, and observe with this notation that

$$2^{(1+\alpha)j/2}\,\psi\left(R^\iota\, D^{(\alpha)}_j\, S_\ell\, \bullet - \delta k\right) = \psi^{(\alpha)}_{j,\ell,\iota}\left(\bullet - \delta A^{-1}_{j,\ell,\iota} k\right), \tag{A.2}$$

with $A_{j,\ell,\iota} := R^\iota\, D^{(\alpha)}_j\, S_\ell$.

A.2. Mathematical properties

The most basic property of α-shearlets that we are interested in is that they indeed form a (redundant) representation system for $L^2(\mathbb{R}^2)$. In mathematical terms, this means that the α-shearlet system forms a frame (Christensen 2016), for a suitable choice of the generators φ, ψ. In particular, if $\varphi, \psi \in L^2(\mathbb{R}^2)$ have compact support and satisfy certain decay and smoothness conditions (see Voigtlaender & Pein 2017, Theorem 5.10 for details), then there is a "minimal sampling density" $\delta_0 > 0$, such that the α-shearlet system is indeed a frame for $L^2(\mathbb{R}^2)$, for all $0 < \delta \leq \delta_0$.

The main motivation for introducing (α)-shearlets was the need for a representation system better adapted to data governed by directional features, which are often present in natural and in astronomical images. One key result relates (α)-shearlets to $C^{1/\alpha}$-cartoon-like functions. Roughly speaking, a function $f \in L^2(\mathbb{R}^2)$ is called a C^β-cartoon-like function, written $f \in \mathcal{E}^\beta(\mathbb{R}^2)$ (with $\beta \in (1, 2]$), if $f = f_1 + f_2 \cdot \mathbb{1}_B$ for certain $f_1, f_2 \in C^\beta_c([0,1]^2)$ and such that the set $B \subset [0,1]^2$ has a boundary curve of regularity C^β. For a more formal definition, we refer to Voigtlaender & Pein (2017, Definition 6.1).

Using this notion, we have the result that the best N-term approximation error with such a frame of α-shearlets (that is, the smallest approximation error obtained by a linear combination of N α-shearlets) is decaying at (almost) the best rate that any dictionary Ψ can reach for C^β-cartoon-like functions (see Voigtlaender & Pein (2017, Theorem 6.3) for a more precise formulation of this result). To obtain this optimal approximation rate, the anisotropy parameter α needs to be adapted to the regularity β of the C^β-cartoon-like functions, that is, $\alpha = 1/\beta$. In general, given a certain data set, or a certain data model, different types of α-shearlet systems will be better adapted to the given data than other α'-shearlet systems.

We close our discussion of the mathematical properties of α-shearlet systems with a brief discussion of the frequency concentration of such systems. To this end, assume for the moment that the "mother shearlet" ψ is concentrated in frequency to the set

$$Q := \{\xi \in \mathbb{R}^2 \,:\, 3^{-1} \leq |\xi_1| \leq 3 \text{ and } |\xi_2| \leq |\xi_1|\},$$

which is a union of two opposing "wedges" (highlighted in green in Fig. A.3). From elementary properties of the Fourier transform, one then sees that each α-shearlet $\psi^{(\alpha)}_{j,\ell,\iota}$ has frequency support in $S^T_\ell D^{(\alpha)}_j R^\iota Q$, where we denote by A^T the transpose of a matrix A. The resulting coverings of the frequency plane for different values of the anisotropy parameter α are shown in Fig. A.3.

Together, Figs. A.2 and A.3 show that the parameter α has three different, but related effects:

- It affects the "shape" of the elements of the α-shearlet system. Indeed, Fig. A.2 shows that height $\approx$ width$^\alpha$.
- It affects the directional selectivity: as seen in Fig. A.3, on scale j, an α-shearlet system can distinguish about $2^{(1-\alpha)j}$ different directions.
- It affects the frequency support of the elements of the α-shearlet system (see Fig. A.3).

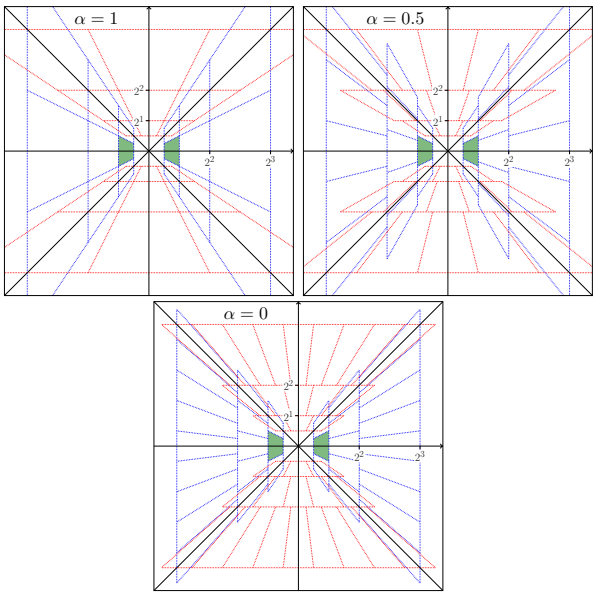

Fig. A.3. Frequency concentration of α-shearlets for different values of α. One sees that each "dyadic annulus" $\{\xi : |\xi| \asymp 2^j\}$ is split into a number $N^{(\alpha)}_j$ of "wedges" representing the different directions. In fact, $N^{(\alpha)}_j \asymp 2^{(1-\alpha)j}$.

A.3. Implementation

The git repository of our implementation of the Euclidean α-shearlet transform can be found online[5], with extensive documentation[6]. Our software package is implemented in Python3 (Van Rossum & Drake 1995), using NumPy (van der Walt et al. 2011).

In this section, we give a rough overview over what the transform computes, and how it can be used. Our software package implements two different versions of the α-shearlet transform: a fully-sampled (non-decimated) version, and a subsampled (decimated) version. For the fully-sampled version, the computed coefficients are the (discrete) convolutions $\varphi * f$ and $\psi^{(\alpha)}_{j,\ell,\iota} * f$ (for a certain range of scales $j = 0, \ldots, j_{\max}$), where the filters φ and $\psi^{(\alpha)}_{j,\ell,\iota}$ are chosen as in Eqs. (A.1) and (A.2). Thus, for a given input image $f \in \mathbb{C}^{N\times N}$, the resulting coefficients form a three-dimensional tensor of dimension $N_{\alpha,j_{\max}} \times N \times N$, where the integer $N_{\alpha,j_{\max}}$ is the total number of α-shearlet filters that is used, and where each $N \times N$ component of the tensor is the discrete convolution of f with one of the α-shearlet filters. When considering $j_{\max}$ many scales (i.e., $j = 0, \ldots, j_{\max} - 1$) and if $\alpha < 1$, then

$$N_{\alpha,j_{\max}} = 1 + 2 \cdot \sum_{j=0}^{j_{\max}-1} \#\{-\lceil 2^{(1-\alpha)j}\rceil, \ldots, \lceil 2^{(1-\alpha)j}\rceil\} \asymp 2^{(1-\alpha)j_{\max}}. \tag{A.3}$$

In particular, for $\alpha = 0$, $N_{0,j_{\max}} \asymp 2^{j_{\max}}$, so that the redundancy of the fully sampled α-shearlet frame grows very quickly when increasing the number of scales.

To motivate the subsampled transform, we note that according to Eq. (A.1), the α-shearlet system does not contain all translations of the functions φ and $\psi^{(\alpha)}_{j,\ell,\iota}$. Rather, φ is shifted along

[5] github.com/dedale-fet/alpha-transform

[6] Available at rawgit.com/dedale-fet/alpha-transform/master/build/html/AlphaTransform.html

the lattice $\delta\mathbb{Z}^2$, and – as seen in Eq. (A.2) – $\psi^{(\alpha)}_{j,\ell,\iota}$ is shifted along the lattice $\delta A^{-1}_{j,\ell,\iota}\mathbb{Z}^2$, with $A_{j,\ell,\iota} = R^\iota D^{(\alpha)}_j S_\ell$. Effectively, this means that the full convolution $f * \psi^{(\alpha)}_{j,\ell,\iota}$ is only sampled at certain points, where the sampling density gets more dense as the scale j increases. The subsampled version of the α-shearlet transform computes these coefficients. Internally, this is achieved by using the "frequency wrapping" approach outlined in Candès et al. (2006, Sects. 3.3 and 6), Woiselle (2010, Chapter 4), and Woiselle et al. (2011) for the case of the curvelet transform. Since each convolution is sampled along a different lattice, the subsampled transform of a given image f is a list of rectangular matrices of varying dimension. This will become clearer in the example below. One can show for the subsampled transform that the total number $M = M(\alpha, j_{\max}, N)$ of α-shearlet coefficients for an $N \times N$ image is bounded, that is, $M(\alpha, j_{\max}, N) \leq M_0 \cdot N^2$, with M_0 independent of α, $j_{\max}, N$. This is in stark contrast to the fully sampled transform (at least for $\alpha < 1$), where the total number of coefficients is $\approx 2^{(1-\alpha)j_{\max}} \cdot N^2$ (see Eq. (A.3)).

The main effect of choosing the fully sampled transform is that one gets a translation-invariant transform (i.e., taking the transform of a shifted image is the same as shifting each component of the coefficient tensor), and the increased redundancy. This increased redundancy can actually be beneficial for certain tasks like denoising, but it can greatly impact the memory footprint and the runtime: computations using the subsampled transform are usually much faster and require much less memory, but yield slightly worse results.

We close this section with a short IPython session showing how our implementation of the α-shearlet transform can be used.

```
>>> # Importing necessary packages
>>> from AlphaTransform import AlphaShearletTransform
    as AST
>>> import numpy as np; from scipy import misc

>>> im = misc.face(gray=True); im.shape
(768, 1024)

>>> # Setting up the transform.
>>> trafo = AST(im.shape[1], im.shape[0], [0.5]*3,
    subsampled=False, verbose=False, real=True) # 1

>>> # Computing the alpha-shearlet coefficients
>>> coeff = trafo.transform(im); print(type(coeff));
    print(coeff.shape) # 2
<class 'numpy.ndarray'>
(27, 768, 1024)
>>> trafo.indices # 3
[-1,
(0, -1, 'h'), (0, 0, 'h'), (0, 1, 'h'),
(0, 1, 'v'), (0, 0, 'v'), (0, -1, 'v'),
(1, -2, 'h'), (1, -1, 'h'), (1, 0, 'h'), ... ]

>>> recon = trafo.inverse_transform(coeff) # 4
>>> np.allclose(recon, im)
True

>>> # Setting up the subsampled transform.
>>> trafo2 = AST(im.shape[1], im.shape[0], [0.5]*3,
    subsampled=True, verbose=False, real=False) # 5

>>> # Computing the subsampled alpha-shearlet
    coefficients
>>> coeff2 = trafo2.transform(im);
    print(type(coeff2)); print(type(coeff2[0]));
    print(coeff2[0].shape); print(coeff2[1].shape) #
    6
<class 'list'>
<class 'numpy.ndarray'>
(129, 129)
(364, 161)
>>> trafo2.indices # 7
[-1,
(0, -1, 'r'), (0, 0, 'r'), (0, 1, 'r'),
(0, 1, 't'), (0, 0, 't'), (0, -1, 't'),
(0, -1, 'l'), (0, 0, 'l'), (0, 1, 'l'),
(0, 1, 'b'), (0, 0, 'b'), (0, -1, 'b'),
(1, -2, 'r'), (1, -1, 'r'), (1, 0, 'r'), ... ]
>>> recon2 = trafo2.inverse_transform(coeff2);
    np.allclose(recon2, im)
True
>>> print(trafo.redundancy); print(trafo2.redundancy)
    # 8
27
12.08676528930664
```

In the line marked with #1, we set up the α-shearlet transform object `trafo`. Roughly speaking, this precomputes all necessary α-shearlet filters, which are stored in the `trafo` object. The first two parameters of the constructor simply determine the shape of the images for which the `trafo` object can be used, while the third parameter determines the number of scales $j_{\max}$ to be used, as well as the value of the anisotropy parameter α. Passing `[alpha_0] * N` will construct an α-shearlet transform with N scales (plus the low-pass) and with α given by `alpha_0`. The `verbose` parameter simply determines how much additional output (like a progress bar) is displayed. The `subsampled` parameter determines whether the non-decimated, or the decimated transform is used. Finally, the `real` parameter determines whether real-valued or complex-valued α-shearlet filters are used. Essentially, real-valued filters have frequency support in the union of two opposing wedges (as shown in Fig. A.3), while for complex-valued filters, one gets two filters for each real-valued one: one complex-valued filter has frequency support in the "left" wedge, while the other one is supported in the "right" wedge.

In line #2, we use the `transform()` method of the constructed `trafo` object to compute the α-shearlet transform of `im`. As seen, the result is an ordinary NumPy array of dimension $N_{\alpha,j_{\max}} \times N_1 \times N_2$, where the input image has dimension $N_1 \times N_2$, and where $N_{\alpha,j_{\max}}$ is the total number of α-shearlet filters used by the transform.

The `indices` property of the `trafo` object (see line #3) can be used to determine to which α-shearlet filter the individual components of the `coeff` array are associated. The value `-1` represents the low-pass filter, while a tuple of the form `(j, l, c)` represents the shearlet filter $\psi^{(\alpha)}_{j,l,\iota}$ as in Eq. (A.1), where $\iota = 0$ if `c` is `'h'` (which stands for the horizontal frequency cone), and where $\iota = 1$ if `c` is `'v'` (vertical frequency cone).

To explain the differences between the fully sampled and the subsampled transform, in line #5, we set up a subsampled transform object `trafo2`. The only difference to the construction of the `trafo` object is that we pass `subsampled=True`, and `real=False`. The reason for this second change is that – at least with the current implementation – the subsampled transform can only be used with complex-valued shearlet filters. We then compute the coefficients (see line #6) just as for the fully sampled transform. We note, however, that the coefficients for the fully sampled transform were a single three-dimensional NumPy

array. For the subsampled transform, however, the coefficients are a list of two-dimensional NumPy arrays. The reason for this is that the number of coefficients varies from scale to scale for the subsampled transform.

The `indices` property (see line #7) for the subsampled transform also differs from that of the fully sampled transform. The reason for this is that we use complex shearlets; therefore, the frequency plane is divided into four cones (top, or `'t'`; right, or `'r'`; bottom, or `'b'`; and left, or `'l'`), instead of the two cones that are used for real-valued shearlet filters.

The main advantage of the subsampled transform is revealed in line #8: the redundancy (that is, the number of α-shearlet coefficients divided by the number of pixels of the input image) for the subsampled transform is much lower, which leads to a lower memory consumption and faster computation times. While the advantage of the subsampled transform might not be overwhelming in the given example, it becomes more pronounced if one uses a larger number of scales. For instance, if we use four scales instead of three, then the redundancy of the fully sampled transform is 41, while that of the subsampled transform is only $\approx$11.4.