Astronomy & Astrophysics

LCS: A learnlet-based sparse framework for blind source separation

V. Bonjean[1],[*] , A. Gkogkou[1] , J.-L. Starck[1,2] , and P. Tsakalides[1]

[1] Institutes of Computer Science and Astrophysics, Foundation for Research and Technology Hellas (FORTH), Greece
[2] Université Paris-Saclay, Université Paris Cité, CEA, CNRS, AIM, 91191 Gif-sur-Yvette, France

ABSTRACT

Blind source separation (BSS) plays a pivotal role in modern astrophysics by enabling the extraction of scientifically meaningful signals from multi-frequency observations. Traditional BSS methods, such as those that rely on fixed wavelet dictionaries, enforce sparsity during component separation but can fall short when faced with the inherent complexity of real astrophysical signals. In this work, we introduce the learnlet component separator (LCS), a novel BSS framework that bridges classical sparsity-based techniques and modern deep learning. LCS utilises the learnlet transform – a structured convolutional neural network designed to serve as a learned, wavelet-like multi-scale representation. This hybrid design preserves the interpretability and sparsity-promoting properties of wavelets while gaining the adaptability and expressiveness of learned models. The LCS algorithm integrates this learned sparse representation into an iterative source separation process, enabling the effective decomposition of multi-channel observations. While conceptually inspired by sparse BSS methods, LCS introduces a learned representation layer that significantly departs from classical fixed-basis assumptions. We evaluated LCS on both synthetic and real datasets and in this paper demonstrate its superior separation performance compared to state-of-the-art methods (average gain of about 5 dB on toy model examples). Our results highlight the potential of hybrid approaches that combine signal processing priors with deep learning to address the challenges of next-generation cosmological experiments.

Key words. methods: data analysis – methods: statistical – techniques: image processing

1. Introduction

The separation of astrophysical components from multi-frequency sky observations is a fundamental task in observational cosmology and astrophysics. It enables the isolation of a source signal from a complex mixture of astrophysical foregrounds and instrumental noise. A prominent example is the extraction of the cosmic microwave background (CMB) from multi-band observations, which played a central role in the success of the *Planck* mission (Planck HFI Core Team 2011). Traditional component separation methods in this context include principal component analysis, which removes the most dominant modes in the frequency domain (Maćkiewicz & Ratajczak 1993), and independent component analysis (ICA) based approaches (Hyvärinen & Oja 2000), which exploit the statistical independence of components. These methods are simple and model-agnostic but can loose or leak part of the cosmological signal. More sophisticated techniques were developed for *Planck*, including internal linear combination (ILC) techniques like needlet-independent linear combination (Delabrouille et al. 2009) and the modified internal linear combination algorithm (Hurier et al. 2013), blind source separation (BSS) methods such as spectral matching ICA (Delabrouille et al. 2003), self-supervised deep learning methods (Bonjean et al. 2024), and sparse techniques like generalised morphological component analysis (GMCA; Bobin et al. 2007). These methods disentangle sources from multi-channel observations by leveraging

assumptions such as statistical independence, spectral diversity, or sparse representability.

In particular, sparse component separation has proven to be a powerful approach for modelling structured, non-Gaussian emissions such as foregrounds that exhibit filamentary or localised morphologies. GMCA exploits the assumption that astrophysical components can be sparsely represented in a certain transform domain. The domains typically used are wavelets, specifically starlet, biorthogonal, or Mexican hat wavelets, and they are chosen based on the complexity of the image under study. By promoting sparsity via morphological diversity, GMCA has been shown to achieve accurate CMB and foreground reconstruction from both simulated and real data in astrophysical applications and beyond (Bobin et al. 2014, 2008; Yong et al. 2009; Xu et al. 2020). However, a key limitation stems from the fixed nature of the wavelet dictionary: while efficient and interpretable, it may not optimally capture the statistical structure of all signal types or adapt to data variability, particularly in non-Gaussian or non-stationary regimes.

In astrophysics, the need for efficient and robust component separation techniques extends beyond the CMB. With the upcoming deployment of radio interferometers like the Square Kilometre Array (SKA; Weltman et al. 2020), there is a renewed interest in separating diffuse emission from target cosmological signals. In particular, HI intensity mapping aims to detect the 21 cm emission from neutral hydrogen across cosmic time, offering a promising probe of large-scale structure, dark energy, and the epoch of re-ionisation (Bull et al. 2015; Santos et al. 2015). However, this faint cosmological signal is contaminated by foregrounds that are several orders of magnitude brighter and include

---

[*] Corresponding author: victor.bonjean40@gmail.com

Galactic synchrotron and free-free emission, as well as extra-galactic point sources (Alonso et al. 2015). Effective foreground cleaning is thus essential for recovering the HI signal with high fidelity. Recent works (Carucci et al. 2020; Makinen et al. 2021; Matshawule et al. 2021; Spinelli et al. 2022; Carucci et al. 2025; Mertens et al. 2024, Gkogkou et al., in prep.) have explored deep learning-based priors, regularisation schemes, and learned sparse representations to tackle the challenge of foreground removal in simulated SKA data.

In parallel, the field of signal processing and machine learning has made significant progress in data-driven sparse modelling. A classical approach in sparse coding is the iterative shrinkage-thresholding algorithm (ISTA; Beck & Teboulle 2009), which solves linear inverse problems with sparsity constraints by iteratively applying a soft-thresholding operation to promote sparse solutions. While ISTA is based on convex optimisation and offers well-characterised convergence guarantees, its iterative nature often results in slow convergence in practice. To address this, Gregor & LeCun (2010) introduced learned ISTA (LISTA), a neural network architecture that unrolls the ISTA iterations into a fixed-depth feed-forward network. By learning the optimal parameters of each iteration (such as the dictionary and thresholding functions) from data, LISTA achieves significantly faster convergence and better adaptability to specific signal classes while maintaining the interpretability of sparse coding. Building upon these ideas, Fahes et al. (2021) developed learned proximal alternating linearised minimisation (LPALM), a hybrid framework that combines deep supervised learning with optimisation-based sparse component separation. LPALM extends the principles of LISTA by integrating it into a proximal optimisation loop tailored for separating astrophysical sources; it has shown promising results in the identification of elemental components in supernova remnants in X-ray data.

Expanding on the concept of hybrid methods that offer both interpretability and learning capability, we investigated the use of neural networks designed to enhance classical signal transforms. Among them, the learnlet transform (Ramzi et al. 2021) introduces a neural network architecture designed to emulate wavelet-like behaviour while learning filters and thresholding parameters directly from data. This approach preserves the interpretability and multi-scale structure of classical wavelets while offering greater adaptability to complex signal distributions. Unlike traditional wavelet-based dictionaries, the learned filters – the learnlets – can adapt to complex and noisy environments, improving the robustness of source separation. While they have already demonstrated effectiveness in deconvolving astronomical images (Akhaury et al. 2022), their integration into structured component separation frameworks remains largely unexplored.

In this work, we introduce the learnlet component separator (LCS), a novel BSS algorithm that embeds a learned sparse representation – called the learnlet transform – into an iterative separation procedure. The learnlet transform is a structured convolutional neural network designed to emulate a wavelet-like multi-scale behaviour while allowing for data-driven adaptation. Its architecture preserves key characteristics of traditional wavelet bases, such as locality, multi-resolution decomposition, and interpretability, while improving upon them through learned filter banks optimised via deep learning for the specific signal class.

The LCS algorithm iteratively estimates the sources that are mixed in each frequency channel by leveraging the sparsity-promoting capabilities of the learnlet transform within a source-mixing matrix estimation loop. This design allows LCS to combine the robustness and theoretical grounding of sparse

representation methods with the adaptability and expressive power of deep neural networks. In this paper we describe the architecture of the LCS, examine its relationship to classical methods like GMCA, and assess its performance through extensive numerical experiments that involve toy models with structured scenes (airplane, boat, and Barbara) and real-world astrophysical data (X-ray supernova remnants and the extraction of CMB and Sunyaev-Zeldovich signals).

Our findings demonstrate that LCS consistently outperforms conventional BSS techniques, especially in complex scenarios involving non-stationary foregrounds or non-ideal mixing conditions (such as in the presence of noise). These results highlight the potential of LCS for future applications to SKA data, where accurate foreground removal will be essential for an accurate extraction of the HI signal in cosmology.

The remainder of this paper is structured as follows. In Sect. 2 we present in detail the learnlet transform, its architecture, and its training on the ImageNet dataset. Section 3 describes the main LCS algorithm for source separation, which incorporates the learnlet transform. Section 4 provides results and comparative evaluations from four distinct experiments involving both toy models and astrophysical images. Finally, Sect. 5 concludes the paper with a discussion of the methods, their relevance to astrophysical applications, and potential future directions.

2. Learnlets

2.1. Architecture

Learnlet is a transform designed for image de-noising, built upon a specialised convolutional neural network architecture, inspired by wavelet transforms and leveraging the representational power of deep learning, as originally introduced in Ramzi et al. (2021). Compared to other state-of-the-art deep learning de-noisers like UNets (Ronneberger et al. 2015), learnlet offers greater interpretability. Its learned filters – referred to as learnlets – form a sparse basis, enabling the imposition of sparsity in subsequent tasks. The transform takes as input a noisy image ($\mathbf{X}$) with $\mathbf{X} = \mathbf{Y} + \mathbf{N}$, where $\mathbf{Y}$ is the ground truth image and $\mathbf{N}$ is a realisation of noise with standard deviation σ, and estimates a de-noised image ($\hat{\mathbf{Y}}$). The architecture is structured as follows:

– We began by applying the starlet transform (Starck et al. 2015) to the input image ($\mathbf{X}$), $\alpha = \Psi\mathbf{X}$, decomposing it into J distinct scales with starlet coefficients (α_j). In our case, $J = 5$.

– The first $J - 1$ scales are sequentially passed through an analysis layer ($\mathcal{A}$), a thresholding layer ($\mathcal{T}$), and a synthesis layer ($\mathcal{S}$). The analysis layer ($\mathcal{A}$) consists of $n_f + 1$ filters with kernel size $h \times w$, producing convolved outputs with the same variance as the input. To ensure this variance consistency, the filters are normalised by the sum of their squared coefficients. A total of $n_f \times (J - 1)$ filters are learned, while the remaining $J - 1$ filters (indexed as $n_f + 1$) are fixed to the identity filter δ. In our implementation, we set $h = w = 5$ and $n_f = 64$.

– Next, a thresholding operation ($\mathcal{T}$) is applied to the output feature maps from the analysis layer for each of the $J - 1$ scales ($\mathcal{A}(\alpha_j)$). We employed hard thresholding, using a threshold of $k_j \times \sigma$ at each scale j, where σ denotes the standard deviation of the input noise. The $J - 1$ threshold parameters (k_j) for each scale are also learned during training and are constrained within the range $0 < k_j < 5$.
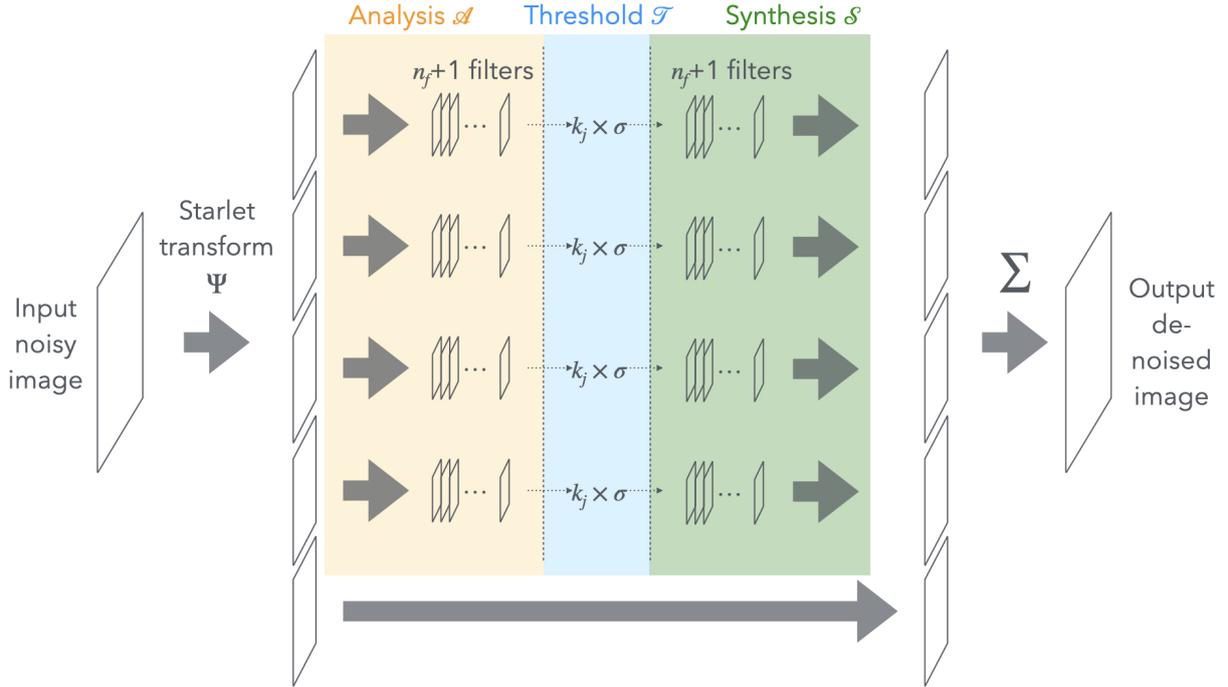
Fig. 1. Architecture of the learnlet network: example with five scales. The first n_f filters of the first $J-1$ scales of both the analysis ($\mathcal{A}$) and the synthesis ($\mathcal{S}$) layers are learned, as are the $J-1$ thresholding parameters (k_j) of the threshold layer ($\mathcal{T}$).

- The thresholded feature maps ($\mathcal{T}(\mathcal{A}(\alpha_j), k_j \times \sigma)$) are then passed to the synthesis layer ($\mathcal{S}$), which also consists of $n_f + 1$ filters of size $h \times w$ for each of the first $J-1$ scales. Of these, $n_f \times (J-1)$ filters are learned, while the remaining $n_f + 1$ filters are set to $\delta - \sum_{i=1}^{n} \mathcal{F}_{\mathcal{S}_{j_i}} * \mathcal{F}_{\mathcal{A}_{j_i}}$, where $\mathcal{F}_{\mathcal{S}_{j_i}}$ and $\mathcal{F}_{\mathcal{A}_{j_i}}$ represent the i-th synthesis and analysis filters at scale j, respectively. This construction ensures that the sum of the outputs from $\mathcal{S}(\mathcal{T}(\mathcal{A}(\alpha_j), k_j \times \sigma))$ exactly reconstructs the input when $\sigma = 0$, guaranteeing perfect reconstruction in the absence of noise.

- The final de-noised map ($\hat{\mathbf{Y}}$) is obtained by summing the de-noised outputs from the first $J-1$ scales along with the unchanged coarse scale (α_J). This is expressed as

$$\hat{\mathbf{Y}} = \alpha_J + \sum_{j=1}^{J-1} \mathcal{S}(\mathcal{T}(\mathcal{A}(\alpha_j), k_j \times \sigma)). \tag{1}$$

A schematic representation of the learnlet architecture is shown in Fig. 1. We re-implemented the original learnlet algorithm from Ramzi et al. (2021) in PyTorch to enable more efficient GPU utilisation. Additionally, we created a user-friendly and well-documented Python class, along with example notebooks, all available on GitHub[1].

2.2. Training on ImageNet

The learnlets must first be trained. To achieve this, we selected a subset of the ImageNet database (Russakovsky et al. 2015), which offers a diverse range of images with rich and complex features – ideal for training generalizable and robust learnlets. The dataset is publicly accessible via their website[2]. Specifically, we used the first 10.000 images from the ImageNet test set,

converting them from red-green-blue to grey scale (with pixel values normalised between 0 and 1) and resizing them to 256×256 pixels. The dataset was then split into 8.000 images for training, 1.000 for validation, and 1.000 for testing.

For all sets, we added synthetic Gaussian noise (**N**) to the images, with the noise standard deviation (σ_N) randomly sampled between 0 and 0.5. This resulted in a collection of noisy images (**X**) and their corresponding clean counterparts (**Y**) such that $\mathbf{X} = \mathbf{Y} + \mathbf{N}$.

We trained the learnlet model using **X** as input and **Y** as the corresponding ground truth output. The network was configured with $n_f = 64$ filters of size 5×5, differing from the original setup in Ramzi et al. (2021), but yielding improved and faster performance. We removed all bias parameters from the convolutional layers. We used an Adam optimiser with a learning rate $\ell_r = 10^{-4}$ and mean squared error (MSE) as the loss function. The training ran on an NVIDIA A100 GPU and completed in approximately 8 hours.

The performance of the learnlet model on the test set is shown in green in Fig. 2. The plot displays the average peak signal-to-noise ratio (PSNR) values (in dB) of the reconstructed images as a function of the input noise standard deviation σ_N. For comparison, we evaluated two UNet architectures (Ronneberger et al. 2015): a lightweight version (UNet-8, shown in blue) initialised with 8 filters, and a more complex variant (UNet-64, shown in orange) initialised with 64 filters. The numbers in parentheses in the legend indicate the total number of trainable parameters in each model, highlighting the markedly lower complexity of learnlet (12 800 parameters) compared to the UNet architectures, which range from 239 440 (UNet-8) to 15 319 680 (UNet-64).

Overall, the learnlet model delivers strong performance, matching that of a small UNet-8 and even outperforming the larger UNet-64 at very low values of σ_N, thanks to its ability

---

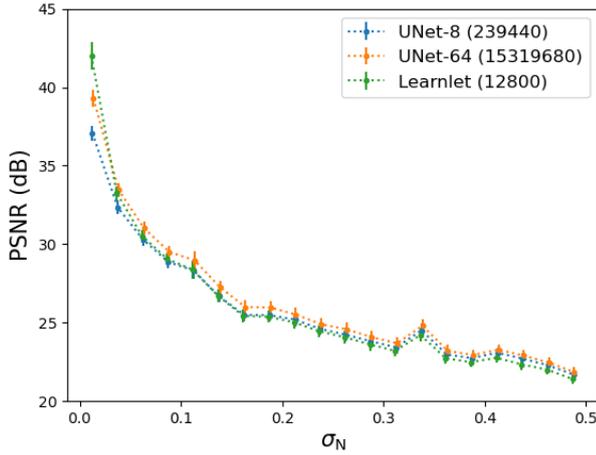[1] https://github.com/vicbonj/learnlet.git
[2] https://www.image-net.org

Fig. 2. PSNR of the de-noised test images for learnlet (green), UNet-8 (blue), and UNet-64 (orange) plotted against the input noise level (σ_N). The legend includes the number of trainable parameters for each network.

to achieve exact reconstruction. Although learnlets may perform slightly worse than a UNet, they offer several advantages: they are fully trustworthy, provide exact reconstruction, are mathematically well defined as frames, inherit the convergence properties of wavelets in proximal iterative algorithms, and require ~100 times fewer parameters. Hence, learnlets provide an interesting tradeoff between efficiency and robustness.

In the appendix we show an example of learned filters and the evolution of the thresholding parameters as a function of σ_N. The learnlets form an optimised sparse basis tailored to natural image structures. Building on this foundation, we introduce a BSS algorithm called LCS, which shares structural similarities with GMCA but incorporates key novelties, which are discussed in the following sections.

3. The learnlet component separator (LCS)

In this section we describe in detail the algorithm LCS in the context of BSS.

3.1. Formalism of BSS

Blind source separation is a core challenge in signal processing. Its aim is to recover unknown source signals (S) from observed mixtures ($\mathbf{Y}$) that may also include noise ($\mathbf{N}$), all without prior knowledge of the mixing process ($\mathbf{A}$). Mathematically, this can be expressed as

$$\mathbf{Y} = \mathbf{A} \cdot \mathbf{S} + \mathbf{N}. \tag{2}$$

BSS has broad applications across multiple fields, including biomedical signal analysis (e.g. electroencephalogram, EEG, and functional magnetic resonance imaging, fMRI), audio processing (e.g. speech separation), and notably in astrophysical imaging, for tasks such as CMB and Sunyaev-Zeldovich (SZ) extraction, HI signal recovery, and source detection.

3.2. LCS main algorithm

The algorithm iteratively estimates both the sources, de-noised using the learnlet transform, and the mixing matrix, drawing inspiration from GMCA. LCS employs an iterative scheme that

alternates between updating the source estimates and refining the mixing matrix:

- **S** estimate: we first estimate $\hat{\mathbf{S}}$ with a fixed estimate of the mixing matrix ($\hat{\mathbf{A}}$), by minimising the equation

$$\min_{\mathbf{S}} \|\mathbf{Y} - \hat{\mathbf{A}} \cdot \mathbf{S}\|_2^2 + \lambda \|\mathcal{L}_\mathbf{S}\|_p, \tag{3}$$

where the second term is obtained, with p=0, by performing a ℓ_0-norm constraint on **S** to promote sparsity. This is done by the hard-thresholding operation from the learnlet transform ($\mathcal{L}_{\mathbf{S}_i}$) for each source component ($\mathbf{S}_i$).

- **A** estimate: the estimate of the mixing matrix, $\hat{\mathbf{A}}$, is refined by minimising the equation

$$\min_{\mathbf{A}} \|\mathbf{Y}^{\text{no-coarse}} - \mathbf{A} \cdot \mathcal{L}_{\hat{\mathbf{S}}}^{\text{no-coarse}}\|_2^2, \tag{4}$$

where the coarse scales have been removed to eliminate low-frequency spatially correlated components that could otherwise introduce ambiguity and degrade the quality of source separation.

Convergence is typically reached when successive iterations produce only minimal changes in all $\hat{\mathbf{S}}$, indicating that both the source separation and the estimate of **A** have stabilised. After the minimisation has converged, an additional post-processing step is performed to de-noise each estimated source $\hat{\mathbf{S}}_i$ using the learnlet transform as a de-noiser, with the noise level estimated individually for each component. The noise level is derived from the known noise covariance ($\Sigma_\mathbf{N}$) and the pseudo-inverse of the estimated mixing matrix ($\hat{\mathbf{A}}$), leading to the computation of the source covariance:

$$\Sigma_{\hat{\mathbf{S}}} = \hat{\mathbf{A}}^+ \Sigma_\mathbf{N} (\hat{\mathbf{A}}^+)^\mathsf{T}. \tag{5}$$

The main differences of LCS and GMCA are as follows:
- Fixed thresholding: In LCS, the threshold is fixed, eliminating the need for a threshold decay strategy, which in GMCA often depends on the image type. We tried several multiplicative factors of the σ_{NMAD} of the sources and found that $1.5 \times \sigma_{\text{NMAD}}$ gives the best results for most of our applications.
- Hard versus soft thresholding: LCS employs hard-thresholding instead of the soft-thresholding used in GMCA, thereby avoiding bias introduced by the latter.
- Source update domain: LCS updates the source estimates $\hat{\mathbf{S}}$ in the real space, whereas GMCA performs this update in the wavelet domain – an approximation if the decomposition is not orthogonal.
- Post-processing de-noising step: LCS includes an additional final de-noising step applied to the estimated sources $\hat{\mathbf{S}}$.

The learnlets $\mathcal{L}_{\hat{\mathbf{S}}_i}$ are individually trained for each of the component i to enhance separation quality by learning optimal bases and thresholding parameters tailored to the morphology, features, and contrasts of each component. However, a shared basis $\mathcal{L}$ can also be used for all components. In this work we explored the use of such a common basis by employing learnlets pre-trained on ImageNet (see Sect. 2.2) as a general-purpose option. Additionally, we investigated component-specific training using numerical simulations or texture datasets, leveraging transfer learning with weights initialised from the ImageNet-trained models.
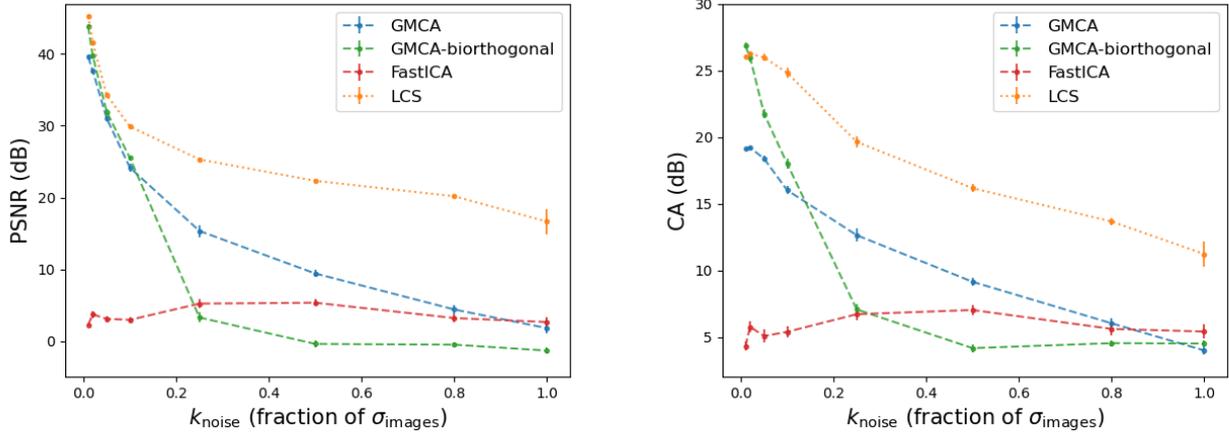
Fig. 3. Performance comparison of various BSS methods on standard test images: LCS (orange), GMCA (blue), GMCA-biorthogonal (green), and FastICA (red). Left panel: mean PSNR values on the source matrix ($\mathbf{S}$) as a function of the input noise level (k_{noise}). Right panel: corresponding mean component angle (CA) values on the mixing matrix ($\mathbf{A}$). Each point represents the average and standard error over ten independent realisations of both the mixing mixing matrices ($\mathbf{A}$) and the noise.

3.3. Quality assessment

For all experiments presented in this paper, we assessed the quality of the source reconstruction $\mathbf{S}$ using the PSNR in decibels. PSNR is a widely used metric in signal processing that quantifies the accuracy of image reconstruction and is defined as

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{\max^2}{\text{MSE}}\right), \tag{6}$$

where max denotes the maximum possible pixel value in the image (set to 1 in our case), and MSE represents the mean squared error between the estimated sources $\hat{\mathbf{S}}$ and the ground truth sources $\mathbf{S}$.

The quality of the mixing matrix reconstruction was evaluated using a metric that we named 'criterion on $\mathbf{A}$' (CA), which measures how closely the product of the pseudo-inverse of the estimated mixing matrix ($\hat{\mathbf{A}}$) with the true matrix ($\mathbf{A}$) approximates the identity matrix. Ideally, this product satisfies $\hat{\mathbf{A}}^+ \cdot \mathbf{A} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. The CA metric, expressed in decibels, is defined as

$$\text{CA} = -10 \cdot \log_{10}\left(\frac{1}{N} \left\| \hat{\mathbf{A}}^+ \cdot \mathbf{A} - \mathbf{I} \right\|_F\right), \tag{7}$$

where N is the number of channels and $\| \cdot \|_F$ denotes the Frobenius norm.

4. Numerical experiments

We conducted four distinct experiments to evaluate the source separation capabilities of learnlets. Two experiments use a shared, general basis $\mathcal{L}$ derived from learnlets trained on ImageNet (see Sect. 2.2), while the other two use individually retrained bases $\mathcal{L}_i$ tailored to specific components. We evaluated both approaches, using standard test photographic images and astrophysical imagery.

4.1. Experiments using a common basis ($\mathcal{L}$)

4.1.1. Experiments with standard test images

In this experiment, the target source signal ($\mathbf{S}$) consists of three components, represented by grey-scale images of Barbara, an airplane, and a boat, common reference points in signal processing for de-noising tasks. Each image has a resolution of 256×256 pixels and is normalised to a [0, 1] range. We performed multiple component separation runs using a randomly generated mixing matrix ($\mathbf{A}$), sampled from a uniform distribution in [0, 1] with six channels, and constrained to have a condition number below 5. The noise matrix ($\mathbf{N}$) varies between realisations. We repeated the experiment ten times for each noise level, where the noise in the observations $\mathbf{Y}$ is scaled as $k_{noise} \times \sigma_{images}$, with k_{noise} taking values in 0.01, 0.02, 0.05, 0.1, 0.25, 0.5, 0.8, 1, and σ_{images} being the standard deviation of the original sources ($\mathbf{S}$). For each realisation, we estimated $\hat{\mathbf{S}}$ and $\hat{\mathbf{A}}$ using LCS and compared the results against two different variants of GMCA: (i) standard GMCA using starlets, and (ii) GMCA-biorthogonal, which uses bi-orthogonal wavelets. We also compared our results with those of the FastICA algorithm. The results are presented in Fig. 3.

In the source reconstruction results shown on the left panel, LCS outperforms all GMCA variants by 3–5 dB in the low-noise regime ($k_{noise} < 0.1$), while FastICA performs poorly given the specific image and channel configuration. As the noise level increases ($k_{noise} > 0.1$), LCS delivers a substantial improvement, achieving gains of 10–15 dB over the other methods. In the mixing matrix reconstruction results shown in the right panel, LCS surpasses all other methods by 5–15 dB across most noise levels. The only exception is at the lowest noise level ($k_{noise} = 0.01$), where GMCA-biorthogonal also performs exceptionally well, slightly outperforming LCS by about 1 dB.

4.1.2. Experiments with astrophysical images

In this experiment, we used X-ray astronomical images from Fahes et al. (2021), which are publicly available on their GitHub repository[3]. We compared them with their LPALM algorithm – a semi-supervised deep learning method for BSS that achieved state-of-the-art performance on this dataset. LPALM is a supervised approach in which the proximal operator is learned from paired mixtures and corresponding ground-truth sources using an unrolled, end-to-end optimisation framework. The model is trained specifically to recover both the source components and
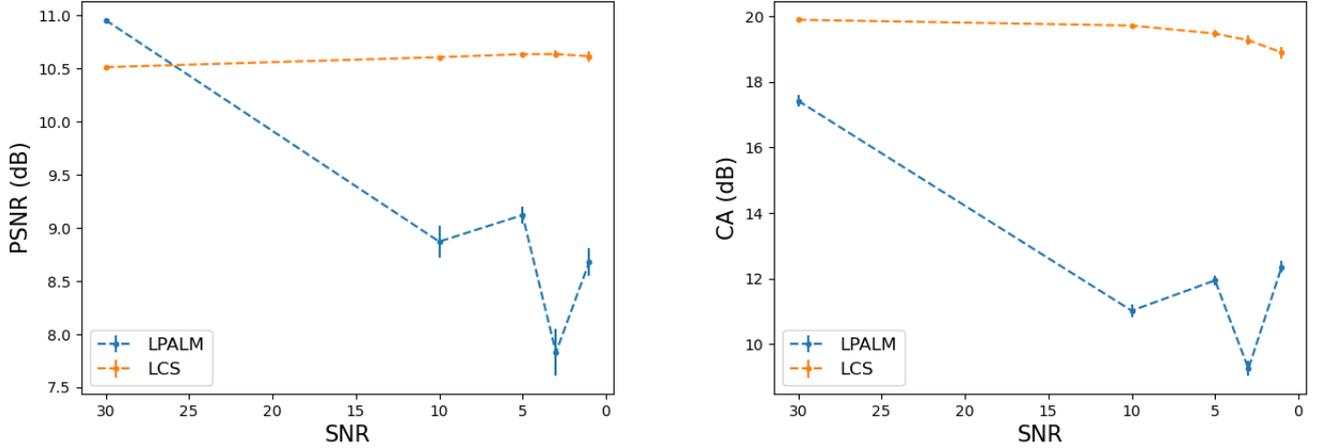
---

[3] https://github.com/mfahes/LPALM.git

Fig. 4. Performance comparison of two BSS methods in the astrophysical case, evaluated as a function of input noise level expressed in terms of S/N: LCS (orange) and LPALM (blue). Each data point represents the mean and standard error over 150 realisations from the test set of Fahes et al. (2021). Left panel: average PSNR for the source estimates (**S**). Right panel: average CA for the mixing matrix estimates (**A**).
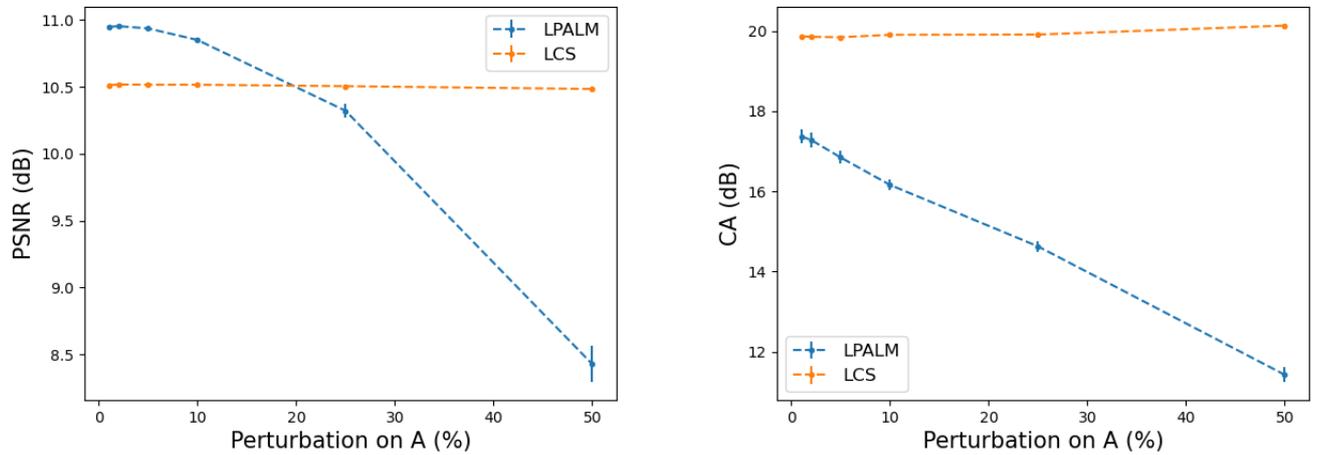

Fig. 5. Comparison of BSS method performance in the astrophysical case at a fixed S/N of 30, as a function of the percentage of perturbation applied to the mixing matrix **A**: LCS (orange) and LPALM (blue). Each point represents the mean and standard error across 150 realisations from the test set of Fahes et al. (2021). Left panel: average PSNR for the source estimates (**S**). Right panel: average CA for the mixing matrix estimates (**A**).

the mixing matrix from mixed observations, assuming the mixing matrix is known during training. The dataset consists of four source components in **S** and 64 channels in the mixing matrix (**A**). We trained LPALM across various noise levels, with S/N values ranging from 30 down to 1, and evaluate both LCS and LPALM across all noise realisations and mixing matrices from their test set, which includes 150 instances.

The results are presented in Fig. 4, with LCS shown in orange and LPALM in blue for comparison. PSNR and CA are plotted against the noise level, expressed as S/N. At a high S/N of 30, corresponding to very low noise, LPALM achieves slightly better performance, with about 0.5 dB higher PSNR – likely due to its image filtering strategy in the final iteration. However, as S/N decreases (i.e. noise increases), LPALM's performance degrades noticeably, while LCS remains largely unaffected by the noise level in this setting, likely thanks to the high number of channels (64) available for source separation.

The LCS consistently provides more accurate estimates of the mixing matrix **A**, which is particularly notable given that LCS operates in a fully blind manner – unlike LPALM, which is semi-blind and explicitly trained for this type of data. In this

experiment, the learnlets used by LCS were not retrained and remained those originally computed from ImageNet. This highlights a key advantage of LCS: it performs well even with little or no task-specific training data. When training data are available for component-specific training with individual bases $\mathcal{L}_i$, the results can be further enhanced, as demonstrated in the following experiments.

In this experiment, we also introduced a small perturbation to the mixing matrix (**A**) during testing at S/N = 30 (corresponding to a low noise level in **Y**) to illustrate the limitations of LPALM as a semi-supervised method and compare its performance with LCS. Since LPALM is trained to recover mixing matrices similar to those in its training set, we perturbed **A** as follows: $\widetilde{\mathbf{A}} = \mathbf{A} + k \times \mathcal{N}(0,1) \times \mathbf{A}$, where k varies from 0.1 to 0.5 and $\mathcal{N}(0,1)$ is a normal distribution with zero mean and unit variance. The resulting PSNR and CA values are shown in Fig. 5. As expected, LPALM's performance begins to degrade significantly once the perturbation of **A** reaches about 20%, impacting both the PSNR and CA. In contrast, LCS – being a fully blind method that does not depend on supervision – remains unaffected by these perturbations. This test further underscores the robustness
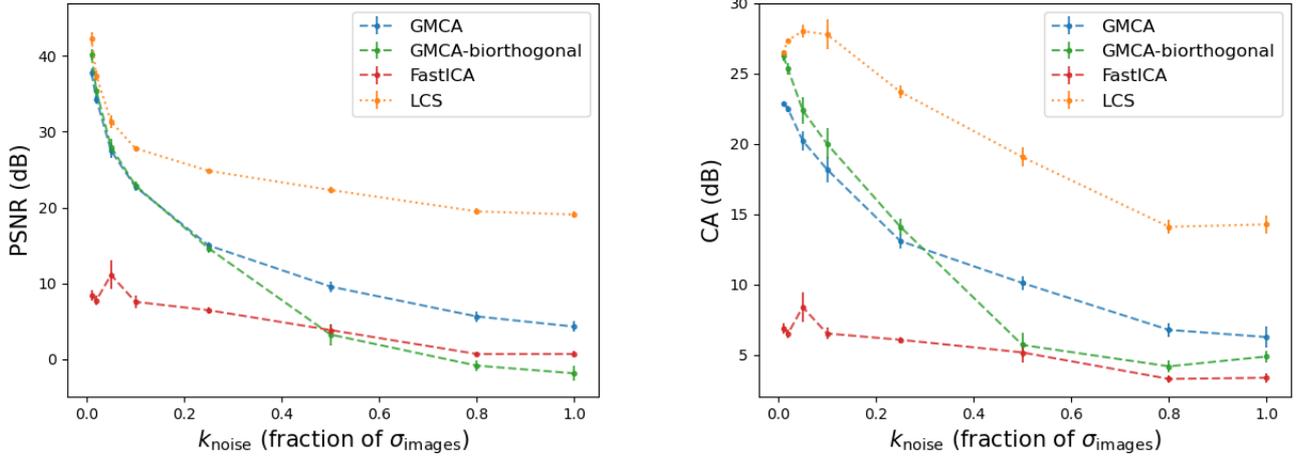
Fig. 6. Same as Fig. 3 but for the multiple-source texture case. LCS is shown in orange, GMCA in blue, GMCA-biorthogonal in green, and FastICA in red.

of LCS and highlights the advantage of a fully unsupervised approach: when the mixing matrix deviates even slightly from the training data, an unsupervised algorithm like LCS maintains stable performance without degradation.

4.2. Experiments using individually retrained bases ($\mathcal{L}_i$)

In the following experiments, instead of using learnlets trained on ImageNet, we performed individual training for each source type in **S** to better optimise the induced sparsity according to the distinct morphologies and features of each component.

4.2.1. Experiments with standard test images

In this experiment, we trained a separate learnlet network for each source to be estimated, resulting in individual bases ($\mathcal{L}_i$). For this toy model, we used the describable texture dataset (DTD) from Cimpoi et al. (2014), which is publicly available on their website[4]. The dataset includes 47 texture categories (such as banded, dotted, spiralled, etc.), each containing 120 images. Here, we focused specifically on the banded and dotted texture classes.

Both networks, $\mathcal{L}_{\text{banded}}$ and $\mathcal{L}_{\text{dotted}}$, are initialised with the ImageNet-pretrained weights from the previous section and then fine-tuned independently. For each texture class, we used the first 119 images, presenting each image ten times with different noise realisations each time.

We defined the target sources (**S**) as the two remaining (120th) images from the banded and dotted classes, which were not used during training. The subsequent steps follow the same procedure as in our experiments with the standard test images: for each noise level k_{noise}, we generated ten realisations of both the mixing matrix (**A**) and the noise (**N**), and then benchmark LCS against GMCA, GMCA-biorthogonal, and FastICA.

The resulting PSNR and CA are displayed in Fig. 6, confirming the same trend observed in Sect. 4.1.1: LCS consistently outperforms the other methods. The CA values are significantly higher across all noise levels, indicating improved source separation. However, this improvement does not fully translate to better **S** estimates here, likely due to the de-noising capability of the learnlet network's final step, which plateaus around 20 dB at high noise levels, as illustrated in Fig. 2 showing the

de-noising performance. In summary, optimising the sparsity basis and threshold parameters individually for each source type – considering their specific morphologies, shapes, and features – further enhances the separation performance compared to using a common, global basis $\mathcal{L}$.

4.2.2. Experiments with CMB and SZ data

In this fourth experiment, we applied LCS to extract the CMB and SZ components from CMB data. Here, the weights for the CMB and SZ are theoretically known: the CMB weights were set to one across all frequencies in units of μK_{CMB}, while the SZ weights followed the frequency-dependent function in the same units:

$$f(x) = x \cdot \frac{e^x + 1}{e^x - 1} - 4, \tag{8}$$

where $x = \frac{h\nu}{k_B T_{\text{CMB}}}$. Specifically, we chose the weights pre-computed from the WebSky simulation presented in Table 2 in Stein et al. (2020).

The challenge in this case arises from the presence of numerous other components with unknown weights that can contaminate the signal, the most significant being the cosmic infrared background (Dole et al. 2006), which is known to correlate with the SZ effect and contaminate SZ maps (Hurier et al. 2013; Planck Collaboration XXIII 2016). We then chose to include only the CMB, SZ, and cosmic infrared background components in our experiments, using numerical simulations from WebSky (Stein et al. 2020) available online[5]. Frequency maps were constructed from five relevant channels from the *Planck* high frequency instrument (HFI): 100, 143, 217, 353, and 545 GHz. No beam effects or additional foreground contaminants were included, and white Gaussian noise was added with standard deviations estimated from the corresponding observed *Planck* frequency maps.

Before applying LCS, we trained individual learnlet networks – similar to the approach described in the previous section – specifically designed to promote the sparsity of each expected component, denoted as $\mathcal{L}_{\text{CMB}}$, $\mathcal{L}_{\text{SZ}}$, and $\mathcal{L}_{\text{CIB}}$. To construct the training data, we divided the full-sky HEALPix maps of each component (at $N_{\text{side}} = 2048$) into 3092 overlapping square patches of 256×256 pixels, corresponding to an angular size

---

[4] https://www.robots.ox.ac.uk/~vgg/data/dtd/index.html

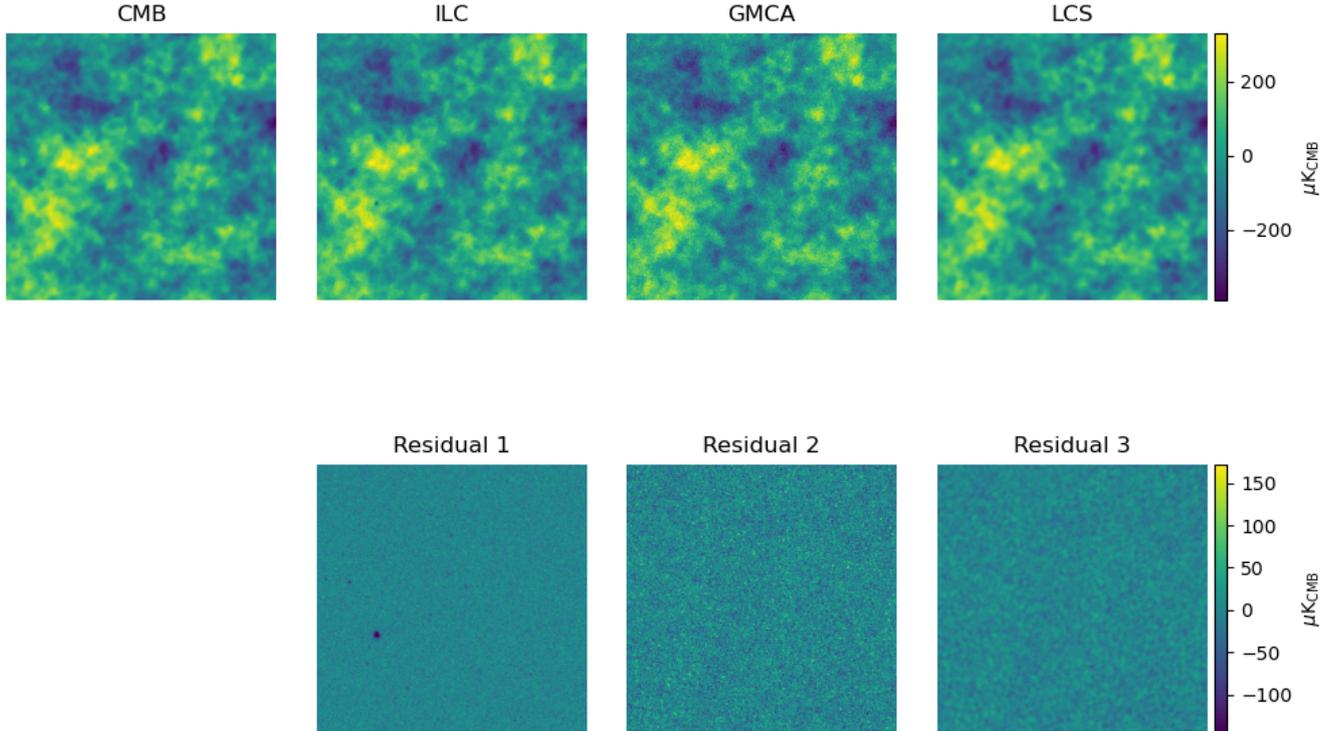[5] https://mocks.cita.utoronto.ca/data/websky/v0.0/

Fig. 7. Results of different methods applied to a patch of the CMB. Top panel: reconstructed CMB patches. Bottom panel: residuals between the methods and the ground truth.

of $\sim 7.2° \times 7.2°$ per patch. These patches cover the entire sky. We identified the patch with the highest SZ signal as the 'target patch' for evaluation by selecting the patch with the highest SZ amplitude. This patch was excluded from the training process to avoid any overlap between training and testing data and the patch containing the highest SZ source signal is shown in Fig. 8. The learnlet networks were then trained on the remaining 3,091 patches (hereafter referred to as the training set) for each component separately. The networks were trained in a supervised manner using the known clean component maps as ground truth. After training, we applied the LCS algorithm - using the trained $\mathcal{L}_{CMB}$, $\mathcal{L}_{SZ}$, and $\mathcal{L}_{CIB}$ transforms - to the noisy frequency maps, restricted to the excluded target patch. This patch was used exclusively for evaluation and comparison with other methods: GMCA, ILC, and constrained ILC. The corresponding original CMB and SZ maps are shown in the left panels of Figs. 7 and 8.

Table 1 reports the PSNR obtained for all methods; we highlight in bold the best PSNR among the different methods on this patch. The corresponding reconstructed maps are shown in Figs. 7 and 8. From the PSNR values in the patch, for the CMB case, GMCA appears to underperform relative to both ILC and LCS, achieving a PSNR of 26.28 dB. LCS delivers the best results, outperforming ILC by approximately 2 dB (35.65 dB for LCS vs 32.88 dB for ILC). Visually, the residual maps reveal that LCS effectively removes SZ contamination, visible as 'holes' in the recovered CMB maps. Regarding SZ, from PSNR values in the patch, ILC and GMCA yield comparable PSNR values of 27.90 dB and 25.56 dB, respectively, with ILC performing slightly better. In contrast, LCS achieves a substantially higher PSNR of 42.94 dB, primarily due to effective noise suppression by the learnlet de-noiser $\mathcal{L}_{SZ}$ applied as a post-processing step of LCS, as illustrated in Fig. 8. Overall, visually and from the PSNR, both the CMB and SZ maps are very well reconstructed using LCS.

Table 1. Results of different methods applied to CMB data.

Method	CMB	SZ
ILC	32.88	27.90
GMCA	26.28	25.56
LCS	**34.65**	**42.94**

However, unlike ILC – which is constructed to ensure that the residual map is (ideally) uncorrelated with the true signal – LCS does not impose such a constraint. As a consequence, some correlation between the residual and the true thermal SZ signal can remain, as visible in Fig. 8 (bottom right). Although LCS provides a visually and quantitatively accurate reconstruction on this patch, this implies that additional work would be required to guarantee unbiased residuals for applications relying on external cross-correlations. This is a natural consequence of the design of LCS, which relies on a sparsity-driven prior instead of enforcing the unbiasedness constraint built into ILC. For cross-correlation analyses, where preserving all of the signal in the cleaned map is essential, an explicit de-biasing step or constraint would need to be incorporated; we leave this for future work.

5. Discussion and conclusion

In this paper we have introduced the LCS, a new BSS framework that effectively integrates traditional sparsity-enforcing methods with the adaptability and expressivity of deep learning architectures. Utilising the learnlet transform – a learned, wavelet-inspired multi-scale representation – LCS incorporates a data-driven prior within an iterative source separation
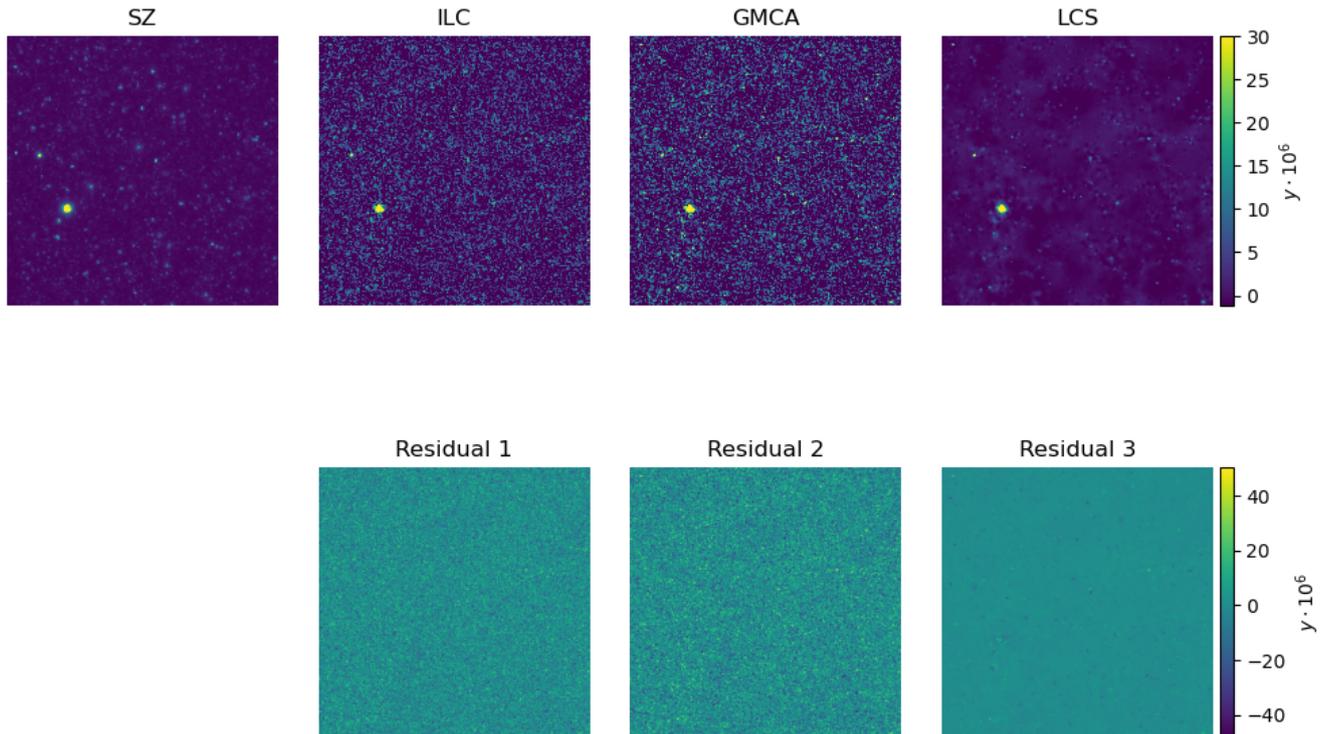
Fig. 8. Same as Fig. 7 but for the SZ patch.

procedure, delivering enhanced performance in complex astrophysical applications.

Our toy model numerical experiments show, quantified by PSNR values of extracted components compared to ground truth, that LCS generally outperforms traditional BSS methods like ICA, GMCA, and ILC, especially when dealing with non-Gaussian sources, complex spatial patterns, or varying noise levels, and especially high noise levels – showing the robustness of learnlets to noise. The data-adaptive learnlet transform allows the model to effectively capture the inherent structure of the sources, while its structured design provides both interpretability and computational efficiency – key advantages for handling the large-scale datasets anticipated from future projects such as SKA. Although this proof-of-concept study has been made by quantifying the PSNR value only, we plan to study in more detail contaminations and angular power spectra of the extracted components in future work (Gkogkou et al., in prep.).

Several promising directions remain for future work. First, expanding the LCS framework to address non-linear or convolutive mixing models (Jiang et al. 2017) could extend its applicability to more realistic physical scenarios. For instance, simultaneously performing deconvolution and component separation will be essential to account for beam effects and evolving structures in radio experiments conducted at, for example, the SKA or MeerKAT, enabling the accurate extraction of the HI signal (Gkogkou et al. 2026). Similarly, this will be critical for achieving the highest possible resolution in CMB or SZ maps from upcoming facilities such as LiteBird or the Simons Observatory. Second, incorporating uncertainty quantification techniques could provide valuable measures of confidence in the separated components, which is vital for rigorous scientific interpretation. Finally, embedding domain-specific physical models or incorporating known spectral priors within the learning process may further improve both the robustness and interpretability of the results.

Beyond astrophysics, the methods developed for the LCS have wide applicability across various signal processing fields where BSS is essential. This includes areas like biomedical imaging, remote sensing, audio band un-mixing, and any discipline that requires interpretable and adaptable source separation techniques.

In conclusion, LCS paves the way for employing learnlets in BSS, providing a flexible, interpretable, and high-performing approach for complex signal decomposition challenges. While its hybrid architecture is especially well suited for astrophysical applications, the underlying concepts readily extend to a broad range of modern signal processing domains.

References

Akhaury, U., Starck, J.-L., Jablonka, P., Courbin, F., & Michalewicz, K. 2022, Front. Astron. Space Sci., 9, 357

Alonso, D., Bull, P., Ferreira, P. G., & Santos, M. G. 2015, MNRAS, 447, 400

Beck, A., & Teboulle, M. 2009, SIAM J. Imaging Sci., 2, 183

Bobin, J., Starck, J.-L., Fadili, J., & Moudden, Y. 2007, IEEE Trans. Image Process., 16, 2662

Bobin, J., Moudden, Y., Starck, J. L., Fadili, J., & Aghanim, N. 2008, Stat. Methodol., 5, 307

Bobin, J., Sureau, F., Starck, J. L., Rassat, A., & Paykari, P. 2014, A&A, 563, A105

Bonjean, V., Tanimura, H., Aghanim, N., Bonnaire, T., & Douspis, M. 2024, A&A, 686, A91

Bull, P., Ferreira, P. G., Patel, P., & Santos, M. G. 2015, ApJ, 803, 21

Carucci, I. P., Bernal, J. L., Cunnington, S., et al. 2025, A&A, 703, A222

Carucci, I. P., Irfan, M. O., & Bobin, J. 2020, MNRAS, 499, 304

Cimpoi, M., Maji, S., Kokkinos, I., et al. 2014, in Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)

Delabrouille, J., Cardoso, J. F., & Patanchon, G. 2003, MNRAS, 346, 1089

Delabrouille, J., Cardoso, J. F., Le Jeune, M., et al. 2009, A&A, 493, 835

Dole, H., Lagache, G., Puget, J. L., et al. 2006, A&A, 451, 417

Fahes, M., Kervazo, C., Bobin, J., & Tupin, F. 2021, arXiv e-prints [arXiv:2112.05694]

Gkogkou, A., Bonjean, V., Starck, J.-L., Spinelli, M., & Tsakalides, P. 2026, A&A, in press, https://doi.org/10.1051/0004-6361/202557229

Gregor, K., & LeCun, Y. 2010, in Proceedings of the 27th International Conference on International Conference on Machine Learning, ICML'10 (Madison, WI, USA: Omnipress), 399

Hurier, G., Macías-Pérez, J. F., & Hildebrandt, S. 2013, A&A, 558, A118

Hyvärinen, A., & Oja, E. 2000, Neural Netw., 13, 411

Jiang, M., Bobin, J., & Starck, J.-L. 2017, arXiv e-prints [arXiv:1703.02650]

Makinen, T. L., Lancaster, L., Villaescusa-Navarro, F., et al. 2021, J. Cosmology Astropart. Phys., 2021, 081

Matshawule, S. D., Spinelli, M., Santos, M. G., & Ngobese, S. 2021, MNRAS, 506, 5075

Maćkiewicz, A., & Ratajczak, W. 1993, Comp. Geosci., 19, 303

Mertens, F. G., Bobin, J., & Carucci, I. P. 2024, MNRAS, 527, 3517

Planck Collaboration XXIII. 2016, A&A, 594, A23

Planck HFI Core Team 2011, A&A, 536, A4

Ramzi, Z., Starck, J.-L., Moreau, T., & Ciuciu, P. 2021, in EUSIPCO 2020 - 28th European Signal Processing Conference, Amsterdam, The Netherlands

Ronneberger, O., Fischer, P., & Brox, T. 2015, arXiv e-prints [arXiv:1505.04597]

Russakovsky, O., Deng, J., Su, H., et al. 2015, Int. J. Comp. Vision (IJCV), 115, 211

Santos, M., Bull, P., Alonso, D., et al. 2015, in Advancing Astrophysics with the Square Kilometre Array (AASKA14), 19

Spinelli, M., Carucci, I. P., Cunnington, S., et al. 2022, MNRAS, 509, 2048

Starck, J., Murtagh, F., & Fadili, J. 2015, Sparse Image and Signal Processing: Wavelets and Related Geometric Multiscale Analysis, 2nd edn. (USA: Cambridge University Press)

Stein, G., Alvarez, M. A., Bond, J. R., van Engelen, A., & Battaglia, N. 2020, J. Cosmology Astropart. Phys., 2020, 012

Weltman, A., Bull, P., Camera, S., et al. 2020, PASA, 37, e002

Xu, X., Li, J., Li, S., & Plaza, A. 2020, IEEE Trans. Geosci. Remote Sens., 58, 2817

Yong, X., Ward, R. K., & Birch, G. E. 2009, in 2009 4th International IEEE/EMBS Conference on Neural Engineering, 343

Appendix A: Interpreting the learnlets

The learnlet network trained on the ImageNet dataset, as described in Sect. 2.2, exhibits interpretability. By examining the shapes of the learned filters – referred to as learnlets – we can identify key features and patterns that contribute to the sparse representation of the images. Figure A.1 displays the learnlets at the first scale, revealing various structures captured during training, such as horizontal, vertical, and diagonal orientations, along with cross-like and small circular patterns.

The threshold parameters k_j are also of particular interest. The values learned during training on the ImageNet dataset, as presented in Sect. 2.2, are shown in Fig. A.2. Notably, these thresholds exhibit a decreasing trend, with higher values at finer scales and lower values at coarser scales. This pattern aligns with expectations, as Gaussian noise tends to be more prominent at fine scales and progressively diminishes at coarser levels, often becoming negligible at the coarsest scale. The network effectively captures this behaviour during training through its learned thresholding mechanism.
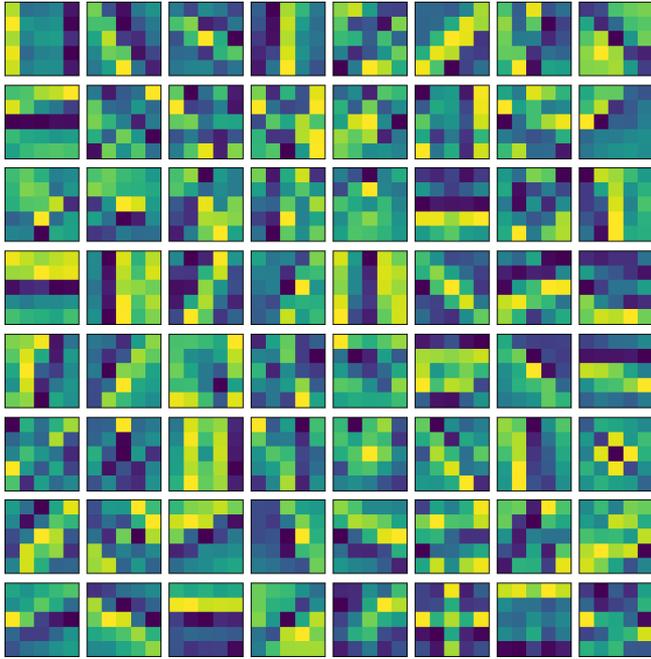

Fig. A.2. Threshold parameters (k_j) learned during the training of the learnlet network on the ImageNet dataset in Sect. 2.2.

Fig. A.1. Example of learnlets from the first scale of the learnlet network trained on the ImageNet dataset in Sect. 2.2. Distinct horizontal, vertical, and diagonal st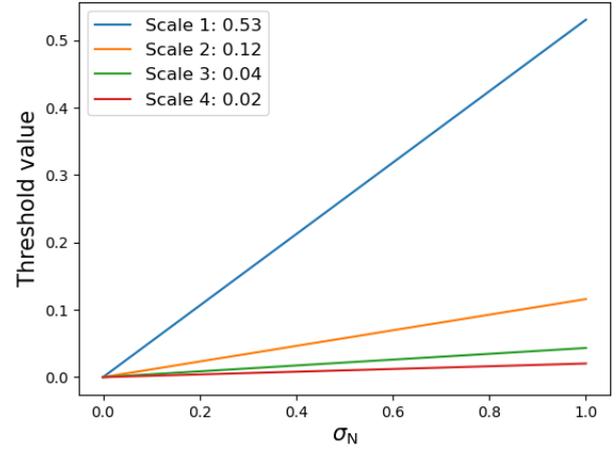ructures are clearly identifiable.