Monthly Notices of the ROYAL ASTRONOMICAL SOCIETY

MNRAS **462,** 4314–4335 (2016) doi:10.1093/mnras/stw1859
Advance Access publication 2016 August 1

Scalable splitting algorithms for big-data interferometric imaging in the SKA era

Alexandru Onose,[1★†] Rafael E. Carrillo,[2†] Audrey Repetti,[1] Jason D. McEwen,[3] Jean-Philippe Thiran,[2] Jean-Christophe Pesquet[4] and Yves Wiaux[1]

[1]*Institute of Sensors, Signals and Systems, Heriot-Watt University, Edinburgh EH14 4AS, UK*
[2]*Signal Processing Laboratory (LTS5), Ecole Polytechnique Fédérale de Lausanne, Lausanne CH-1015, Switzerland*
[3]*Mullard Space Science Laboratory, University College London, Surrey RH5 6NT, UK*
[4]*Laboratoire d'Informatique Gaspard Monge, Université Paris-Est, Marne la Vallée F-77454, France*

Accepted 2016 July 25. Received 2016 July 22; in original form 2016 January 15

ABSTRACT

In the context of next-generation radio telescopes, like the Square Kilometre Array (SKA), the efficient processing of large-scale data sets is extremely important. Convex optimization tasks under the compressive sensing framework have recently emerged and provide both enhanced image reconstruction quality and scalability to increasingly larger data sets. We focus herein mainly on scalability and propose two new convex optimization algorithmic structures able to solve the convex optimization tasks arising in radio-interferometric imaging. They rely on proximal splitting and forward-backward iterations and can be seen, by analogy, with the CLEAN major-minor cycle, as running sophisticated CLEAN-like iterations in parallel in multiple data, prior, and image spaces. Both methods support any convex regularization function, in particular, the well-studied ℓ_1 priors promoting image sparsity in an adequate domain. Tailored for big-data, they employ parallel and distributed computations to achieve scalability, in terms of memory and computational requirements. One of them also exploits randomization, over data blocks at each iteration, offering further flexibility. We present simulation results showing the feasibility of the proposed methods as well as their advantages compared to state-of-the-art algorithmic solvers. Our MATLAB code is available online on GitHub.

Key words: techniques: image processing – techniques: interferometric.

1 INTRODUCTION

Radio-interferometry (RI) allows the observation of radio emissions with great sensitivity and angular resolution. The technique has been extensively investigated and provides valuable data driving many research directions in astronomy, cosmology or astrophysics (Thompson, Moran & Swenson 2001). Next-generation radio telescopes, such as the LOw Frequency ARray (LOFAR; van Haarlem et al. 2013) and the future Square Kilometre Array (SKA; Dewdney et al. 2009), are envisaged to produce giga-pixel images and achieve a dynamic range of six or seven orders of magnitude. This will be an improvement over current instruments by around two orders of magnitude, in terms of both resolution and sensitivity. The amount of data acquired will be massive and the methods solving the inverse problems associated with the image reconstruction need to be fast and to scale well with the number of measurements. Such challenges provided motivation for vigorous research to reformulate imaging and calibration techniques for RI (Wijnholds et al. 2014).

The construction of the first phase of SKA is scheduled to start in 2018. It will consist of two subsystems: a low-frequency aperture array, the SKA1-low, operating in the 50–350 MHz frequency range and containing approximately 131 000 antenna elements; a mid-frequency array of reflector dishes, the SKA1-mid, operating above 350 MHz, consisting of 197 dishes (Dewdney et al. 2009; Broekema, van Nieuwpoort & Bal 2015). Both subsystems are planned to operate on the order of 65 000 frequency bands. Data rate estimates in this first phase are around five terabits per second for each subsystem (Broekema et al. 2015) and will present a great challenge for the infrastructure and signal processing. The celebrated CLEAN algorithm (Högbom 1974) and its variants do not scale well given the large dimension of the problem. They rely on local greedy iterative procedures and are slow compared to modern convex optimization techniques, which are guaranteed to converge towards a global optimal solution. Moreover, they are not designed for large-scale parallelization or distributed computing (Carrillo, McEwen & Wiaux 2014).

In the past few years, sparse models and convex optimization techniques have been applied to RI imaging, showing the potential

★ E-mail: a.onose@hw.ac.uk
† The authors have contributed equally to the work herein.

to outperform state-of-the-art imaging algorithms in the field (Rau et al. 2009; Wiaux et al. 2009a; Li, Cornwell & de Hoog 2011; Carrillo, McEwen & Wiaux 2012; Carrillo et al. 2013; Carrillo et al. 2014; Garsden et al. 2015). These methods typically solve the imaging problem by minimizing an objective function defined as a sum of a data term, dependent on the measured visibilities, and several regularization terms, usually promoting sparsity and positivity. Scalable algorithms, specifically tailored for large-scale problems using parallel and distributed schemes, are just now beginning to gain attention in the context of imaging (Carrillo et al. 2014; Ferrari et al. 2014) and calibration (Yatawatta 2015) for next-generation radio telescopes.

In this context, proximal splitting methods are very popular due to their ability to decompose the original problem into several simpler, easier to solve, sub-problems, each one associated with one term of the objective function (Combettes & Pesquet 2011). Another class of algorithms currently gaining traction for large-scale problems in optimization is based on primal-dual (PD) methods (Komodakis & Pesquet 2015). Such methods efficiently split the optimization problem and, at the same time, maintain a highly parallelizable structure by solving concomitantly for a dual formulation of the original problem. Building on such tools, the simultaneous direction method of multipliers (SDMM) was recently proposed in the context of RI imaging by Carrillo et al. (2014). It achieves the complete splitting of the functions defining the minimization task. In the big-data context, SDMM scales well with the number of measurements, however, an expensive matrix inversion is necessary when updating the solution, which limits the suitability of the method for the recovery of very large images.

The scope of this article is to propose two new algorithmic structures for RI imaging. We study their computational performance and parallelization capabilities by solving the sparsity averaging optimization problem proposed in the sparsity averaging reweighted analysis (SARA) algorithm (Carrillo et al. 2012), previously shown to outperform the standard CLEAN methods. The application of the two algorithms is not limited to the SARA prior, any other convex prior functions being supported. We assume a known model for the measured data such that there is no need for calibration. We use SDMM, solving the same minimization problem, to compare the computational burden and parallelization possibilities. Theoretical results ensure convergence, i.e. all algorithms reach the same solution. We also showcase the reconstruction performance of the two algorithms coupled with the SARA prior in comparison with CS-CLEAN (Schwab 1984) and MORESANE (Dabbech et al. 2015).

The first algorithmic solver is a sub-iterative version of the well-known alternating direction method of multipliers (ADMM). The second is based on the PD method and uses forward-backward (FB) iterations, typically alternating between gradient (forward) steps and projection (backward) steps. Such steps can be seen as interlaced CLEAN-like updates. Both algorithms are highly parallelizable and allow for an efficient distributed implementation. ADMM, however, offers only partial splitting of the objective function leading to a sub-iterative algorithmic structure. The PD method offers the full splitting for both operators and functions. It does not need sub-iterations or any matrix inversion. Additionally, it can attain increased scalability by using randomized updates. It works by selecting only a fraction of the visibilities at each iteration, thus achieving great flexibility in terms of memory requirements and computational load per iteration, at the cost of requiring more iterations to converge. Our simulations suggest no significant increase in the total computation cost.

The remainder of this article is organized as follows. Section 2 introduces the RI imaging problem and describes the state-of-the-art image reconstruction techniques used in radio astronomy. In Section 3, we review some of the main tools from convex optimization needed for RI imaging. Section 4 formulates the optimization problem for RI imaging, given the large-scale data scenario and presents the proposed algorithms, ADMM and PD, respectively. We discuss implementation details and their computational complexity in Section 5. Numerical experiments evaluating the performance of the algorithms are reported in Section 6. Finally, we briefly present the main contributions and envisaged future research directions in Section 7.

2 RADIO-INTERFEROMETRIC IMAGING

Radio-interferometric data, the visibilities, are produced by an array of antenna pairs that measure radio emissions from a given area of the sky. The projected baseline components, in units of the wavelength of observation, are commonly denoted (u, v, w), where w identifies the component in the line of sight and $\boldsymbol{u} = (u, v)$ the components in the orthogonal plane. The sky brightness distribution x is described in the same coordinate system, with components l, m, n and with $\boldsymbol{l} = (l, m)$ and $n(\boldsymbol{l}) = \sqrt{1 - l^2 - m^2},\ l^2 + m^2 \leq 1$. The general measurement equation for non-polarized monochromatic RI imaging can be stated as

$$y(\boldsymbol{u}) = \int D(\boldsymbol{l}, \boldsymbol{u}) x(\boldsymbol{l}) e^{-2i\pi \boldsymbol{u}\cdot\boldsymbol{l}} \mathrm{d}^2\boldsymbol{l}, \tag{1}$$

with $D(\boldsymbol{l}, \boldsymbol{u}) = \frac{1}{n(\boldsymbol{l})} \bar{D}(\boldsymbol{l}, \boldsymbol{u})$ quantifying all the direction dependent effects (DDEs). Some dominant DDEs can be modelled analytically, like the w component which is expressed as $\bar{D}_w(\boldsymbol{l}, \boldsymbol{u}) = e^{-2i\pi w(n(\boldsymbol{l})-1)}$. At high dynamic ranges, however, unknown DDEs, related to the primary beam or ionospheric effects, also affect the measurements introducing the need for calibration. Here we work in the absence of DDEs.

The recovery of x from the visibilities relies on algorithms solving a discretized version of the inverse problem (1). We denote by $\boldsymbol{x} \in \mathbb{R}^N$ the intensity image of which we take M visibility measurements $\boldsymbol{y} \in \mathbb{C}^M$. The measurement model is defined by

$$\boldsymbol{y} = \boldsymbol{\Phi x} + \boldsymbol{n}, \tag{2}$$

where the measurement operator $\boldsymbol{\Phi} \in \mathbb{C}^{M\times N}$ is a linear map from the image domain to the visibility space and $\boldsymbol{y}$ denotes the vector of measured visibilities corrupted by the additive noise $\boldsymbol{n}$. Due to limitations in the visibility sampling scheme, equation (2) defines an ill-posed inverse problem. Furthermore, the large number of the data points, $M \gg N$, introduces additional challenges related to the computational and memory requirements for finding the solution. In what follows, we assume the operator $\boldsymbol{\Phi}$ to be known is advance such that no calibration step is needed to estimate it.

Due to the highly iterative nature of the reconstruction algorithms, a fast implementation of all operators involved in the image reconstruction is essential, for both regularization and data terms. To this purpose, the measurement operator is modelled as the product between a matrix $\mathbf{G} \in \mathbb{C}^{M\times n_{\mathrm{o}}N}$ and an n_{o}-oversampled Fourier operator,

$$\boldsymbol{\Phi} = \mathbf{GFZ}. \tag{3}$$

The matrix $\mathbf{Z} \in \mathbb{R}^{n_{\mathrm{o}}N\times N}$ accounts for the oversampling and the scaling of the image to pre-compensate for possible imperfections in the interpolation (Fessler & Sutton 2003). In the absence of DDEs,

G only contains compact support kernels that enable the computation of the continuous Fourier samples from the discrete Fourier coefficients provided by **F**. Alternatively, seen as a transform from the u–v space to the discrete Fourier space, $\mathbf{G}^\dagger$, the adjoint operator of **G**, grids the continuous measurements on to a uniformly sampled Fourier space associated with the oversampled discrete Fourier coefficients provided by **FZ**. This representation of the measurement operator enables a fast implementation, thanks to the use of the fast Fourier transform for **F** and to the fact that the convolution kernels used are, in general, modelled with compact support in the Fourier domain, which leads to a sparse matrix **G**.[1]

2.1 Classical imaging algorithms

Various methods have been proposed for solving the inverse problem defined by (2). The standard imaging algorithms belong to the CLEAN family and perform a greedy non-linear deconvolution based on local iterative beam removal (Högbom 1974; Schwarz 1978; Schwab 1984; Thompson et al. 2001). A sparsity prior on the solution is implicitly introduced since the method reconstructs the image pixel by pixel. Thus, CLEAN is very similar to the matching pursuit (MP) algorithm (Mallat & Zhang 1993). It may also be seen as a regularized gradient descent method. It minimizes the residual norm $\|\boldsymbol{y} - \boldsymbol{\Phi x}\|_2^2$ via a gradient descent subject to an implicit sparsity constraint on $\boldsymbol{x}$. An update of the solution takes the following form

$$\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)} + \mathcal{T}\left(\boldsymbol{\Phi}^\dagger\left(\boldsymbol{y} - \boldsymbol{\Phi x}^{(t-1)}\right)\right), \tag{4}$$

where $\boldsymbol{\Phi}^\dagger$ is the adjoint of the linear operator $\boldsymbol{\Phi}$. In the astronomy community, the computation of the residual image $\boldsymbol{\Phi}^\dagger(\boldsymbol{y} - \boldsymbol{\Phi x}^{(t-1)})$, which represents a gradient step of the residual norm, is being referred to as the major cycle, while the deconvolution performed by the operator $\mathcal{T}$ is named the minor cycle. All proposed versions of CLEAN use variations of these major and minor cycles (Rau et al. 2009). CLEAN builds the solution image iteratively by searching for atoms associated with the largest magnitude pixel from the residual image. A loop gain factor controls how aggressive is the update step, by only allowing a fraction of the chosen atoms to be used.

Multiple improvements of CLEAN have been suggested. In the multiscale version (Cornwell 2008), the sparsity model is augmented through a multiscale decomposition. An adaptive scale variant was proposed by Bhatnagar & Cornwell (2004) and can be seen as MP with overcomplete dictionaries since it models the image as a superposition of atoms over a redundant dictionary. Another class of solvers, the maximum entropy method (Ables 1974; Gull & Daniell 1978; Cornwell & Evans 1985), solves a regularized global optimization problem through a general entropy prior. In practice, however, CLEAN and its variants have been preferred even though they are slow and require empirically chosen configuration parameters. Furthermore, these methods also lack the scalability required for working with huge, SKA-like data.

[1] Assuming pre-calibrated data in the presence of DDEs, the line of **G** associated with frequency $\boldsymbol{u}$, is explicitly given by the convolution of the discrete Fourier transform of $D(\boldsymbol{l}, \boldsymbol{u})$, centred on $\boldsymbol{u}$, with the associated gridding kernel. This maintains the sparse structure of **G**, since the DDEs are generally modelled with compact support in the Fourier domain. A non-sparse **G** drastically increases the computational requirements for the implementation of the measurement operator. However, it is generally transparent to the algorithms since they do not rely on the sparsity structure explicitly. This is the case for all the algorithmic structures discussed herein.

2.2 Compressed sensing in RI

Imaging algorithms based on convex optimization and using sparsity-aware models have also been proposed, especially under the theoretical framework of compressed sensing (CS), reporting superior reconstruction quality with respect to CLEAN and its multiscale versions. CS proposes both the optimization of the acquisition framework, going beyond the traditional Nyquist sampling paradigm, and the use of non-linear iterative algorithms for signal reconstruction, regularizing the ill-posed inverse problem through a low dimensional signal model (Candès 2006; Donoho 2006). The key premise in CS is that the underlying signal has a sparse representation, $\boldsymbol{x} = \boldsymbol{\Psi\alpha}$ with $\boldsymbol{\alpha} \in \mathbb{C}^D$ containing only a few non-zero elements (Fornasier & Rauhut 2011), in a dictionary $\boldsymbol{\Psi} \in \mathbb{C}^{N\times D}$, e.g. a collection of wavelet bases or, more generally, an overcomplete frame.

The first study of CS applied to RI was done by Wiaux et al. (2009a), who demonstrated the versatility of convex optimization methods and their superiority relative to standard interferometric imaging techniques. A CS approach was developed by Wiaux, Puy & Vandergheynst (2010) to recover the signal induced by cosmic strings in the cosmic microwave background. McEwen & Wiaux (2011) generalized the CS imaging techniques to wide field-of-view observations. Non-coplanar effects and the optimization of the acquisition process, were studied by Wiaux et al. (2009b) and Wolz et al. (2013). All the aforementioned works solve a synthesis-based problem defined by

$$\min_{\boldsymbol{\alpha}} \|\boldsymbol{\alpha}\|_1 \quad \text{subject to} \quad \|\boldsymbol{y} - \boldsymbol{\Phi\Psi\alpha}\|_2 \le \epsilon, \tag{5}$$

where ϵ is a bound on the ℓ_2 norm of the noise $\boldsymbol{n}$. Synthesis-based problems recover the image representation $\boldsymbol{\alpha}$ with the final image obtained from the synthesis relation $\boldsymbol{x} = \boldsymbol{\Psi\alpha}$. Here, the best model for the sparsity, the non-convex ℓ_0 norm, is replaced with its closest convex relaxation, the ℓ_1 norm, to allow the use of efficient convex optimization solvers. Re-weighting schemes are generally employed to approximate the ℓ_0 norm from its ℓ_1 relaxation (Candès, Wakin & Boyd 2008; Daubechies et al. 2010). Imaging approaches based on unconstrained versions of (5) have also been studied (Wenger et al. 2010; Li et al. 2011; Hardy 2013; Garsden et al. 2015). For example, Garsden et al. (2015) applied a synthesis-based reconstruction method to LOFAR data.

As opposed to synthesis-based problems, analysis-based approaches recover the signal itself, solving

$$\min_{\boldsymbol{x}} \|\boldsymbol{\Psi}^\dagger \boldsymbol{x}\|_1 \quad \text{subject to} \quad \|\boldsymbol{y} - \boldsymbol{\Phi x}\|_2 \le \epsilon. \tag{6}$$

The SARA algorithm, based on the analysis approach and an average sparsity model, was introduced by Carrillo et al. (2012). Carrillo et al. (2014) proposed a scalable algorithm, based on SDMM, to solve (6). For such large-scale problems, the use of sparsity operators $\boldsymbol{\Psi}$ that allow for a fast implementation is fundamental. Hybrid analysis-by-synthesis greedy approaches have also been proposed by Dabbech et al. (2015).

To provide an analogy between CLEAN and the FB iterations employed herein, we can consider one of the most basic approaches, the unconstrained version of the minimization problem (6), namely $\min_{\boldsymbol{x}} \|\boldsymbol{\Psi}^\dagger \boldsymbol{x}\|_1 + \beta\|\boldsymbol{y} - \boldsymbol{\Phi x}\|_2^2$ with β a free parameter. To solve it, modern approaches using FB iterations perform a gradient step together with a soft-thresholding operation in the given basis $\boldsymbol{\Psi}^\dagger$ (Combettes & Pesquet 2007b). This FB iterative structure is conceptually extremely close to the major-minor cycle structure of CLEAN. At a given iteration, the forward (gradient) step consists in doing a step in the opposite direction to the gradient of the ℓ_2 norm of the residual. It is essentially equivalent to a major cycle of CLEAN. The backward (soft-thresholding) step consists in decreasing the absolute values of all the coefficients of $\boldsymbol{\Psi}^\dagger \boldsymbol{x}$ that are above a certain threshold by the threshold value, and setting to zero those below the threshold. This step is very similar to the minor cycle of CLEAN, with the soft-threshold value being an analogous to the loop gain factor. The soft-thresholding intuitively works by removing small and insignificant coefficients, globally, on all signal locations simultaneously, while CLEAN iteratively builds up the signal by picking up parts of the most important coefficients, a local procedure, until the residuals become negligible. Thus, CLEAN can be intuitively understood as a very specific version of the FB algorithm. As will be discussed in Section 4, from the perspective of CLEAN, the algorithms presented herein can be viewed as being composed of complex CLEAN-like FB steps performed in parallel in multiple data, prior and image spaces.

3 CONVEX OPTIMIZATION

Optimization techniques play a central role in solving the large-scale inverse problem (2) from RI. Some of the main methods from convex optimization (Bauschke & Combettes 2011) are presented in what follows.

Proximal splitting techniques are very attractive due to their flexibility and ability to produce scalable algorithmic structures. Examples of proximal splitting algorithms include the Douglas–Rachford method (Combettes & Pesquet 2007a; Boţ & Hendrich 2013), the projected gradient approach (Calamai & Moré 1987), the iterative thresholding algorithm (Daubechies, Defrise & De Mol 2004; Beck & Teboulle 2009), the ADMM (Boyd et al. 2011) or the SDMM (Setzer, Steidl & Teuber 2010). All proximal splitting methods solve optimization problems like

$$\min_{z} g_1(z) + \ldots + g_n(z), \tag{7}$$

with g_i, $i \in \{1, \ldots, n\}$, proper, lower semicontinuous, convex functions. No assumptions are required about the smoothness, each non-differentiable function being incorporated into the minimization through its proximity operator (C1). Constrained problems are reformulated to fit (7) through the use of the indicator function (C2) of the convex set $\mathcal{C}$ defined by the constraints. As a general framework, proximal splitting methods minimize (7) iteratively by handling each function g_i, possibly non-smooth, through its proximity operator. A good review of the main proximal splitting algorithms and some of their applications to signal and image processing is presented by Combettes & Pesquet (2011).

PD methods (Komodakis & Pesquet 2015) introduce another framework over the proximal splitting approaches and are able to achieve full splitting. All the operators involved, not only the gradient or proximity operators, but also the linear operators, can be used separately. Due to this, no inversion of operators is required, which gives important computational advantages when compared to other splitting schemes (Combettes & Pesquet 2012). The methods solve optimization tasks of the form

$$\min_{z} g_1(z) + g_2(\mathbf{L}z), \tag{8}$$

with g_1 and g_2 proper, lower semicontinuous convex functions and $\mathbf{L}$ a linear operator. They are easily extended to problems, similar to (7), involving multiple functions. The minimization (8), usually referred to as the primal problem, accepts a dual problem (Bauschke & Combettes 2011),

$$\min_{\boldsymbol{v}} g_1^*(-\mathbf{L}^\dagger \boldsymbol{v}) + g_2^*(\boldsymbol{v}), \tag{9}$$

where $\mathbf{L}^\dagger$ is the adjoint of the linear operator $\mathbf{L}$ and g_2^* is the Legendre–Fenchel conjugate function of g_2, defined in (C3). Under our assumptions for g_1 and g_2 and, if a solution to (8) exists, efficient algorithms for solving together the primal and dual problems can be devised (Combettes & Pesquet 2012; Condat 2013; Vũ 2013). Such PD approaches are able to produce highly scalable algorithms that are well suited for solving inverse problems similar to (2). They are flexible and offer a broad class of methods ranging from distributed computing to randomized or block coordinate approaches (Combettes & Pesquet 2015; Pesquet & Repetti 2015).

Augmented Lagrangian (AL) methods (Bertsekas 1982) have been traditionally used for solving constrained optimization problems through an equivalent unconstrained minimization. In our context, the methods can be applied for finding the solution to a constrained optimization task equivalent to (8),

$$\min_{z,\boldsymbol{r}} g_1(z) + g_2(\boldsymbol{r}), \quad \text{subject to } \boldsymbol{r} = \mathbf{L}z, \tag{10}$$

by the introduction of the slack variable $\boldsymbol{r}$. The solution is found by searching for a saddle point of the augmented Lagrange function associated with (10),

$$\max_{\boldsymbol{s}} \min_{z,\boldsymbol{r}} g_1(z) + g_2(\boldsymbol{r}) + \frac{\boldsymbol{s}^\dagger}{\mu}\left(\mathbf{L}z - \boldsymbol{r}\right) + \frac{1}{2\mu}\|\mathbf{L}z - \boldsymbol{r}\|_2^2. \tag{11}$$

The vector $\boldsymbol{s}$ and parameter μ, correspond to the Lagrange multipliers. No explicit assumption is required on the smoothness of the functions g_1 and g_2. Several algorithms working in this framework have been proposed. The ADMM (Boyd et al. 2011; Yang & Zhang 2011) is directly applicable to the minimization (10). A generalization of the method, solving (7), is the SDMM (Setzer et al. 2010). It finds the solution to an extended augmented Lagrangian, defined for multiple functions g_i. Both methods can also be characterized from the PD perspective (Boyd et al. 2011; Komodakis & Pesquet 2015). Algorithmically, they split the minimization step by alternating between the minimization over each of the primal variables, z and $\boldsymbol{r}$, followed by a maximization with respect to the multipliers $\boldsymbol{s}$, performed via a gradient ascent.

4 LARGE-SCALE OPTIMIZATION

The next-generation telescopes will be able to produce a huge amount of visibility data. To this regard, there is much interest in the development of fast and well performing reconstruction algorithms (McEwen & Wiaux 2011; Carrillo et al. 2014). Highly scalable algorithms, distributed or parallelized, are just now beginning to gather traction (Carrillo et al. 2014; Ferrari et al. 2014). Given their flexibility and parallelization capabilities, the PD and AL algorithmic frameworks are prime candidates for solving the inverse problems from RI.

4.1 Convex optimization algorithms for RI

Under the CS paradigm, we can redefine the inverse problem as the estimation of the image $\boldsymbol{x} \in \mathbb{R}^N$ given the measurements $\boldsymbol{y} \in \mathbb{C}^M$ under the constraint that the image is sparse in an overcomplete dictionary $\boldsymbol{\Psi}$. Since the solution of interests is an intensity image, we also require $\boldsymbol{x}$ to be real and positive. The analysis formulation (6) is more tractable since it generally produces a simpler optimization problem when overcomplete dictionaries are used (Elad, Milanfar & Rubinstein 2007). Additionally, the constrained formulation offers an easy way of defining the minimization given accurate noise estimates.

Thus, we state the reconstruction task as the convex minimization problem (Carrillo et al. 2013, 2014)

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) + l(\boldsymbol{\Psi}^\dagger \boldsymbol{x}) + h(\boldsymbol{\Phi}\boldsymbol{x}) \tag{12}$$

with the functions involved including all the aforementioned constraints,

$$\begin{aligned} &l = \|\cdot\|_1, \\ &f = \iota_{\mathcal{C}}, \qquad \mathcal{C} = \mathbb{R}_+^N, \\ &h(\boldsymbol{z}) = \iota_{\mathcal{B}}(\boldsymbol{z}), \ \mathcal{B} = \{\boldsymbol{z} \in \mathbb{C}^M : \|\boldsymbol{z} - \boldsymbol{y}\|_2 \leq \epsilon\}. \end{aligned} \tag{13}$$

The function f introduces the reality and positivity requirement for the recovered solution, l represents the sparsity prior in the given dictionary $\boldsymbol{\Psi}$ and h is the term that ensures data fidelity constraining the residual to be situated in an ℓ_2 ball defined by the noise level ϵ.

We set the operator $\boldsymbol{\Psi} \in \mathbb{C}^{N\times n_{\rm b}N}$ to be a collection of $n_{\rm b}$ sparsity inducing bases (Carrillo et al. 2014). The SARA wavelet bases (Carrillo et al. 2012) are a good candidate but problem (12) is not restricted to them. A re-weighted ℓ_1 approach (Candès et al. 2008) may also be used by implicitly imposing weights on the operator $\boldsymbol{\Psi}$ but it is not specifically dealt with herein since it does not change the algorithmic structure. This would serve to approximate the ℓ_0 pseudo-norm, $\|\boldsymbol{\Psi}^\dagger \boldsymbol{x}\|_0$, by iteratively re-solving the same problem as in (12) with refined weights based on the inverse of the solution coefficients from the previous re-weighted problem.

An efficient parallel implementation can be achieved from (2) by splitting of the data into multiple blocks

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_{n_{\rm d}} \end{bmatrix}, \qquad \boldsymbol{\Phi} = \begin{bmatrix} \boldsymbol{\Phi}_1 \\ \vdots \\ \boldsymbol{\Phi}_{n_{\rm d}} \end{bmatrix} = \begin{bmatrix} \mathbf{G}_1\mathbf{M}_1 \\ \vdots \\ \mathbf{G}_{n_{\rm d}}\mathbf{M}_{n_{\rm d}} \end{bmatrix} \mathbf{FZ}. \tag{14}$$

Since $\mathbf{G}_j \in \mathbb{C}^{M_j \times n_{\rm o}N_j}$ is composed of compact support kernels, the matrices $\mathbf{M}_j \in \mathbb{R}^{n_{\rm o}N_j \times n_{\rm o}N}$ can be introduced to select only the parts of the discrete Fourier plane involved in computations for block j, masking everything else. The selected, $n_{\rm o}N_j$, $N_j \leq N$, frequency points are directly linked to the continuous u–v coordinates associated with each of the visibility measurements from block $\boldsymbol{y}_j$. Thus, for a compact grouping of the visibilities in the u–v space, each block only deals with a limited frequency interval. These frequency ranges are not disjoint since a discrete frequency point is generally used for multiple visibilities due to the interpolation kernels and DDEs modelled through the operator $\mathbf{G}_j$. Since both have a compact support in frequency domain, without any loss of generality, we consider for each block j an overlap of $n_{\rm v}$ such points.

We rewrite (2) for each data block as

$$\boldsymbol{y}_j = \boldsymbol{\Phi}_j \boldsymbol{x} + \boldsymbol{n}_j, \tag{15}$$

with $\boldsymbol{n}_j$ being the noise associated with the measurements $\boldsymbol{y}_j$. Thus, we can redefine the minimization problem (12) as

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) + \sum_{i=1}^{n_{\rm b}} l_i(\boldsymbol{\Psi}_i^\dagger \boldsymbol{x}) + \sum_{j=1}^{n_{\rm d}} h_j(\boldsymbol{\Phi}_j \boldsymbol{x}), \tag{16}$$

where, similarly to (13), we have

$$\begin{aligned} &l_i = \|\cdot\|_1, \\ &h_j(\boldsymbol{z}) = \iota_{\mathcal{B}_j}(\boldsymbol{z}), \quad \mathcal{B}_j = \{\boldsymbol{z} \in \mathbb{C}^{M_j} : \|\boldsymbol{z} - \boldsymbol{y}_j\|_2 \leq \epsilon_j\}. \end{aligned} \tag{17}$$

Here, ϵ_j represents the bound on the noise for each block. For the sparsity priors, the ℓ_1 norm is additively separable and the splitting of the bases used,

$$\boldsymbol{\Psi} = \left[\boldsymbol{\Psi}_1 \ \ldots \ \boldsymbol{\Psi}_{n_{\rm b}}\right], \tag{18}$$

with $\boldsymbol{\Psi}_i \in \mathbb{C}^{N\times N}$ for $i \in \{1, \ldots, n_{\rm b}\}$, is immediate. The new formulation involving the ℓ_1 terms remains equivalent to the original one. Note that there are no restrictions on the number of blocks $\boldsymbol{\Psi}$ is split into. However, a different splitting strategy may not allow for the use of fast algorithms for the computation of the operator.

Hereafter, we focus on the block minimization problem defined in (16) and we describe two main algorithmic structures for finding the solution. The first class of methods uses a proximal ADMM and details the preliminary work of Carrillo et al. (2015). The second is based on the PD framework and introduces to RI, a new algorithm able to achieve the full splitting previously mentioned. These methods have a much lighter computational burden than the SDMM solver previously proposed by Carrillo et al. (2014). They are still able to achieve a similar level of parallelism, either through an efficient implementation in the case of ADMM or, in the case of PD, by making use of the inherent parallelizable structure of the algorithm. The main bottleneck of SDMM, which the proposed algorithms avoid, is the need to compute the solution of a linear system of equations, at each iteration. Such operation can be prohibitively slow for the large RI data sets and makes the method less attractive. The structure of SDMM is presented in Appendix B, Algorithm 3. For its complete description in the RI context, we direct the reader to Carrillo et al. (2014), the following presentation being focused on the ADMM and PD algorithms.

4.2 Dual FB-based ADMM

The ADMM is only applicable to the minimization of a sum of two functions and does not exhibit any intrinsic parallelization structure. However, by rewriting the minimization problem from (16) as

$$\min_{\boldsymbol{x}} \bar{f}(\boldsymbol{x}) + \bar{h}(\boldsymbol{\Phi}\boldsymbol{x}), \tag{19}$$

an efficient parallel implementation may be achieved. We define the two functions involved in as

$$\bar{f}(\boldsymbol{x}) = f(\boldsymbol{x}) + \sum_{i=1}^{n_{\rm b}} l_i(\boldsymbol{\Psi}_i^\dagger \boldsymbol{x}), \quad \bar{h}(\boldsymbol{\Phi}\boldsymbol{x}) = \sum_{j=1}^{n_{\rm d}} h_j(\boldsymbol{\Phi}_j \boldsymbol{x}). \tag{20}$$

Furthermore, since $\bar{h}$ is a sum of indicator functions $\iota_{\mathcal{B}_j}(\boldsymbol{\Phi}_j \boldsymbol{x})$, we can redefine it as $\bar{h}(\boldsymbol{\Phi}\boldsymbol{x}) = \iota_{\bar{\mathcal{B}}}(\boldsymbol{\Phi}\boldsymbol{x})$, with $\bar{\mathcal{B}} = \mathcal{B}_1 \times \mathcal{B}_2 \times \ldots \times \mathcal{B}_{n_{\rm d}}$.

ADMM iteratively searches for the solution to an augmented Lagrangian function similar to (11). The computations are performed in a serial fashion and explicit parallelization may only be introduced inside each of its three algorithmic steps. Thus, at each iteration, ADMM alternates between the minimization

$$\min_{\boldsymbol{x}} \mu \bar{f}(\boldsymbol{x}) + \frac{1}{2}\|\boldsymbol{\Phi}\boldsymbol{x} + \boldsymbol{s} - \boldsymbol{r}\|_2^2 \tag{21}$$

over the variable of interest $\boldsymbol{x}$ and the minimization involving the slack variable $\boldsymbol{r}$,

$$\min_{\boldsymbol{r}} \mu \bar{h}(\boldsymbol{r}) + \frac{1}{2}\|\boldsymbol{r} - \boldsymbol{\Phi}\boldsymbol{x} - \boldsymbol{s}\|_2^2. \tag{22}$$

These are followed by a gradient ascent with a step ϱ performed for the Lagrange multiplier variable $\boldsymbol{s}$. Given the definition of the

Algorithm 1 Dual forward-backward ADMM.

```
 1: given $\boldsymbol{x}^{(0)}, \boldsymbol{r}_j^{(0)}, \boldsymbol{s}_j^{(0)}, \boldsymbol{q}_j^{(0)}, \kappa, \rho, \varrho$
 2: repeat for $t = 1, \ldots$
 3:     $\tilde{\boldsymbol{b}}^{(t)} = \mathbf{FZ}\boldsymbol{x}^{(t-1)}$
 4:     $\forall j \in \{1, \ldots, n_\mathrm{d}\}$ set
 5:         $\boldsymbol{b}_j^{(t)} = \mathbf{M}_j \tilde{\boldsymbol{b}}^{(t)}$
 6:     end
 7:     $\forall j \in \{1, \ldots, n_\mathrm{d}\}$ distribute $\boldsymbol{b}_j^{(t)}$ and do in parallel
 8:         $\boldsymbol{r}_j^{(t)} = \mathcal{P}_{\mathcal{B}_j}\left(\mathbf{G}_j \boldsymbol{b}_j^{(t)} + \boldsymbol{s}_j^{(t-1)}\right)$
 9:         $\boldsymbol{s}_j^{(t)} = \boldsymbol{s}_j^{(t-1)} + \varrho\left(\mathbf{G}_j \boldsymbol{b}_j^{(t)} - \boldsymbol{r}_j^{(t)}\right)$
10:         $\boldsymbol{q}_j^{(t)} = \mathbf{G}_j^\dagger\left(\mathbf{G}_j \boldsymbol{b}_j^{(t)} + \boldsymbol{r}_j^{(t)} - \boldsymbol{s}_j^{(t)}\right)$
11:     end and gather $\boldsymbol{q}_j^{(t)}$
12:     $\tilde{\boldsymbol{x}}^{(t)} = \boldsymbol{x}^{(t-1)} - \rho \mathbf{Z}^\dagger \mathbf{F}^\dagger \sum_{j=1}^{n_\mathrm{d}} \mathbf{M}_j^\dagger \boldsymbol{q}_j^{(t)}$
13:     $\boldsymbol{x}^{(t)} = \textsc{DualFB}\left(\tilde{\boldsymbol{x}}^{(t)}, \kappa\right)$
14: until convergence
15: function $\textsc{DualFB}\left(\boldsymbol{z}, \kappa\right)$
16:     given $\boldsymbol{d}_i^{(0)}, \eta$
17:     $\bar{\boldsymbol{z}}^{(0)} = \mathcal{P}_{\mathcal{C}}\left(\boldsymbol{z}\right)$
18:     repeat for $k = 1, \ldots$
19:         $\forall i \in \{1, \ldots, n_\mathrm{b}\}$ do in parallel
20:             $\boldsymbol{d}_i^{(k)} = \frac{1}{\eta}\left(\mathcal{I} - \mathcal{S}_{\kappa\|\boldsymbol{\Psi}\|_\mathrm{S}}\right)\left(\eta \boldsymbol{d}_i^{(k-1)} + \boldsymbol{\Psi}_i^\dagger \bar{\boldsymbol{z}}^{(k-1)}\right)$
21:             $\tilde{\boldsymbol{d}}_i^{(k)} = \boldsymbol{\Psi}_i \boldsymbol{d}_i^{(k)}$
22:         end
23:         $\bar{\boldsymbol{z}}^{(k)} = \mathcal{P}_{\mathcal{C}}\left(\boldsymbol{z} - \sum_{i=1}^{n_\mathrm{b}} \tilde{\boldsymbol{d}}_i^{(k)}\right)$
24:     until convergence
25: return $\bar{\boldsymbol{z}}^{(k)}$
```

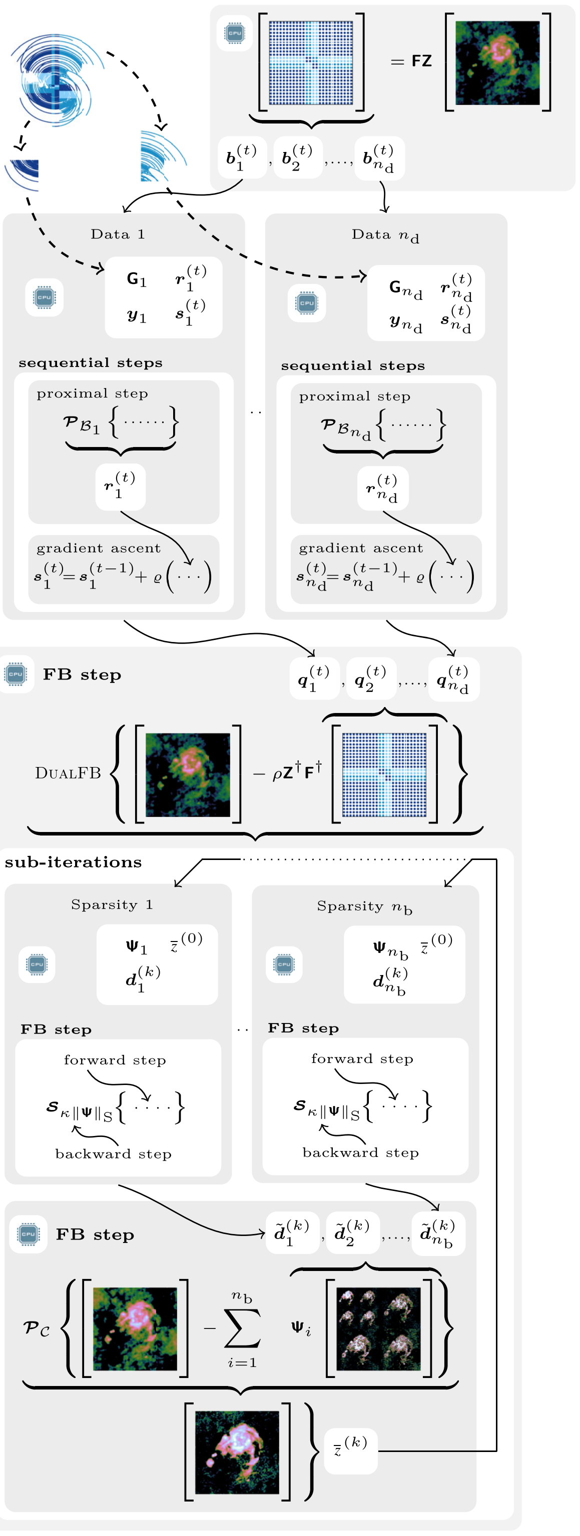
function $\bar{h}(\boldsymbol{r})$, the minimization involving $\boldsymbol{r}$ can be split into n_d independent sub-problems

$$\min_{\boldsymbol{r}_j} \mu \bar{h}_j(\boldsymbol{r}_j) + \frac{1}{2}\left\|\boldsymbol{r}_j - \boldsymbol{\Phi}_j \boldsymbol{x} - \boldsymbol{s}_j\right\|_2^2, \quad j \in \{1, \ldots, n_\mathrm{d}\}. \tag{23}$$

This minimization amounts to computing the proximity operator of $\mu \bar{h}_j$ at $\boldsymbol{\Phi}_j \boldsymbol{x} + \boldsymbol{s}_j$, which, given the definition of the function $\bar{h}_j$, reduces to a projection operation. The method imposes that every $\boldsymbol{r}_j$ approaches $\boldsymbol{\Phi}_j \boldsymbol{x}$ while $\boldsymbol{x}$ converges towards the solution. The convergence speed is governed by the Lagrange multiplier μ and by the ascent step ϱ associated with the maximization over the Lagrange multiplier variable $\boldsymbol{s}$.

A proximal version of ADMM deals with the non-smooth functions from (21) and (23) by approximating the solution via proximal splitting. Algorithm 1 presents the details. In Fig. 1, we present a diagram of the algorithm to provide further insight into its parallelization and distribution capabilities. It can also be used to understand the algorithm from the CLEAN perspective, performing FB CLEAN-like updates in multiple data, prior and image spaces. Data fidelity is enforced through the slack variables $\boldsymbol{r}_j^{(t)}$, by minimizing (23) and thus constraining the residual to belong to the ℓ_2 balls $\mathcal{B}_j$. This accepts a closed form solution and, for each ball j, represents the projection,

$$\mathcal{P}_{\mathcal{B}_j}(\boldsymbol{z}) \triangleq \begin{cases} \epsilon_j \dfrac{\boldsymbol{z} - \boldsymbol{y}_j}{\|\boldsymbol{z} - \boldsymbol{y}_j\|_2} + \boldsymbol{y}_j & \|\boldsymbol{z} - \boldsymbol{y}_j\|_2 > \epsilon_j \\ \boldsymbol{z} & \|\boldsymbol{z} - \boldsymbol{y}_j\|_2 \le \epsilon_j \end{cases} \tag{24}$$



= FZ
$b_1^{(t)}$, $b_2^{(t)}$, ..., $b_{n_d}^{(t)}$
Data 1
Data n_d
G_1 $r_1^{(t)}$ y_1 $s_1^{(t)}$
G_{n_d} $r_{n_d}^{(t)}$ y_{n_d} $s_{n_d}^{(t)}$
sequential steps
proximal step
$\mathcal{P}_{\mathcal{B}_1}\{\cdots\}$
$\mathcal{P}_{\mathcal{B}_{n_d}}\{\cdots\}$
gradient ascent
$s_1^{(t)} = s_1^{(t-1)} + \varrho(\cdots)$
$s_{n_d}^{(t)} = s_{n_d}^{(t-1)} + \varrho(\cdots)$
FB step
$q_1^{(t)}$, $q_2^{(t)}$, ..., $q_{n_d}^{(t)}$
DualFB
$-\rho Z^\dagger F^\dagger$
sub-iterations
Sparsity 1
Sparsity n_b
Ψ_1 $\bar{z}^{(0)}$ $d_1^{(k)}$
Ψ_{n_b} $\bar{z}^{(0)}$ $d_{n_b}^{(k)}$
forward step
$\mathcal{S}_{\kappa\|\Psi\|_S}\{\cdots\}$
backward step
$\tilde{d}_1^{(k)}$, $\tilde{d}_2^{(k)}$, ..., $\tilde{d}_{n_b}^{(k)}$
$\mathcal{P}_{\mathcal{C}}$
$-\sum_{i=1}^{n_b}$ Ψ_i
$\bar{z}^{(k)}$

Figure 1. The diagram of the structure of ADMM, detailed in Algorithm 1, showcasing the parallelism capabilities and overall computation flow. The algorithm performs in parallel proximal and gradient updates (similarly to the CLEAN performing major-minor cycle) for all data fidelity terms. Its structure is sub-iterative and enforces sparsity and positivity through the dual FB algorithm. These updates, performed in parallel for each sparsity basis, can be again seen as analogous to CLEAN. Thus, the whole algorithm can be seen as composed of interlaced CLEAN-like proximal splitting and FB updates running in parallel in multiple data, prior, and image spaces.

on to the feasible regions defined by it. Given the structure of the function $\bar{h}$, this is implemented in parallel with distributed computations and presented in Algorithm 1, step 8, together with the update of the Lagrange variables $\boldsymbol{s}_j^{(t)}$, in step 9. The variables $\boldsymbol{b}_j^{(t)} \in \mathbb{C}^{n_{\text{o}}N_j}$, computed in steps 3-6, are required in the computations and need to be transmitted to the different processing nodes. The nodes compute the solution updates $\boldsymbol{q}_j^{(t)} \in \mathbb{C}^{n_{\text{o}}N_j}$ in step 10, after which they are centralized and used to revise the previous solution estimate $\boldsymbol{x}^{(t-1)}$ and to compute $\boldsymbol{x}^{(t)}$. Thus, by carefully defining the minimization problem, a high degree of parallelism is achieved. Note that this step can easily incorporate all types of weighting of the data specific to RI.

For our specific problem, the minimization over $\boldsymbol{x}$ from (21) does not accept a closed form solution. We approximate it by using a FB step. The forward step corresponds to a gradient step and the backward step is an implicit sub-gradient-like step performed through the proximity operator. Thus, in step 12, the solution is updated using the descent step ρ, in the direction of the gradient of the smooth part. This is followed by the iterative dual FB (Combettes, Dũng & Vũ 2011) updates necessary to approximate the proximity operator to the non-smooth $\bar{f}$. Algorithm 1, function DualFB, details the required sub-iterations. In steps 23 and 20, the method alternates between projections on to the convex set $\mathcal{C}$, which, component wise, are defined as

$$\Big(\boldsymbol{\mathcal{P}}_{\mathcal{C}}(\boldsymbol{z})\Big)_k \triangleq \begin{cases} \Re(z_k) & \Re(z_k) > 0 \\ 0 & \Re(z_k) \le 0 \end{cases} \quad \forall k, \tag{25}$$

and the application of the proximity operator to the sparsity prior functions l_i, which is the component wise soft-thresholding operator

$$\Big(\boldsymbol{\mathcal{S}}_{\alpha}(\boldsymbol{z})\Big)_k \triangleq \begin{cases} \dfrac{z_k\{|z_k| - \alpha\}_+}{|z_k|} & |z_k| > 0 \\ 0 & |z_k| = 0 \end{cases} \quad \forall k, \tag{26}$$

with threshold α. The soft threshold resulting for the algorithm is $\eta\rho\mu$. However, since μ is a free parameter, we re-parametrize the operation to use the soft threshold $\kappa\|\boldsymbol{\Psi}\|_{\text{S}}$, with κ as a new scale-free parameter, independent of the operator $\boldsymbol{\Psi}$ used. Here, we denote by $\|\boldsymbol{\Psi}\|_{\text{S}}$ the operator norm of the sparsifying transform. The operator $\{\cdot\}_+$ from (26) sets the negative values to 0. The parameter η serves as an update step for the sub-problem. In step 20, we have additionally used the Moreau decomposition (C4) to replace the proximity operator of the conjugates l_i^* with that of the functions l_i, with $\boldsymbol{\mathcal{I}}$ denoting the identity operator. The computations involving each basis $\boldsymbol{\Psi}_i^\dagger$ are to be performed in parallel, locally. Distributed processing is problematic here due to the large size of the image $\bar{z}^{(k)}$ that would need to be transmitted.

4.3 PD algorithms with randomization

The main advantage that makes the PD algorithms attractive for solving inverse problems is their flexibility and scalability. They are able to deal with both differentiable and non-differentiable functions and are applicable to a broad range of minimization tasks. The inherent parallelization on the level of splitting the functions gives a direct approach for solving (16). Another important aspect is given by the use of randomization, allowing the update for a given component function to be performed less often and thus lowering the computational cost per iteration. Block coordinate computations are also supported but are not explicitly used herein.

We define the minimization task to be solved using PD methods, similarly to (16), as

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) + \gamma \sum_{i=1}^{n_{\text{b}}} l_i(\boldsymbol{\Psi}_i^\dagger \boldsymbol{x}) + \sum_{j=1}^{n_{\text{d}}} h_j(\boldsymbol{\Phi}_j \boldsymbol{x}), \tag{27}$$

where γ is an additional tuning parameter. Note that the minimization problem does not change, regardless of the value γ takes due to the use of the indicator functions in f and h_j which are invariant to scaling. This fits under the framework introduced by Condat (2013), Vũ (2013) and Pesquet & Repetti (2015) and we devise a PD algorithm towards finding the solution. The method iteratively alternates between solving the primal problem (27) and the dual problem,

$$\min_{\substack{\boldsymbol{u}_i \\ \boldsymbol{v}_j}} f^*\left(-\sum_{i=1}^{n_{\text{b}}} \boldsymbol{\Psi}_i \boldsymbol{u}_i - \sum_{j=1}^{n_{\text{d}}} \boldsymbol{\Phi}_j^\dagger \boldsymbol{v}_j\right) + \frac{1}{\gamma}\sum_{i=1}^{n_{\text{b}}} l_i^*(\boldsymbol{u}_i) + \sum_{j=1}^{n_{\text{d}}} h_j^*(\boldsymbol{v}_j), \tag{28}$$

essentially converging towards a Kuhn–Tucker point. This produces the algorithmic structure of Algorithm 2 where additionally we have used the Moreau decomposition (C4) to rewrite the proximal operations and replace the function conjugates. A diagram of the structure is presented in Fig. 2 further exemplifying the conceptual analogy between the PD algorithm and CLEAN. The algorithm allows the full split of the operations and performs all the updates on the dual variables in parallel. The update of the primal variable, the image of interest $\boldsymbol{x}^{(t)}$, requires the contribution of all dual variables $\boldsymbol{v}_i^{(t)}$ and $\boldsymbol{u}_j^{(t)}$. The algorithm uses the update steps τ, σ_i and ς_j to iteratively revise the solution and allows for a relaxation with the factor λ. FB iterations, consisting of a gradient descent step coupled with a proximal update, are used to update both the primal and the dual variables. These FB updates can be seen as CLEAN-like steps performed in the multiple signal spaces associated with the primal and the dual variables. In the deterministic case, the active sets $\mathcal{P}$ and $\mathcal{D}$ are fixed such that all the dual variables are used. The randomization capabilities of the algorithm are presented later on, given a probabilistic construction of the active sets.

When applied in conjunction with the functions from (17), the primal update from step 28 is performed through the projection (25) on to the positive orthant defined by $\mathcal{C}$. The dual variables are updated in steps 10 and 19 using the proximity operators for h_j and l_i, which become the projection on to an ℓ_2 ball $\mathcal{B}_j$ defined by (24) and the component wise soft-thresholding operator (26). We use the Moreau decomposition (C4) to replace the proximity operator of the conjugate functions l_i^* and h_j^* with that of the function l_i and h_j, respectively. The identity operator is denoted by $\boldsymbol{\mathcal{I}}$. Step 19 also contains a re-parametrization similar to the one performed for ADMM. We replace the implicit algorithmic soft-threshold size $\frac{\gamma}{\sigma_i}$ with $\kappa\|\boldsymbol{\Psi}\|_{\text{S}}$ by appropriately choosing the free parameter γ. This ensures that we are left with the scale-free parameter κ independent to the operator $\boldsymbol{\Psi}$. Steps 11, 20 and 29 represent the relaxation of the application of the updates. To make use of the parallelization, the application of the operators $\mathbf{G}_j^\dagger$ and $\boldsymbol{\Psi}_i$ is also performed in parallel, in steps 12 and 21. Note that the splitting of the operators is presented in (14), more specifically $\boldsymbol{\Phi}_j = \mathbf{G}_j\mathbf{M}_j\mathbf{FZ}, \forall j \in \{1, \ldots, n_{\text{d}}\}$. These operations are given in steps 4–7.

The computation of the dual variables $\boldsymbol{u}_i^{(t)}$ associated with the sparsity priors requires the current solution estimate. This solution

Algorithm 2 Randomized forward-backward PD.

```
1:  given $\boldsymbol{x}^{(0)}, \tilde{\boldsymbol{x}}^{(0)}, \boldsymbol{u}_i^{(0)}, \boldsymbol{v}_j^{(0)}, \tilde{\boldsymbol{u}}_i^{(0)}, \tilde{\boldsymbol{v}}_j^{(0)}, \kappa, \tau, \sigma_i, \varsigma_j, \lambda$
2:  repeat for $t = 1, \ldots$
3:      generate sets $\mathcal{P} \subset \{1, \ldots, n_\mathrm{b}\}$ and $\mathcal{D} \subset \{1, \ldots, n_\mathrm{d}\}$
4:      $\tilde{\boldsymbol{b}}^{(t)} = \mathbf{FZ}\tilde{\boldsymbol{x}}^{(t-1)}$
5:      $\forall j \in \mathcal{D}$ set
6:          $\boldsymbol{b}_j^{(t)} = \mathbf{M}_j \tilde{\boldsymbol{b}}^{(t)}$
7:      end
8:      run simultaneously
9:          $\forall j \in \mathcal{D}$ distribute $\boldsymbol{b}_j^{(t)}$ and do in parallel
10:             $\bar{\boldsymbol{v}}_j^{(t)} = \left(\mathcal{I} - \mathcal{P}_{\mathcal{B}_j}\right)\left(\boldsymbol{v}_j^{(t-1)} + \mathbf{G}_j \boldsymbol{b}_j^{(t)}\right)$
11:             $\boldsymbol{v}_j^{(t)} = \boldsymbol{v}_j^{(t-1)} + \lambda\left(\bar{\boldsymbol{v}}_j^{(t)} - \boldsymbol{v}_j^{(t-1)}\right)$
12:             $\tilde{\boldsymbol{v}}_j^{(t)} = \mathbf{G}_j^\dagger \boldsymbol{v}_j^{(t)}$
13:         end and gather $\tilde{\boldsymbol{v}}_j^{(t)}$
14:         $\forall j \in \{1, \ldots n_\mathrm{d}\} \setminus \mathcal{D}$ set
15:             $\boldsymbol{v}_j^{(t)} = \boldsymbol{v}_j^{(t-1)}$
16:             $\tilde{\boldsymbol{v}}_j^{(t)} = \tilde{\boldsymbol{v}}_j^{(t-1)}$
17:         end
18:         $\forall i \in \mathcal{P}$ do in parallel
19:             $\bar{\boldsymbol{u}}_i^{(t)} = \left(\mathcal{I} - \mathcal{S}_{\kappa\|\boldsymbol{\Psi}\|_\mathrm{S}}\right)\left(\boldsymbol{u}_i^{(t-1)} + \boldsymbol{\Psi}_i^\dagger \tilde{\boldsymbol{x}}^{(t-1)}\right)$
20:             $\boldsymbol{u}_i^{(t)} = \boldsymbol{u}_i^{(t-1)} + \lambda\left(\bar{\boldsymbol{u}}^{(t)} - \boldsymbol{u}_i^{(t-1)}\right)$
21:             $\tilde{\boldsymbol{u}}_i^{(t)} = \boldsymbol{\Psi}_i \boldsymbol{u}_i^{(t)}$
22:         end
23:         $\forall i \in \{1, \ldots n_\mathrm{b}\} \setminus \mathcal{P}$ set
24:             $\boldsymbol{u}_i^{(t)} = \boldsymbol{u}_i^{(t-1)}$
25:             $\tilde{\boldsymbol{u}}_i^{(t)} = \tilde{\boldsymbol{u}}_i^{(t-1)}$
26:         end
27:     end
28:     $\bar{\boldsymbol{x}}^{(t)} = \mathcal{P}_\mathcal{C}\left(\boldsymbol{x}^{(t-1)} - \tau\left(\mathbf{Z}^\dagger\mathbf{F}^\dagger \sum_{j=1}^{n_\mathrm{d}} \varsigma_j \mathbf{M}_j^\dagger \tilde{\boldsymbol{v}}_j^{(t)} + \sum_{i=1}^{n_\mathrm{b}} \sigma_i \tilde{\boldsymbol{u}}_i^{(t)}\right)\right)$
29:     $\boldsymbol{x}^{(t)} = \boldsymbol{x}^{(t-1)} + \lambda\left(\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t-1)}\right)$
30:     $\tilde{\boldsymbol{x}}^{(t)} = 2\bar{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t-1)}$
31: until convergence
```

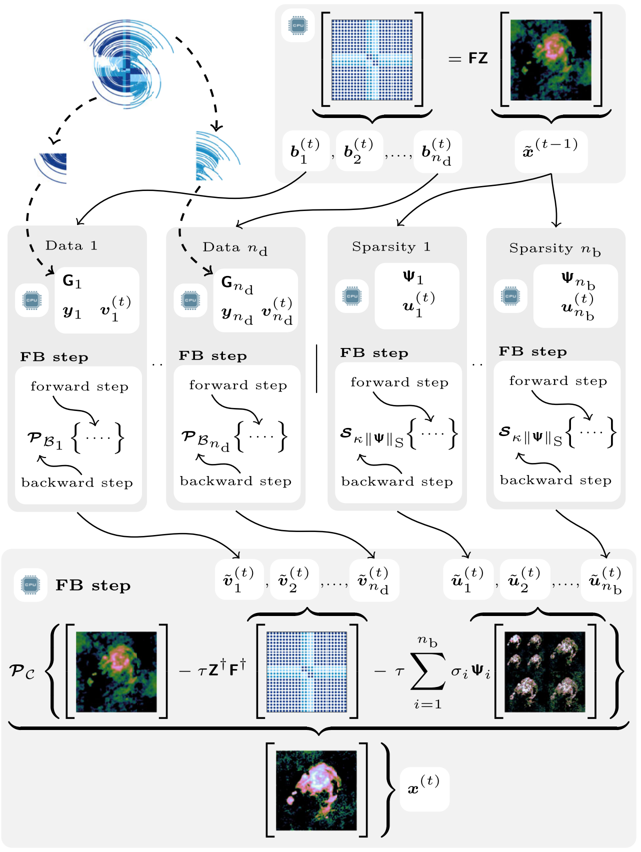

= FZ
$b_1^{(t)}, b_2^{(t)}, \ldots, b_{n_\mathrm{d}}^{(t)}$
$\tilde{x}^{(t-1)}$
Data 1
G_1
y_1 $v_1^{(t)}$
Data n_d
G_{n_d}
y_{n_d} $v_{n_\mathrm{d}}^{(t)}$
Sparsity 1
Ψ_1
$u_1^{(t)}$
Sparsity n_b
Ψ_{n_b}
$u_{n_\mathrm{b}}^{(t)}$
FB step
forward step
$\mathcal{P}_{\mathcal{B}_1}\{\cdots\}$
$\mathcal{P}_{\mathcal{B}_{n_\mathrm{d}}}\{\cdots\}$
$\mathcal{S}_{\kappa\|\Psi\|_\mathrm{S}}\{\cdots\}$
backward step
$\tilde{v}_1^{(t)}, \tilde{v}_2^{(t)}, \ldots, \tilde{v}_{n_\mathrm{d}}^{(t)}$
$\tilde{u}_1^{(t)}, \tilde{u}_2^{(t)}, \ldots, \tilde{u}_{n_\mathrm{b}}^{(t)}$
$\mathcal{P}_\mathcal{C}$
$-\tau Z^\dagger F^\dagger$
$-\tau \sum_{i=1}^{n_\mathrm{b}} \sigma_i \Psi_i$
$x^{(t)}$

Figure 2. The diagram of structure of PD, detailed in Algorithm 2, showcasing the parallelism capabilities and overall computation flow. In contrast with ADMM, the PD algorithm is able to perform all updates on the dual variables $\boldsymbol{v}_i^{(t)}$ and $\boldsymbol{u}_j^{(t)}$ using FB iterations and in parallel. The update of the primal variable $\boldsymbol{x}^{(t)}$ is also an FB step. Viewed though the perspective of the intuitive similarity between an FB iteration and CLEAN, this translates to performing CLEAN-like iterations in parallel in multiple data, prior, and image spaces.

estimate is then revised with the updates $\tilde{\boldsymbol{u}}_i^{(t)}$ computed from the dual variables. Both $\boldsymbol{x}^{(t)}$ and $\tilde{\boldsymbol{u}}_i^{(t)}$ are of size N and their communication might not be desirable in a loosely distributed system. In such case, all computations involving $\boldsymbol{u}_i^{(t)}$ can be performed in parallel but not in a distributed fashion. The dual variables $\boldsymbol{v}_j^{(t)}$, associated with the data fidelity functions, should be computed over a distributed computing network. They only require the communication of the updates $\boldsymbol{b}_j^{(t)} \in \mathbb{C}^{n_\mathrm{o}N_j}$ and dual updates $\tilde{\boldsymbol{v}}_j^{(t)} \in \mathbb{C}^{n_\mathrm{o}N_j}$ which remains feasible.

The main challenge associated with the inverse problem defined by (2) is linked with the dimensionality of the data. The large data size is a limiting factor not only from the computational perspective but also from that of memory availability. A randomization of the computations following the same PD framework (Pesquet & Repetti 2015) is much more flexible at balancing memory and computational requirements. By selectively deciding which data fidelity and sparsity prior functions are active at each iterations, full control over the memory requirements and computational cost per iteration can be achieved. In Algorithm 2, this is controlled by changing the sets $\mathcal{P}$, containing the active sparsity prior dual variables, and $\mathcal{D}$, which governs the selection of the data fidelity dual variables. At each iteration, each dual variable has a given probability of being selected, $p_{\mathcal{P}_i}$ for the sparsity prior, and $p_{\mathcal{D}_j}$ for the data fidelity, respectively. These probabilities are independent of each other. Note that the algorithm has inertia still performing the primal updates using all dual variables even though some dual variables remain unchanged.

5 IMPLEMENTATION DETAILS AND COMPUTATIONAL COMPLEXITY

An efficient implementation of the ADMM and the PD algorithms takes advantage of the data split and of the implicit parallelization from the definition of the minimization problem. For presentation simplicity, we consider the processing to be split between a central meta-node, a single processing unit or possibly a collection of nodes, centralizing the update on the desired solution $\boldsymbol{x}^{(t)}$ and performing the computations associated with the sparsity priors, and a number of data fidelity nodes dealing with the constraints involving the balls $\mathcal{B}_j$. The computation of the sparsity prior terms can be easily parallelized, however, the distribution of the data can be too costly. In this case, a shared memory architecture might be more appropriate than distributed processing. For the data nodes, the communication cost is low and a distributed approach is feasible. We have assumed

Table 1. Complexity of ADMM (top) and PD (bottom) algorithms for one iteration. Each node has its computational load listed. The ADMM algorithm iterates $n_{\bar{f}}$ times over steps 18–23. The serial nature of its structure can be observed, the nodes not operating simultaneously. The PD methods alternate between updating the primal and the dual variables. All dual variables are computed in parallel. The visibility data are assumed to be split into compact blocks composed of an equal number of visibilities in the u–v space.

Algorithm 1 (ADMM)	Central node	n_d data fidelity nodes
steps 3–6	$\mathcal{O}\left(n_\mathrm{o}N \log n_\mathrm{o}N\right)$	–
steps 8–10	–	$\mathcal{O}\left(2\frac{n_\mathrm{s}n_\mathrm{o}}{n_\mathrm{d}}MN_j + M_j\right)$
step 12	$\mathcal{O}\left(n_\mathrm{o}N \log n_\mathrm{o}N\right) + \mathcal{O}\left(n_\mathrm{o}N + n_\mathrm{d}n_\mathrm{v}\right)$	–
$n_{\bar{f}}\times$ steps 19–23	$\mathcal{O}\left(2n_\mathrm{b}N\right)$	–
Algorithm 2 (PD)	**Central node**	**n_d data fidelity nodes**
steps 4–7	$\mathcal{O}\left(n_\mathrm{o}N \log n_\mathrm{o}N\right)$	–
steps 9–26	$p_{\mathcal{P}_i}\mathcal{O}\left(2n_\mathrm{b}N\right)$	$p_{\mathcal{D}_j}\mathcal{O}\left(2\frac{n_\mathrm{s}n_\mathrm{o}}{n_\mathrm{d}}MN_j + M_j\right)$
steps 28–30	$\mathcal{O}\left(n_\mathrm{o}N \log n_\mathrm{o}N\right) + \mathcal{O}\left((n_\mathrm{b}+n_\mathrm{o})N + n_\mathrm{d}n_\mathrm{v}\right)$	–

these two different strategies for dealing with the different terms in the presentation of Algorithms 1 and 2.

Most of the operations to be performed are proportional with N since the main variable of interest $\boldsymbol{x}^{(t)}$ is the image to be recovered. The most demanding operation performed on $\boldsymbol{x}^{(t)}$ is the application of the oversampled Fourier operators. When computed with a fast Fourier algorithm (FFT; Cooley & Tukey 1965), the computational cost of the transforms F and $\mathsf{F}^\dagger$ applied to n_o-oversampled data scales as $\mathcal{O}\left(n_\mathrm{o}N \log n_\mathrm{o}N\right)$. It should be noted that the FFT implementation can be sped up by using multiple processing cores or nodes. The wavelet operators $\boldsymbol{\Psi}$ and $\boldsymbol{\Psi}^\dagger$ are applied to the image $\boldsymbol{x}^{(t)}$ as well. The Discrete Wavelet Transform (DWT) can be performed with fast wavelet implementations using lifting schemes or filter banks (Cohen, Daubechies & Vial 1993; Daubechies & Sweldens 1998; Mallat 2008) and achieves a linear complexity of $\mathcal{O}(N)$ for compactly supported wavelets. A distributed processing of the operations involved in the application of each sparsity basis $\boldsymbol{\Psi}_i$ may be used. However, this requires the communication of the current solution estimate, which might not be feasible. We consider that these computations are performed locally, on the central meta-node.

For the data nodes, a manageable computational load and an efficient communication can be achieved by both algorithms by adopting a balanced and compact split of the data; splitting the data into blocks of similar size having a compact frequency range as proposed in (14). An overlap of size n_v between discrete frequency ranges is necessary for an efficient interpolation (Fessler & Sutton 2003) to the uniform frequency grid which allows fast Fourier computations or to include DDEs (Wolz et al. 2013). Besides this overlap, each block only deals with a limited frequency range reducing the communication performed. In such case, the matrices M_j mask out the frequencies outside the range associated with the blocks $\boldsymbol{y}_j$. Furthermore, the use of compact support interpolation kernels and DDEs with compact support in the Fourier domain makes G_j sparse, which lowers the computational load significantly. We consider it has a generic sparsity percentage n_s.

Details on the levels of parallelization and the scaling to multiple nodes for both methods are presented below. As mentioned earlier, the main computational difficulties arise from working with large images and data sets, thus making important the way the complexity of the algorithms scales with N and M. An overview of the complexity requirements is presented in Table 1.

5.1 Alternating direction method of multipliers

The efficient implementation of ADMM for the problem defined by (19) offloads the data fidelity computations to the data nodes. As can be seen from Fig. 1 and Table 1, the basic structure of the algorithm is serial and the processing is just accelerated by parallelising each serial step.

The iterative updates follow the operations presented in Algorithm 1. The central node computes an estimate $\tilde{\boldsymbol{x}}^{(t)}$ of the solution and iteratively updates it to enforce sparsity and positivity. The update from step 12 requires $\mathcal{O}\left(n_\mathrm{o}N \log n_\mathrm{o}N\right)$ operations for the computation of the oversampled FFT. Given a compact partitioning of the matrix G, the sum involving the updates $\boldsymbol{q}_j^{(t)}$ requires computations of the order $\mathcal{O}(n_\mathrm{o}N) + \mathcal{O}(n_\mathrm{d}n_\mathrm{v})$. Note that it may be accelerated by using the data node network, however, since generally n_v is not large, the gain remains small. The computation of the Fourier coefficients from step 3 also incurs a complexity $\mathcal{O}\left(n_\mathrm{o}N \log n_\mathrm{o}N\right)$.

For the approximation of the proximal operator of the function $\bar{f}$, the algorithm essentially remains serial and requires a number $n_{\bar{f}}$ of iterations. In this case, the complexity of each update performed for the sparsity prior is dominated by the application of the operators $\boldsymbol{\Psi}$ and $\boldsymbol{\Psi}^\dagger$, which, given an efficient implementation of the DWT, requires $\mathcal{O}(N)$ operations. The updates $\boldsymbol{d}_i^{(k)}$ and $\tilde{\boldsymbol{d}}_i^{(k)}$ from step 20 and 21, may be computed in parallel. Given a serial processing, however, this would need $\mathcal{O}(n_\mathrm{b}N)$ computations. Note that although in this case the complexity scales linearly with N, the scaling constants can make the computations to be of the same level as the FFT.

The data fidelity nodes perform steps 8–10 in parallel using the Fourier coefficients $\boldsymbol{b}_j^{(t)}$ pre-computed in step 5. The computations are heavier due to the linear operator G_j. As mentioned earlier, the operator has a very sparse structure. This reduces the computation cost for applying G_j or $\mathsf{G}_j^\dagger$ to $\mathcal{O}(n_\mathrm{s}M_jn_\mathrm{o}N_j)$, where $n_\mathrm{o}N_j$ is the number of uniformly gridded, frequency points associated with each visibility block $\boldsymbol{y}_j$. The remaining operations only involve vectors of size M_j. The overall resulting complexity per node is $\mathcal{O}(n_s M_j n_o N_j) + \mathcal{O}(M_j)$. Under the assumption that the blocks contain an equal number of visibilities, this further reduces to $\mathcal{O}(\frac{n_s}{n_d} M n_o N_j) + \mathcal{O}(M_j)$. The communication required between the central and the data fidelity nodes is of order $n_o N_j$, the size of frequency range of each data block.

5.2 PD algorithm

An implementation of the PD algorithms benefits from the full split achieved by the methods which allows for the computation of all the dual variables to be completed in parallel. The processing is performed in two synchronous alternating serial steps to update the primal and dual variables, respectively. Each step is however highly parallelizable. The central node uses the current estimate of the solution $\boldsymbol{x}^{(t-1)}$ and distributes the oversampled Fourier transform coefficients $\boldsymbol{b}_j^{(t)}$ to the data fidelity nodes. The data fidelity and central nodes compute simultaneously the dual variables and provide the updates $\tilde{\boldsymbol{v}}_j^{(t)}$ and $\tilde{\boldsymbol{u}}_i^{(t)}$ to be centralized and included in the next solution estimate on the central node. Such a strategy requires at each step the propagation of variables of size $n_o N_j$, between the central and data fidelity nodes. As suggested in Algorithms 2, the computation of the sparsity prior dual variables is also highly parallelizable. However, the communication of the current image estimate is required, limiting the possibility to distribute the data due to its large size. We leave the computation to be performed by the central node, without an explicit exploitation of the possible parallelism.

All dual variables can be computed simultaneously as can be seen in Fig. 2. The data fidelity nodes need to apply the linear operators $\mathbf{G}_j$ as in steps 10 and 12. Similarly to ADMM, this incurs the heaviest computational burden. Given the very sparse structure of the matrix $\mathbf{G}_j$, this accounts for a complexity of $\mathcal{O}(n_s M_j n_o N_j)$, with $n_o N_j$ being the previously mentioned number of, uniformly gridded, frequency points for the visibilities $\boldsymbol{y}_j$. The remaining operations only involve vectors of size M_j and thus the overall resulting complexity is $\mathcal{O}(2 n_s M_j n_o N_j) + \mathcal{O}(2M_j)$. The wavelet decomposition from steps 19 and 21 achieves a linear complexity of $\mathcal{O}(N)$ for compactly supported wavelets. The other operations from steps 19 and 20 are of order $\mathcal{O}(N)$ resulting in a load for the sparsity prior nodes that scales linearly with N.

In step 28 of Algorithm 2, the summing of the sparsity prior updates requires $\mathcal{O}(n_b N)$ operations. For the ℓ_2 data fidelity terms, given a compact partitioning in frequency for the matrix $\mathbf{G}$, the computation requires $\mathcal{O}(n_o N) + \mathcal{O}(n_d n_v)$ operations. The computational cost of the transforms $\mathbf{F}$ and $\mathbf{F}^\dagger$, steps 4 and 28, scales as $\mathcal{O}(n_o N \log n_o N)$ since this requires the FFT computation of the n_o-oversampled image. The remaining operations, including the projection, are $\mathcal{O}(N)$, giving the complexity of the primal update step $\mathcal{O}(n_o N \log n_o N) + \mathcal{O}((n_b + n_o)N) + \mathcal{O}(N) + \mathcal{O}(n_d n_v)$. We kept the terms separate to give insight on how the algorithms scales for different configurations. Similarly to ADMM, the sums may be performed over the network in a distributed fashion, further reducing the complexity and leaving the primal update step dominated by the Fourier computations.

The randomized PD algorithm introduces an even more scalable implementation. To achieve a low computational burden per data node, the number of nodes has to be very large in order to reduce the size of M_j and N_j for each block. The randomized algorithms achieve greater flexibility by allowing some of the updates for the sparsity prior or data fidelity dual variables, to be skipped at the current iteration. Given a limited computing infrastructure, by carefully choosing the probabilities, we can ensure that data fit into memory and that all available nodes are processing parts of it. The average computational burden per iteration is lowered proportionally to the probability of selection, $p_{\mathcal{P}_i}$ and $p_{\mathcal{D}_j}$. In practice, this also produces an increase in the number of iterations needed to achieve convergence, requiring a balanced choice for the probabilities.

5.3 Splitting the data

As reported earlier, the modality in which the data are split can have a big impact in the scalability of the algorithms. Ideally, each data node should process an identical number of visibilities for the computation to be spread evenly. If the visibilities used by one node are, however, spread over the whole u–v plane, their processing requires all the discrete Fourier points. Due to this, a compact grouping in frequency domain is also important since it determines the size of the data to be communicated. Ideally, the splitting should be performed taking into account the computing infrastructure and should balance the communication and computation loads which are linked to the size of N_j and M_j.

6 SIMULATIONS AND RESULTS

We study the performance of the algorithms developed herein for different configuration parameters and compare the reconstruction performance against CS-CLEAN (Schwab 1984) and MORESANE (Dabbech et al. 2015). We denote the methods as follows: SDMM, the method introduced by Carrillo et al. (2014); ADMM, the approach described in Algorithm 1; PD and PD-R, the algorithms presented in Algorithm 2 without and with randomization, respectively; MORESANE, the algorithm[2] from Dabbech et al. (2015); CS-CLEAN, the Cotton–Schwab CLEAN (Schwab 1984) algorithm.[3] For both MORESANE and CS-CLEAN, we perform tests for three types of weighting: natural weighting denoted by -N, uniform weighting denoted by -U and Briggs weighting with the robustness parameter set to 1 denoted by -B.

The reconstruction performance is assessed in terms of the signal-to-noise ratio,

$$\mathrm{SNR} = 20 \log_{10} \left(\frac{\|\boldsymbol{x}^\circ\|_2}{\|\boldsymbol{x}^\circ - \boldsymbol{x}^{(t)}\|_2} \right), \tag{29}$$

where $\boldsymbol{x}^\circ$ is the original image and $\boldsymbol{x}^{(t)}$ is the reconstructed estimate of the original, averaged over 10 simulations performed for different noise realizations. For the tests involving the comparison with CS-CLEAN and MORESANE on the VLA and SKA coverages, we do not perform this averaging. In the latter case, we also report the dynamic range

$$\mathrm{DR} = \frac{\sqrt{N}\|\boldsymbol{\Phi}\|_S^2}{\|\boldsymbol{\Phi}^\dagger(\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{x})\|_2} \max_{k,l} x_{k,l} \tag{30}$$

obtained by all algorithms.

6.1 Simulation setup

In the first part of the simulations, we evaluate the influence of the different configuration parameters for PD, PD-R and ADMM. Here, we also validate their performance against SDMM, a previously proposed solver (Carrillo et al. 2014) for the same optimization task. The test images, as shown in Fig. 3, represent a small 256 × 256

[2] We have used the MORESANE implementation from WS-CLEAN (Offringa et al. 2014), https://sourceforge.net/p/wsclean/wiki/Home/.
[3] We have used the CS-CLEAN implementation of LWIMAGER from CASACORE, https://github.com/casacore/.

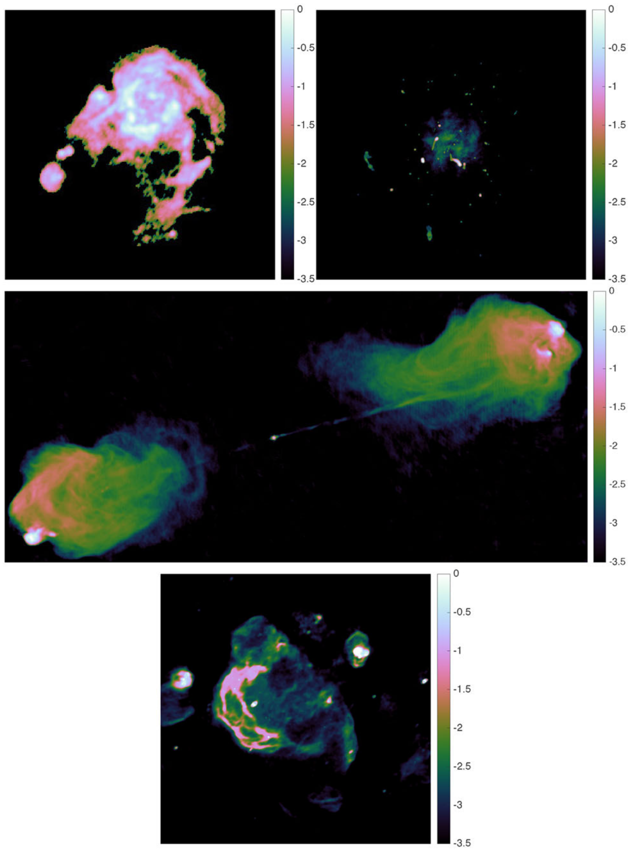


Figure 3. The test images, from left to right, top to bottom, a 256 × 256 image of the M31 galaxy, a 512 × 512 galaxy cluster image, a 477 × 1025 image of Cygnus A and a 1024 × 1024 image of the W28 supernova remnant, all shown in $\log_{10}$ scale.

image of the H II region of the M31 galaxy, a 512 × 512 high-dynamic range image of a galaxy cluster with faint extended emissions, and a 477 × 1025 image of the Cygnus A radio galaxy, respectively. The galaxy cluster image was produced using the FARADAY tool (Murgia et al. 2004). We reconstruct them from simulated visibility data. We use a u–v coverage generated randomly through Gaussian sampling, with zero mean and variance of 0.25 of the maximum frequency, creating a concentration of visibility data in the centre of the plane, for low frequencies. We introduce holes in the coverage with an inverse Gaussian profile, placing the missing spectrum information predominantly in high frequency. This generates very generic profiles and allows us to study the algorithm performance with a large number of different coverages. A typical u–v coverage is presented in Fig. 4.

The second part of the simulations involves testing the algorithm reconstruction using simulated VLA and SKA coverages[4] corresponding to 5 and 9 h of observations, respectively. The coverages are presented in Fig. 4. For the tests, we use an additional large 1024 × 1024 image, also presented in Fig. 3, representing the W28 supernova remnant.[5] We showcase the reconstruction quality and speed of convergence for PD and ADMM without performing any re-weighting[6] and compare the results with those produced by CS-CLEAN and MORESANE.

In both cases, we have normalized the frequencies to the interval $[-\pi, \pi]$. The visibilities are corrupted by zero mean complex Gaussian noise producing a signal-to-noise level of 20 dB. The bound ϵ_j, for the ball $\mathcal{B}_j$ defined by (17), can be therefore estimated based on the noise variance σ_χ^2 of the real and imaginary parts of the noise, the residual norm being distributed according to a χ^2 distribution with $2M_j$ degrees of freedom. Thus, we impose that the square of the global bound ϵ^2 is two standard deviations above the mean of the χ^2 distribution, $\epsilon^2 = (2M + 2\sqrt{4M})\sigma_\chi^2$. The resulting block constraints must satisfy $\sum_{j=1}^{n_\mathrm{d}} \epsilon_j^2 = \epsilon^2$. When all blocks have the same size, this results in $\epsilon_j^2 = (2M_j + \frac{2}{\sqrt{n_\mathrm{d}}}\sqrt{4M_j})\sigma_\chi^2$.

We work with pre-calibrated measurements. For simplicity, we assume, without loss of generality, the absence of DDEs and a small field of view, the measurement operator reducing to a Fourier matrix sampled at the M frequencies that characterize the visibility points. We have used an oversampled Fourier transform $\mathbf{F}$ with $n_\mathrm{o} = 4$ and a matrix $\mathbf{G}$ that performs an interpolation of the frequency data, linking the visibilities to the uniformly sampled frequency space. The 8 × 8 interpolation kernels (Fessler & Sutton 2003) average nearby uniformly distributed frequency values to estimate the value at the frequencies associated with each visibility. A scaling is also introduced in image space to pre-compensate for imperfections in the interpolation. This allows for an efficient implementation of the operator.

To detail the behaviour of the algorithms, we vary the number of blocks n_d used for the data fidelity term. Tests are performed for 4, 16 and 64 blocks. In each case, the blocks are generated such that they have an equal number of visibility points, which cover a compact region in the u–v space. An example of the grouping for the 16 blocks is overlaid on the randomly generated coverage from Fig. 4. The figure also contains, marked with dashed lines, an example of the discrete frequency points required to model the visibilities for two of the blocks, under our previous assumptions, for the M31 image. The number of discrete frequency points required for each block would only grow slightly in the presence of DDEs due to their, possible larger, compact support. The overall structure from Fig. 4 would remain similar. For the SKA and VLA coverages, the data are also split into blocks of equal size. The resulting block structure is also presented in Fig. 4. As sparsity prior, we use the SARA collection of wavelets (Carrillo et al. 2012), namely a concatenation of a Dirac basis with the first eight Daubechies wavelets. We split the collection of bases into $n_\mathrm{b} = 9$ individual basis.

6.2 Choice of parameters

The ADMM, PD and PD-R algorithms converge given that (D1) and (D3), respectively, are satisfied. To ensure this, we set for PD $\sigma = \frac{1}{\|\boldsymbol{\Psi}\|_\mathrm{S}^2}$, $\varsigma = \frac{1}{\|\boldsymbol{\Phi}\|_\mathrm{S}^2}$ and $\tau = 0.49$. The relaxation parameter is set to 1. For the ADMM algorithm, we set $\rho = \frac{1}{\|\boldsymbol{\Phi}\|_\mathrm{S}^2}$ and $\eta = \frac{1}{\|\boldsymbol{\Psi}\|_\mathrm{S}^2}$. The ascent step is set $\varrho = 0.9$. The maximum number of sub-iterations is set to $n_{\bar{f}} = 100$. We consider the convergence achieved, using a criterion similar to (31), when the relative solution variation for $\bar{z}^{(k)}$ is below 10^{-3}. The norms of the operators are computed a priori using the power iterative method. They act as a normalization of the

[4] The SKA and VLA u–v coverages are generated using the CASA and CASACORE software package: https://casa.nrao.edu/ and https://github.com/casacore.

[5] Image courtesy of NRAO/AUI and Brogan et al. (2006).

[6] Performing the re-weighting improves the reconstruction (Carrillo et al. 2012, 2014) but falls outside the scope of this study.

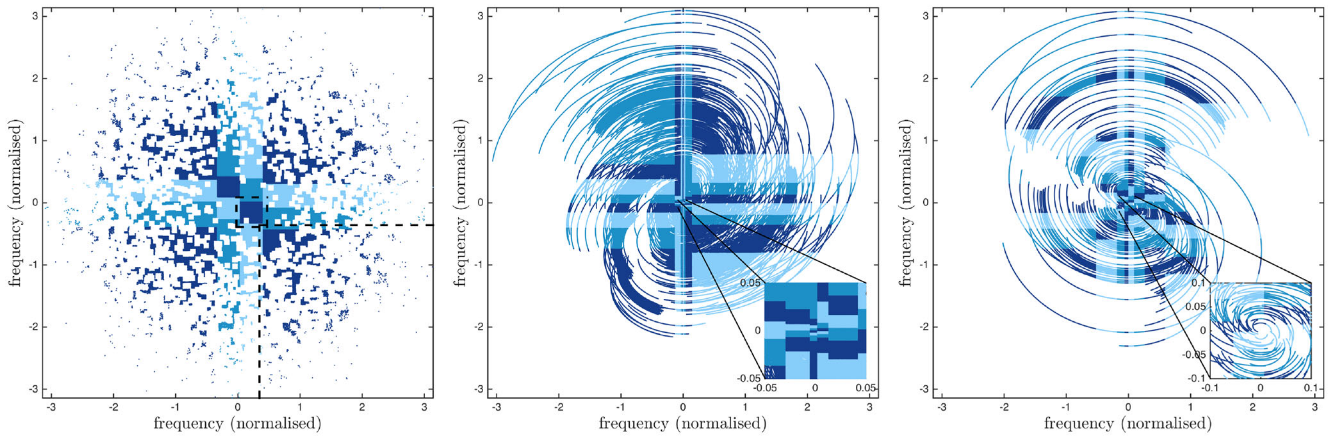


Figure 4. (left) An example of randomly generated coverage with the number of visibilities $M = 655\,360$. The visibilities are split into 16 equal size blocks, marked with different colours, with compact u–v grouping. The dashed lines mark the parts of the discrete Fourier space involved in the computations associated with the central bottom-right and the bottom-right blocks, respectively. In this case, the whole discrete frequency space is considered to have 512 × 512 points. (centre) The SKA u–v coverage for 5 h of observation corresponding to $M = 5791\,800$. (right) The VLA u–v coverage for 9 h of observations corresponding to $M = 1788\,480$. The SKA and VLA data are split into 64 blocks containing an equal number of visibilities.

updates, enabling the algorithm to deal with different data or image scales.

We leave the normalized soft-threshold values κ as a configuration parameter for both PD and ADMM. SDMM has a similar parameter κ. It influences the convergence speed which is of interest since, given the scale of the problem, we want to minimize the computational burden which is inherently linked to the number of iterations performed. We aim at providing a general prescription for this tuning parameter, similarly to the standard choices for the loop gain factor used by CLEAN. Intuitively, this soft-thresholding parameter can be seen as analogous to this factor, deciding how aggressive we are in enforcing the sparsity requirements. The stopping parameter $\bar{\delta}$, essentially linked to the accuracy of the solution given a certain convergence speed, is also configurable. For simplicity, we also set equal probabilities for PD-R, namely $p_{\mathcal{P}_i} = p_{\mathcal{P}}, \forall i$ and $p_{\mathcal{D}_j} = p_{\mathcal{D}}, \forall j$ and we show how the different choices affect the performance. We choose to randomize only over the data fidelity terms since the SARA sparsity prior is light from the computational perspective when compared to the data fidelity term, thus $p_{\mathcal{P}} = 1$ for all tests performed. Different strategies for the choice of probabilities, with values different for each block, are also possible. For example, setting a higher probability for the blocks containing low-frequency data will recover faster a coarse image. The details are incorporated into the solution through the lower probability updates of the high-frequency data. An overview of all the parameters used for defining the optimization task and for configuring both ADMM and PD algorithms is presented in Appendix A, Tables A1 and A2, respectively.

We ran MORESANE with five major loops and a major loop gain 0.9. The loop gain inside MORESANE was set to 0.1. We use the model image to compare against the other methods. CS-CLEAN was run with two loop gain factors, $l_g = 0.1$ and 0.001. The results shown are the best of the two. We compare against the model image convolved with a Gaussian kernel associated with the main beam. We scale the resulting image to be closest to the true model image in the least-square sense. Additionally, we also present results with the main beam scaled by a factor b chosen such that the best SNR is achieved. This introduces a large advantage for CS-CLEAN when compared to the other algorithms. To avoid edge artefacts, both MORESANE and CS-CLEAN were configured to produce a padded double-sized image and only the centre was used for comparison.

For PD, PD-R and ADMM, the stopping criterion for the algorithms is composed of two criteria. We consider the constraints satisfied when the global residual norm is in the vicinity of the bound ϵ of the global ℓ_2 ball, namely below a threshold $\bar{\epsilon}$. This is equivalent to stopping if $\sum_{j=1}^{n_d} \|\boldsymbol{y}_j - \boldsymbol{\Phi}_j \boldsymbol{x}^{(t)}\|_2^2 \leq \bar{\epsilon}^2$. We set $\bar{\epsilon}^2 = (2M + 3\sqrt{4M})\sigma_\chi^2$, namely three standard deviations above the mean. The second criterion relates to the relative variation of the solution, measured by

$$\delta = \frac{\|\boldsymbol{x}^{(t)} - \boldsymbol{x}^{(t-1)}\|_2}{\|\boldsymbol{x}^{(t)}\|_2}. \qquad (31)$$

The iterations stop when the ℓ_2 ball constraints are satisfied and when the relative change in the solution norm is small, $\delta \leq \bar{\delta}$. The data fidelity requirements are explicitly enforced, ensuring that we are inside or very close to the feasible region. However, this does not guarantee the minimization of the ℓ_1 prior function. The algorithms should run until the relative variation of the solution is small between iterations. To better understand the behaviour of the algorithms, for most simulations, we perform tests over a fixed number of iterations without applying the stopping conditions above.

The stopping criterion for MORESANE and CS-CLEAN was set to be three standard deviations above the noise mean. This level was seldom reached by CS-CLEAN after the deconvolution, the algorithm seeming to stop because of the accumulation of false detections leading to the increase of the residual between iterations.

6.3 Results using random coverages

We begin by analysing the evolution of the SNR for the ADMM and PD algorithms in comparison with that produced by the previously proposed SDMM solver. Fig. 5 contains the SNR as a function of number of iterations for the three algorithms for the reconstruction of the M31 image from $M = 10N$, $5N$ and $2N$ visibilities. The two newly introduced algorithms have the same convergence rate as SDMM but have a much lower computational burden per iteration, especially the PD method. In these tests, all three method use the parameter $\kappa = 10^{-3}$, suggested also by Carrillo et al. (2014).

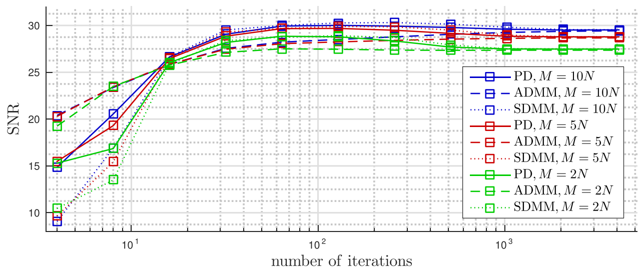


Figure 5. The evolution of the SNR for PD, ADMM and SDMM as a function of the number of iterations for the M31 test image. The configuration parameter, $\kappa = 10^{-3}$, is the same for ADMM, PD and SDMM. The number of visibilities M used is $10N$, $5N$ and $2N$. The input data are split into four blocks.

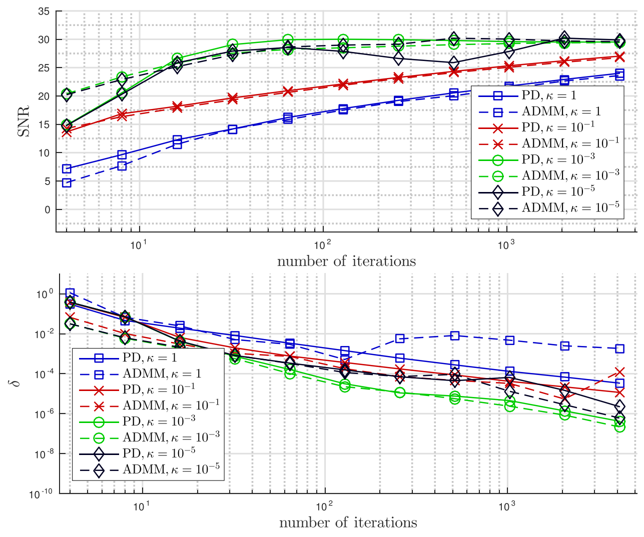


Figure 6. The reconstruction of the M31 image from $M = 10N$ visibilities. The input data are split into four blocks. (top) The evolution of the SNR for PD and ADMM as a function of the number of iterations for different values of the parameter κ. (bottom) The value of δ for both methods.

The reconstruction performance is comparable for the different test cases, the PD and ADMM obtaining the same reconstruction quality. Adding more data improves the reconstruction SNR by 2–3 dB because the noise is better averaged. However, note that the SNR gain stagnates slightly when more visibility data are added mainly because the holes in the frequency plane are still not covered. The problem remains very ill-posed with similar coverage. In a realistic situation, adding more data will also fill the coverage more and the SNR improvement will be larger. Since all three algorithms explicitly solve the same minimization problem, they should have similar behaviour for any other test case.

We continue by investigating the performance of the PD and ADMM algorithms as a function of the parameter κ in Figs 6–8 for the reconstruction of the M31, Cygnus A and galaxy cluster test images, respectively. The parameter κ serves as a normalized threshold and essentially governs the convergence speed. The values $\kappa = 10^{-3}$ to $\kappa = 10^{-5}$ generally produce good and consistent performance. This behaviour was also observed for similar tests, with smaller M. Larger values for κ reduce the convergence speed since they emphasize greatly the sparsity prior information at the expense of the data fidelity. The smaller values place less weight on the sparsity prior and, after an initial fast convergence due to the data fidelity term, typically require more iterations to minimize the ℓ_1

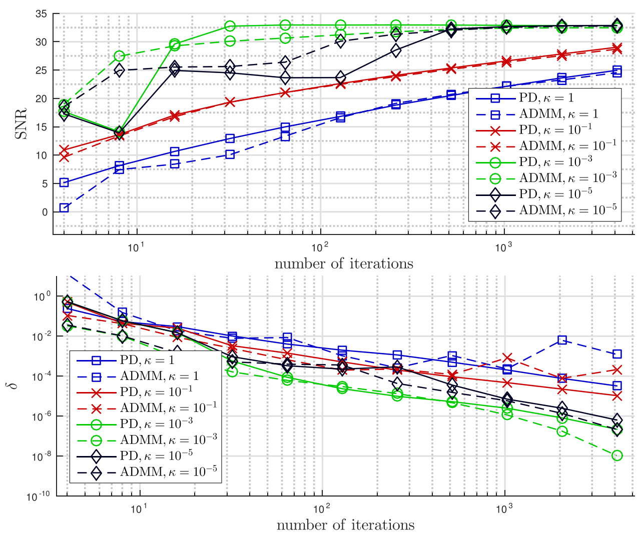


Figure 7. The reconstruction of the Cygnus A image from $M = N$ visibilities. The input data are split into four blocks. (top) The evolution of the SNR for PD and ADMM as a function of the number of iterations for different values of the parameter κ. (bottom) The value of δ for both methods.

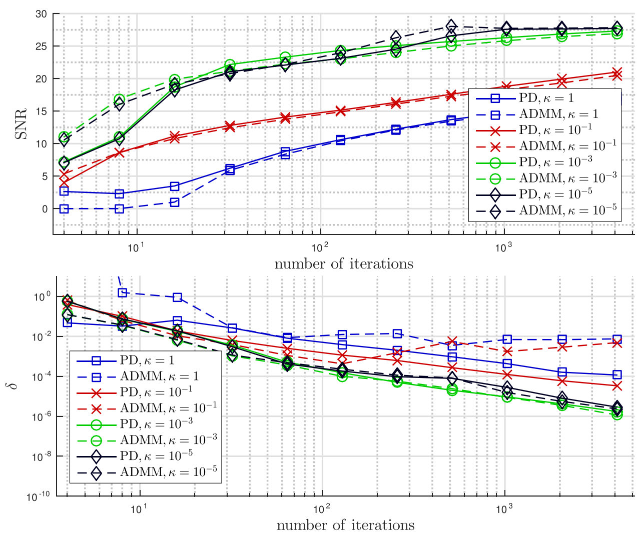


Figure 8. The reconstruction of the galaxy cluster image from $M = 2N$ visibilities. The input data are split into four blocks. (top) The evolution of the SNR for PD and ADMM as a function of the number of iterations for different values of the parameter κ. (bottom) The value of δ for both methods.

prior. The average variation of the solution norm δ is also reported since the stopping criterion is based on it. It links the convergence speed with the recovery performance. For the galaxy cluster, the tests exhibits slower convergence speed when compared to the M31 and Cygnus A tests. The values $\kappa = 10^{-3}$ and 10^{-5} produce similar behaviour. It should be also noted that the variation of the solution decreases smoothly until convergence and that ADMM shows a larger variability.

The convergence speed of the randomized algorithm, PD-R, is studied in Figs 9–11 for the M31, Cygnus A and galaxy cluster test images, with three choices for the data splitting. As expected, the convergence speed decreases when the probability of update

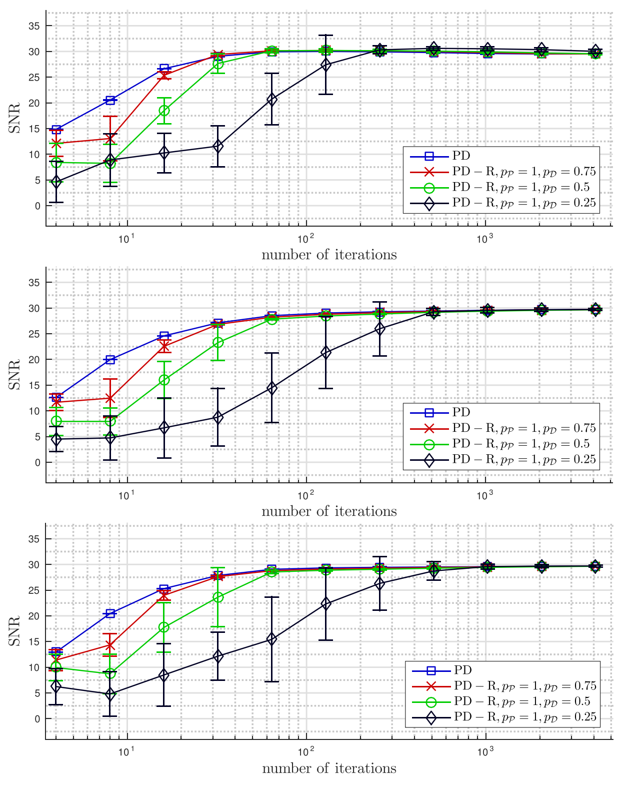


Figure 9. The SNR for the reconstruction of the M31 image from $M = 10N$ visibilities for the PD and PD-R algorithms with parameter $\kappa = 10^{-3}$. The algorithms split the input data into: (top) 4 blocks, (middle) 16 blocks, (bottom) 64 blocks.

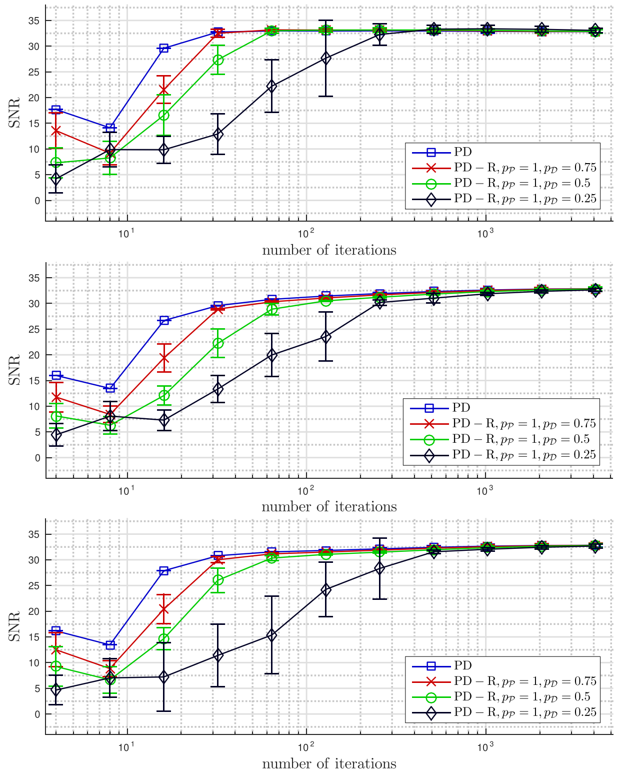


Figure 10. The SNR for the reconstruction of the Cygnus A image from $M = N$ visibilities for the PD and PD-R algorithms with parameter $\kappa = 10^{-3}$. The algorithms split the input data into: (top) 4 blocks, (middle) 16 blocks, (bottom) 64 blocks.

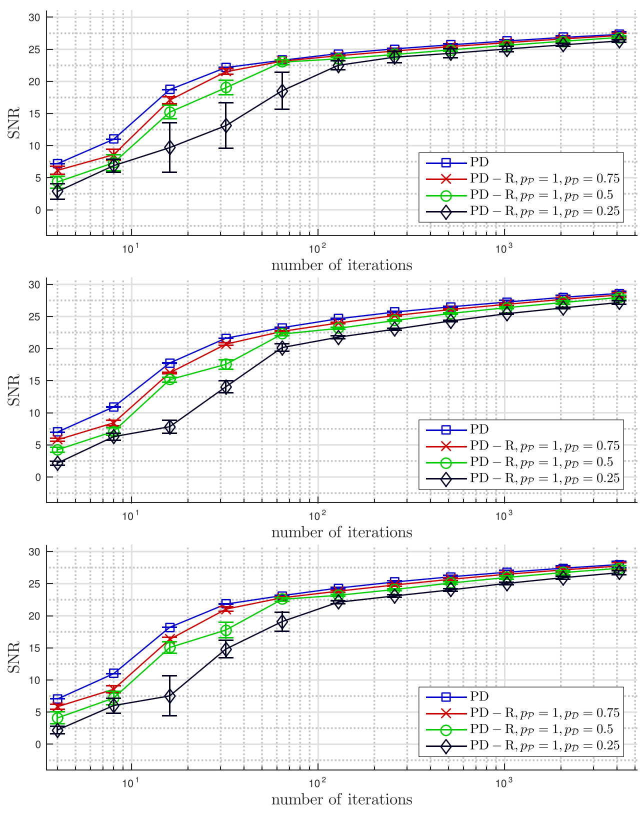


Figure 11. The SNR for the reconstruction of the galaxy cluster image from $M = 2N$ visibilities for the PD and PD-R algorithms with parameter $\kappa = 10^{-3}$. The algorithms split the input data into: (top) 4 blocks, (middle) 16 blocks, (bottom) 64 blocks.

$p_{\mathcal{D}}$ is lowered. The number of iterations required for convergence increases greatly for probabilities below 0.25. Similar behaviour is achieved for the reconstruction of the test images from a smaller number of measurements. Again, the convergence speed for the galaxy cluster test image is slower. There is also a very small decrease in the convergence speed for all tests when the data are split into a larger number of blocks. This is due to the fact that, in order to reach the same global ϵ, the resulting bounds imposed per block are more constraining and due to the fact that achieving a consensus between a larger number of blocks is more difficult.

Generally, the convergence speed decreases gradually as the probability $p_{\mathcal{D}}$ gets lower, PD-R remaining competitive and able to achieve good complexity as can be seen in Fig. 12. Here, we exemplify the performance in more detail when using the 64 blocks with parameter $\kappa = 10^{-3}$, the stopping threshold $\bar{\delta} = 10^{-4}$ and the ℓ_2 ball stopping threshold $\bar{\epsilon}^2 = (2M + 3\sqrt{4M})\sigma_\chi^2$. Our tests show that the total number of iterations performed is roughly inversely proportional to the probability $p_{\mathcal{D}}$. Additionally, we provide a basic estimate of the overall global complexity given the data from Table 1 and the number of iterations required. We only take into account the computationally heaviest operations, the FFT and the operations involving the data fidelity terms. The computations involving the sparsity priors are performed in parallel with the data fidelity computations and are much lighter. Since the analysis is made up to a scaling factor, for better consistency, we normalized the complexity of PD-R with respect to that of the PD.

The total complexity of PD-R remains similar to that of the non-randomized PD which makes PD-R extremely attractive. Generally, if the main computational bottleneck is due to the data term and not

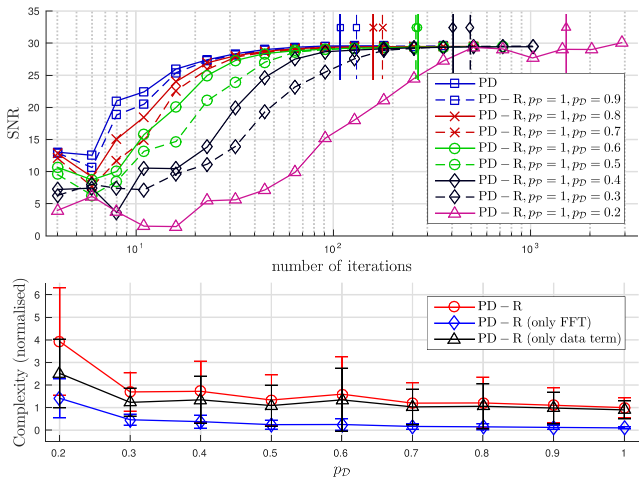


Figure 12. (top) The evolution of the SNR for PD-R for different probabilities for the reconstruction of the M31 test image from $M = 10N$ measurements. The average number of iterations performed for $\kappa = 10^{-3}$, $\bar{\delta} = 10^{-4}$ and $\bar{\epsilon}^2 = \left(2M + 3\sqrt{4M}\right)\sigma_\chi^2$ is marked by a vertical line. (bottom) The total complexity of PD-R and the parts of its total complexity due to the FFT and the data term computations, all normalized with respect to the average total complexity of PD. The visibilities are split into 64 equal size blocks.

to the FFT computations, it is expected that the total complexity of PD-R will remain comparable to that of the non-randomized PD. This is of great importance since, for a very large number of visibilities when the data does not fit in memory on the processing nodes, PD-R may be the only feasible alternative. When a more accurate stopping criterion is used, either with a smaller $\bar{\epsilon}_j$ or relative variation of the solution $\bar{\delta}$, the randomized algorithms start to require increasingly more iterations to converge and their relative complexity grows. Randomization over the sparsity bases is also possible but, due to the low computational burden of the priors we use, it is not of interest herein. However, randomization over the prior functions can become an important feature when computationally heavier priors are used or when the images to be reconstructed are very large.

6.4 Results with the VLA and SKA coverages

In Fig. 13, we present the SNR evolution as a function of the number of iterations for the PD and ADMM algorithms for the reconstruction of the Cygnus A and galaxy cluster images using the VLA coverage, and of the W28 supernova remnant test image using the SKA coverage. The visibilities are split into 64 equal size blocks and the parameter $\kappa = 10^{-5}$. We also overlay on the figures the SNR achieved using CS-CLEAN and MORESANE with the different types of weighting.

The dirty images produced using natural weighting for the same tests are presented in Fig. 14. For all three test cases, we showcase the reconstructed images, the reconstruction error images and the dirty residual images in Figs 15–17. We present the naturally weighted residual images for all methods even when they perform the deconvolution using a different weighting. Since any other type of weighting essentially biases the data and decreases the sensitivity of the reconstruction, this is the more natural choice of visualizing the remaining information in the residual image. Although both CS-CLEAN and MORESANE generally achieve better reconstruction for other weighting types, we present the naturally weighted dirty

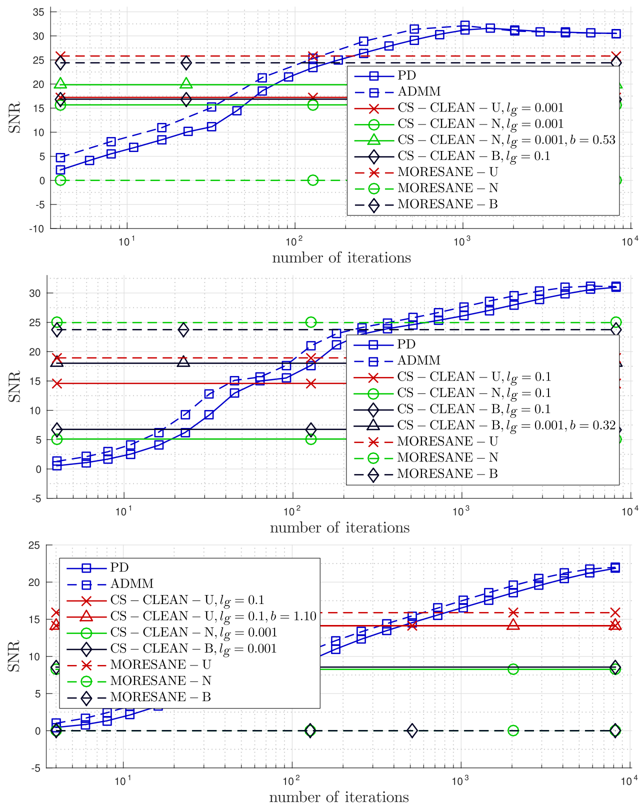


Figure 13. The SNR achieved by the algorithms studied for the reconstruction of (from top to bottom) the Cygnus A and the galaxy cluster images using the VLA coverage, and of the W28 supernova remnant image using the SKA coverage. For the PD and ADMM algorithms, we report the evolution of the SNR as a function of the iteration number. They use $\kappa = 10^{-5}$ and the data split into 64 equal-size blocks. The horizontal lines represent the final SNR achieved using CS-CLEAN and MORESANE.

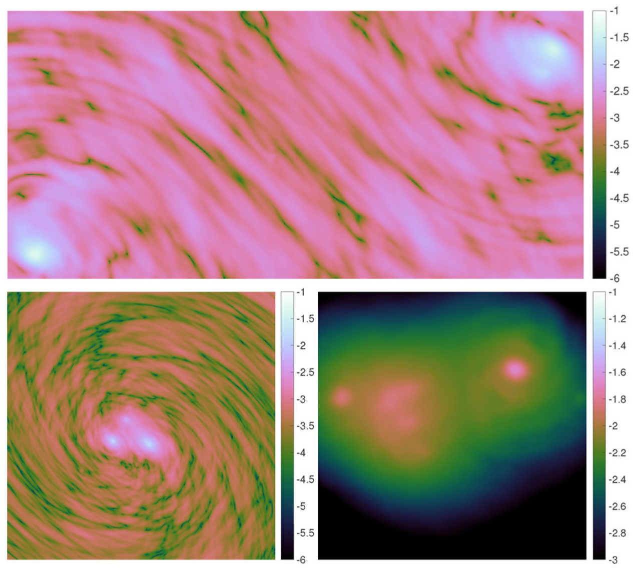


Figure 14. The log scale absolute value of the dirty images using natural weighting corresponding to (top) the Cygnus A and (bottom, left) the galaxy cluster test images, using the VLA coverage, and to (bottom, right) the W28 supernova remnant test image using the SKA coverage.

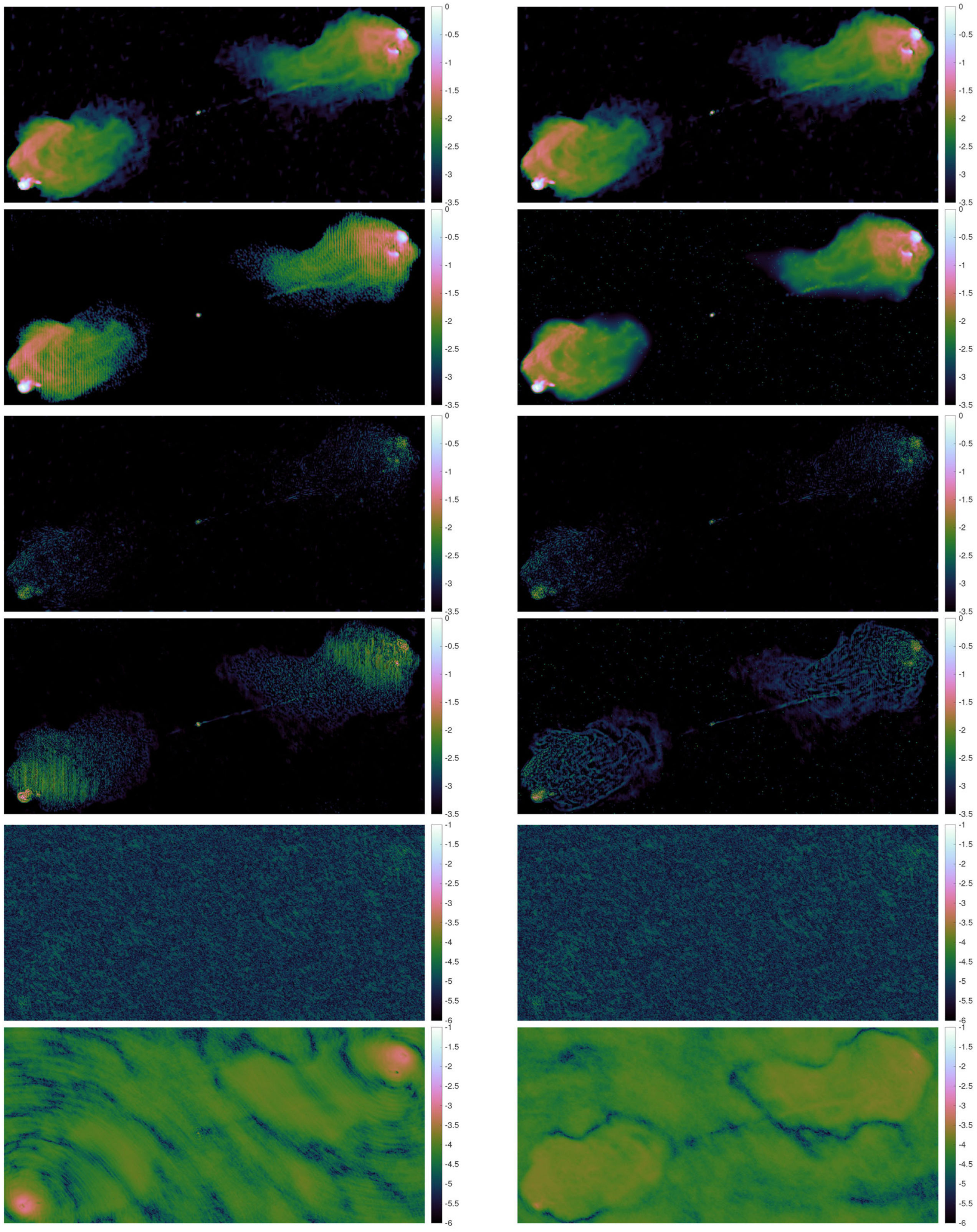


Figure 15. (top four images) Log scale reconstructed images; (middle four images) log scale of the absolute value of the estimation errors; (bottom four images) log scale of the absolute value of the naturally weighted residual images, for the 477 × 1025 Cygnus A test image using the VLA coverage. For each group, the algorithms are: (top left) PD with the reconstruction SNR = 30.51 dB and the corresponding DR = 108 620; (top right) ADMM with the reconstruction SNR = 30.52 dB and DR = 107 050; (bottom left) CS-CLEAN-N with $l_g = 0.001$ and $b = 0.53$ with the reconstruction SNR = 19.95 dB and DR = 10 773; (bottom right) MORESANE-U with the reconstruction SNR = 25.82 dB and DR = 11 661. The images correspond to the best results obtained by all algorithms as presented in Fig. 13.

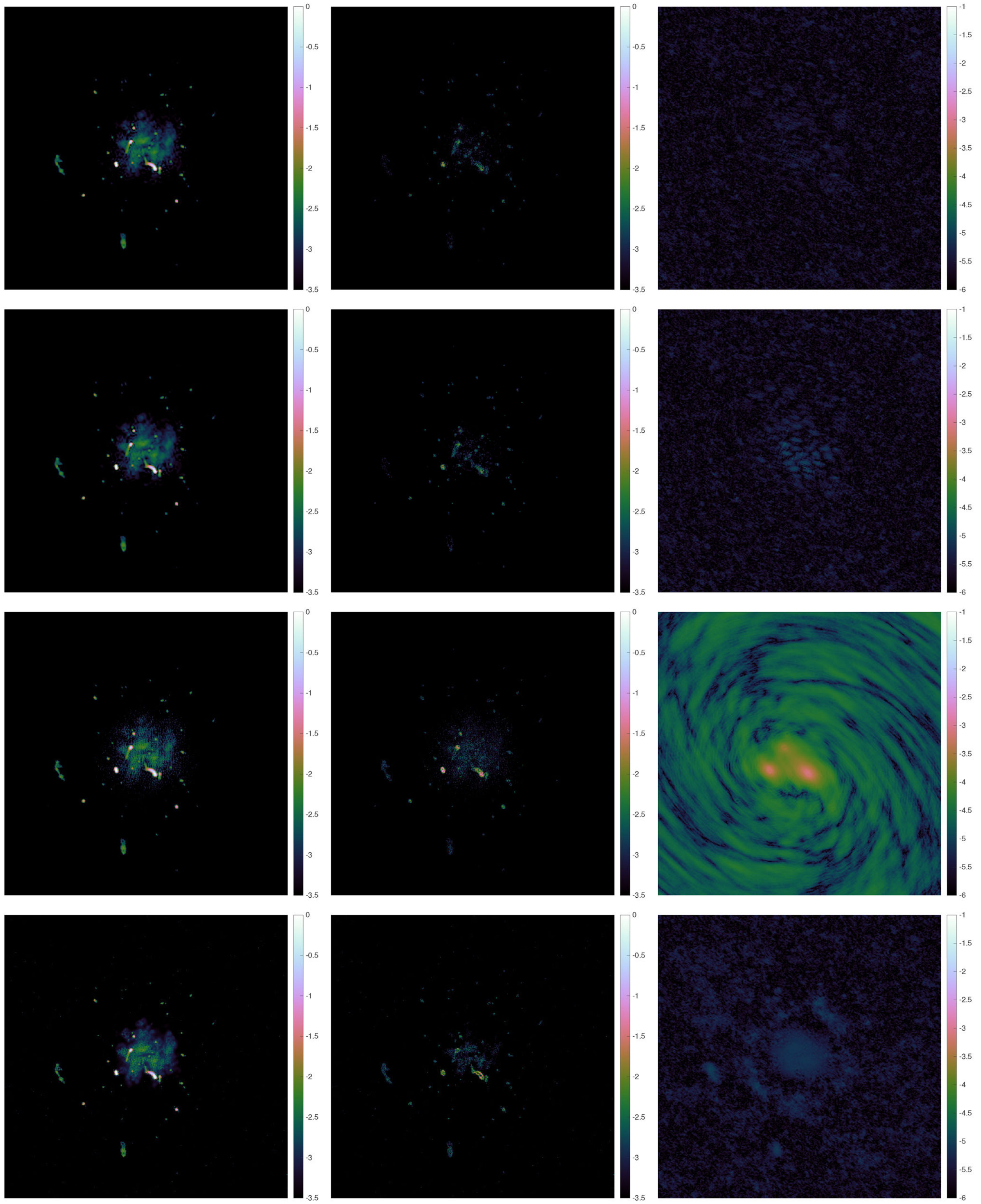


Figure 16. (left to right) The reconstructed images, absolute value of the estimation errors, and absolute value of the naturally weighted residual images, all in log scale, for the 512 × 512 galaxy cluster test image using the VLA coverage. The algorithms are: (from top to bottom) PD having the reconstruction SNR = 30.98 dB and the corresponding DR = 475 300; ADMM having the reconstruction SNR = 31.08 dB and DR = 432 070; CS-CLEAN-N with $l_g = 0.001$ and $b = 0.32$ having the reconstruction SNR = 18.03 dB and DR = 21 884; MORESANE-N having the reconstruction SNR = 24.96 dB and DR = 351 850. The images correspond to the best results obtained by all algorithms as presented in Fig. 13.

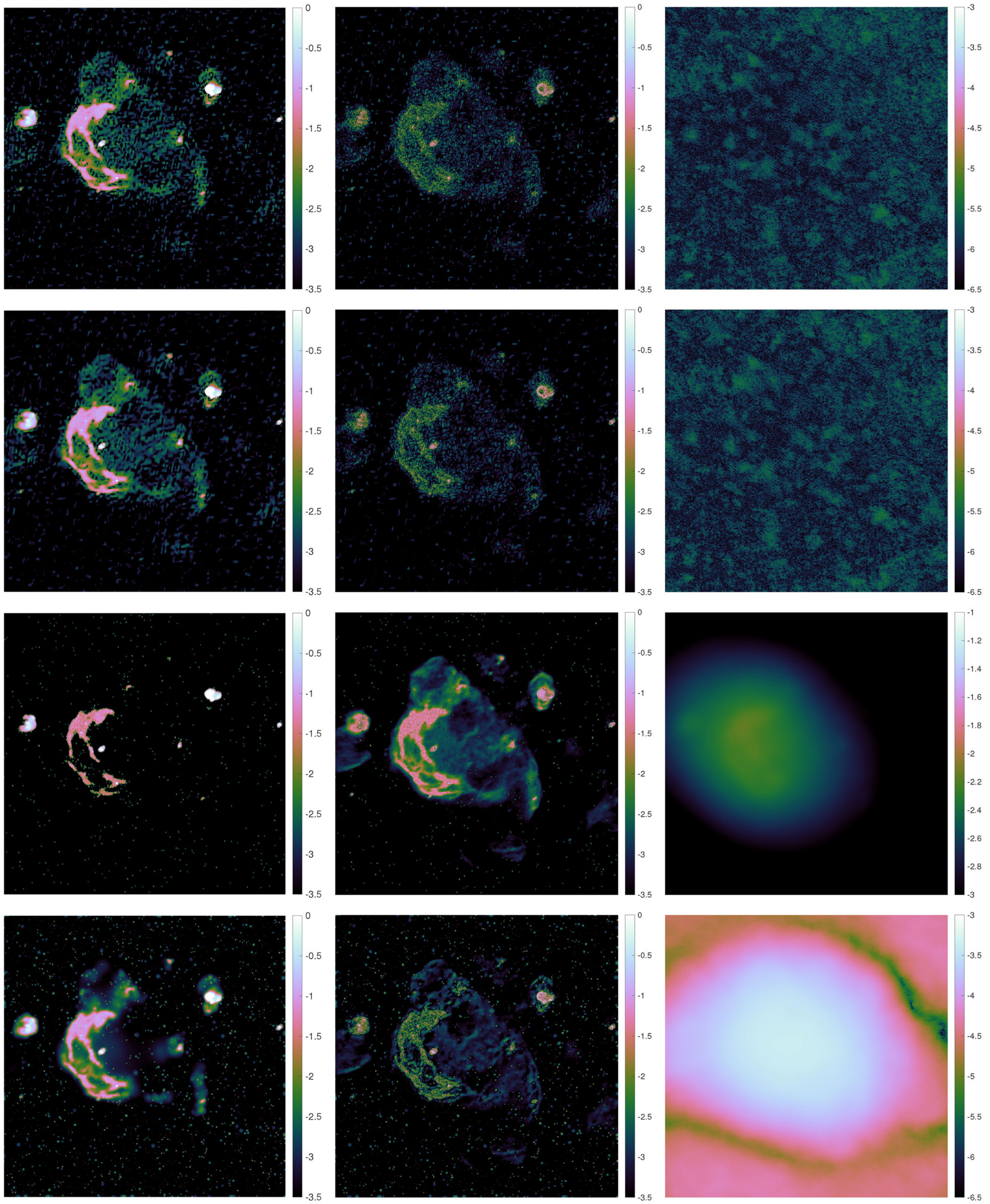


Figure 17. (left to right) The reconstructed images, absolute value of the estimation errors, and absolute value of the naturally weighted residual images, all in log scale, for the 1024 × 1024 W28 test image using the SKA coverage. The algorithms are: (from top to bottom) PD having the reconstruction SNR = 21.86 dB and the corresponding DR = 737 720; ADMM having the reconstruction SNR = 21.99 dB and DR = 735 620; CS-CLEAN-U with $l_g = 0.1$ and $b = 1.1$ having the reconstruction SNR = 14.14 dB and DR = 515; MORESANE-B having the reconstruction SNR = 15.89 dB and DR = 10 990. The images correspond to the best results obtained by all algorithms as presented in Fig. 13. Note that the scale for the residual image of CS-CLEAN-U is in the same range as the dirty image presented in Fig. 14, while for PD, ADMM and MORESANE, the scale of the residual image is below that.

residual, since it represents an unbiased estimation of the remaining structures.

For the reconstruction of the Cygnus A and galaxy cluster images, the methods developed herein outperform MORESANE, using the best performing type of weighting, by approximately 5 dB. Comparing against CS-CLEAN with the best weighting and beam size *b*, the SNR is around 10 dB in favour of the reconstruction performed by the PD and ADMM methods. Visually, both CS-CLEAN and MORESANE fail to recover properly the jet present in the Cygnus A image, while for PD and ADMM, it is clearly visible. It should be noted that the residual images show also very little structure for PD and ADMM while CS-CLEAN and MORESANE still allow for a more structured residual image. This is partially due to the biasing of the data when the uniform and Briggs weighting is performed. PD and ADMM also achieve a better reconstruction of the galaxy cluster image. They are able to better estimate the three bright sources in the centre of the image. They are, however, slower to converge if compared to the recovery of the Cygnus A image. MORESANE-N also performs well for this test image and is able to produce a relatively smoother residual image in comparison to the Cygnus A case. Note also that the performance of both CS-CLEAN and MORESANE is inconsistent and varies greatly with the weighting type.

The last test is performed for the reconstruction of the W28 supernova remnant image using the SKA coverage. In this case, the coverage is dominated by the low-frequency points and lowers the convergence speed of both PD and ADMM algorithms. Both PD and ADMM achieve good SNR, again around 5 dB over that reached by MORESANE. CS-CLEAN is 2 dB worse than MORESANE and is only able to recover the brightest sources as can be seen in Fig. 17. Again, both of our methods are able to recover more of the faint regions surrounding the bright sources. The dirty residual images show less structure for the methods developed herein since they work directly with the naturally weighted visibilities. Note that in Fig. 17, in order to achieve a better visualization, the scale of the dirty residual images for CS-CLEAN is different than that of the other methods. Also, the performance of both CS-CLEAN and MORESANE is again very inconsistent and varies greatly with the weighting type.

Both PD and ADMM methods show decreased convergence speed for the recovery of the galaxy cluster and W28 supernova remnant images. A future study should, possibly by using generalized proximity operators (Pesquet & Repetti 2015), address the acceleration of the convergence which is influenced by the relative distribution of the visibilities in frequency. Coverages dominated by low-frequency points, like the SKA one, generally produce slower convergence speed. Furthermore, if a faster convergence is achieved, a reweighing ℓ_1 approach becomes more attractive and should increase the reconstruction quality significantly.

7 CONCLUSIONS

We proposed two algorithmic frameworks based on ADMM and PD approaches for solving the RI imaging problem. Both methods are highly parallelizable and allow for an efficient distributed implementation which is fundamental in the context of the high dimensionality problems associated with the future SKA radio telescope. The structure of ADMM is sub-iterative, which, for much heavier priors than the ones used herein, may become a bottleneck. The PD algorithm achieves greater flexibility, in terms of memory requirements and computational burden per iteration, by using full splitting and randomized updates. Through the analogy between the CLEAN major-minor loop and an FB iteration, both methods can be understood as being composed of sophisticated CLEAN-like iterations running in parallel in multiple data, prior and image spaces.

The reconstruction quality for both ADMM and PD methods is similar to that of SDMM. The computational burden is much lower. Experimental results with realistic coverages show impressive performance in terms of parallelization and distribution, suggesting scalability to extremely large data sets. We give insight into the performance as a function of the configuration parameters and provide a parameter setup, with the normalized soft-thresholding values between 10^{-3} and 10^{-5}, that produce consistently stable results for a broad range of tests. The solution to the optimization problem solved herein was shown to greatly outperform the standard methods in RI which further motivates the use of our methods. Our tests also confirm the reconstruction quality in the high-dynamic range regime.

Our MATLAB code is available online on GitHub, http://basp-group.github.io/pd-and-admm-for-ri/. In the near future, we intend to provide an efficient implementation, using the MPI communication library, for a distributed computing infrastructure. This will be included in the PURIFY C++ package, which currently only implements a sequential version of SDMM. The acceleration of the algorithms for coverages dominated by low-frequency points will also be investigated, by leveraging a generalized proximal operator. Additionally, recent results suggest that the conditions for convergence for the randomized PD can be relaxed, which would accelerate the convergence speed making these methods to be even more competitive. We also envisage to use the same type of framework to image in the presence of DDEs, such as the w component, as well as to jointly solve the calibration and image reconstruction problems.

ACKNOWLEDGEMENTS

This work was supported by the UK Engineering and Physical Sciences Research Council (EPSRC, grants EP/M011089/1 and EP/M008843/1) and UK Science and Technology Facilities Council (STFC, grant ST/M00113X/1), as well as by the Swiss National Science Foundation (SNSF) under grant 200020-146594. We would like to thank Federica Govoni and Matteo Murgia for providing the simulated galaxy cluster image.

REFERENCES

Ables J. G., 1974, A&AS, 15, 686
Bauschke H. H., Combettes P. L., 2011, Convex Analysis and Monotone Operator Theory in Hilbert Spaces. Springer-Verlag, New York
Beck A., Teboulle M., 2009, SIAM J. Img. Sci., 2, 183
Bertsekas D. P., 1982, Constrained Optimization and Lagrange Multiplier Methods. Academic Press, New York
Bhatnagar S., Cornwell T. J., 2004, A&A, 426, 747
Boyd S., Parikh N., Chu E., Peleato B., Eckstein J., 2011, Found. Trends Mach. Learn., 3, 1
Boţ R. I., Hendrich C., 2013, SIAM J. Opt., 23, 2541
Broekema P. C., van Nieuwpoort R. V., Bal H. E., 2015, J. Instrum., 10, C07004
Brogan C. L., Gelfand J. D., Gaensler B. M., Kassim N. E., Lazio T. J. W., 2006, ApJ, 639, L25
Calamai P. H., Moré J. J., 1987, Math. Program., 39, 93
Candès E. J., 2006, Proc. Int. Congress Math., Vol. 3, Compressive Sampling. Eur. Math. Soc., Madrid, p. 1433
Candès E. J., Wakin M. B., Boyd S. P., 2008, J. Fourier Anal. Appl., 14, 877
Carrillo R. E., McEwen J. D., Wiaux Y., 2012, MNRAS, 426, 1223

Carrillo R. E., McEwen J. D., Ville D. V. D., Thiran J.-P., Wiaux Y., 2013, IEEE Signal Process. Lett., 20, 591
Carrillo R. E., McEwen J. D., Wiaux Y., 2014, MNRAS, 439, 3591
Carrillo R. E., Kartik V., Thiran J.-P., Wiaux Y., 2015, A Scalable Algorithm for Radio-interferometric Imaging. Sig. Proc. Adapt. Sparse Struct. Repr. Cambridge Univ. Press, Cambridge
Cohen A., Daubechies I., Vial P., 1993, Appl. Comput. Harmon. Anal., 1, 54
Combettes P. L., Pesquet J.-C., 2007a, IEEE J. Sel. Top. Signal Process., 1, 564
Combettes P. L., Pesquet J.-C., 2007b, SIAM J. Opt., 18, 1351
Combettes P. L., Pesquet J.-C., 2011, in Bauschke H. H., Burachik R. S., Combettes P. L., Elser V., Luke D. R., Wolkowicz H., eds, Fixed-Point Algorithms for Inverse Problems in Science and Engineering. Springer-Verlag, New York, p. 185
Combettes P. L., Pesquet J.-C., 2012, Set-Valued Var. Anal., 20, 307
Combettes P. L., Pesquet J.-C., 2015, SIAM J. Opt., 25, 1221
Combettes P. L., Dũng D., Vũ B. C., 2011, J. Math. Anal. Appl., 380, 680
Condat L., 2013, J. Opt. Theory Appl., 158, 460
Cooley J. W., Tukey J. W., 1965, Math. Comput., 19, 297
Cornwell T. J., 2008, IEEE J. Sel. Top. Signal Process., 2, 793
Cornwell T. J., Evans K. F., 1985, A&A, 143, 77
Dabbech A., Ferrari C., Mary D., Slezak E., Smirnov O., Kenyon J. S., 2015, A&A, 576, A7
Daubechies I., Sweldens W., 1998, J. Fourier Anal. Appl., 4, 247
Daubechies I., Defrise M., De Mol C., 2004, Commun. Pure Appl. Math., 57, 1413
Daubechies I., DeVore R., Fornasier M., Güntürk C. S., 2010, Commun. Pure Appl. Math., 63, 1
Dewdney P., Hall P., Schilizzi R. T., Lazio T. J. L. W., 2009, Proc. IEEE, 97, 1482
Donoho D. L., 2006, IEEE Trans. Inf. Theory, 52, 1289
Elad M., Milanfar P., Rubinstein R., 2007, Inverse Probl., 23, 947
Ferrari A., Mary D., Flamary R., Richard C., 2014, Distributed Image Reconstruction for Very Large Arrays in Radio Astronomy, IEEE 8th Sensor Array Multich. Sig. Proc. Workshop (SAM), 1507.00501, A Coruna, p. 389
Fessler J., Sutton B., 2003, IEEE Trans. Signal Process., 51, 560
Fornasier M., Rauhut H., 2011, Handbook of Mathematical Methods in Imaging. Springer-Verlag, New York
Garsden H. et al., 2015, A&A, 575, A90
Gull S. F., Daniell G. J., 1978, Nature, 272, 686
Hardy S. J., 2013, A&A, 557, A134
Högbom J. A., 1974, A&A, 15, 417
Komodakis N., Pesquet J.-C., 2015, IEEE Signal Process. Mag., 32, 31
Li F., Cornwell T. J., de Hoog F., 2011, A&A, A31, 528
Mallat S., 2008, A Wavelet Tour of Signal Processing, 3rd edn. Academic Press, New York
Mallat S., Zhang Z., 1993, IEEE Trans. Signal Process., 41, 3397
McEwen J. D., Wiaux Y., 2011, MNRAS, 413, 1318
Moreau J. J., 1965, Bull. Soc. Math. France, 93, 273
Murgia M., Govoni F., Feretti L., Giovannini G., Dallacasa D., Fanti R., Taylor G. B., Dolag K., 2004, A&A, 424, 429
Offringa A. R. et al., 2014, MNRAS, 444, 606
Pesquet J.-C., Repetti A., 2015, J. Nonlinear Convex Anal., 16, 2453
Pesquet J.-C., Pustelnik N., 2012, Pac. Journal Optimization, 8, 273
Rau U., Bhatnagar S., Voronkov M. A., Cornwell T. J., 2009, Proc. IEEE, 97, 1472
Schwab F. R., 1984, AJ, 89, 1076
Schwarz U. J., 1978, A&A, 65, 345
Setzer S., Steidl G., Teuber T., 2010, J. Vis. Comun. Image Represent., 21, 193
Thompson A. R., Moran J. M., Swenson G. W., 2001, Interferometry and Synthesis in Radio Astronomy. Wiley-Interscience, New York
van Haarlem M. P. et al., 2013, A&A, 556, 1
Vũ B. C., 2013, Adv. Comput. Math., 38, 667
Wenger S., Magnor M., Pihlströsm Y., Bhatnagar S., Rau U., 2010, PASP, 122, 1367
Wiaux Y., Jacques L., Puy G., Scaife A. M. M., Vandergheynst P., 2009a, MNRAS, 395, 1733
Wiaux Y., Puy G., Boursier Y., Vandergheynst P., 2009b, MNRAS, 400, 1029
Wiaux Y., Puy G., Vandergheynst P., 2010, MNRAS, 402, 2626
Wijnholds S. J., van der Veen A.-J., de Stefani F., la Rosa E., Farina A., 2014, IEEE Int. Conf. Acous., Speech Sig. Proc., Signal Processing Challenges for Radio Astronomical Arrays, Florence, p. 5382
Wolz L., McEwen J. D., Abdalla F. B., Carrillo R. E., Wiaux Y., 2013, MNRAS, 463, 1993
Yang J., Zhang Y., 2011, SIAM J. Sci. Comput., 33, 250
Yatawatta S., 2015, MNRAS, 449, 4506

APPENDIX A: PARAMETER OVERVIEW

An overview of the parameters used to define the minimization problems is presented in Table A1. The configuration parameters for the algorithms are presented in Table A2.

Table A1. Overview of the parameters for defining the optimization problem (12).

Optimization problem definition	
$\boldsymbol{\Psi}_i$	the n_b wavelet bases in which the signal is considered sparse; other priors can be incorporated as well by redefining the functions l_i and their associated proximity operators
n_b	the number of data blocks generally linked to the computing infrastructure
$\mathcal{B}_j$	the ℓ_2 balls imposing data fidelity; they are linked to the modality in which the data are split into blocks y_j
ϵ_j	the size of the ℓ_2 balls defining the data fidelity; they are linked to the statistics of the noise; herein ϵ_j are set based the χ^2 distribution associated with the noise

Table A2. The configuration parameters for the ADMM (top) and PD (bottom) algorithms.

Algorithm 1 (ADMM)	
$\kappa > 0$	configurable; influences the convergence speed
$\bar{\delta} \leq 10^{-3}$, $\bar{\epsilon}_j$	configurable; stopping criteria; linked to the accuracy of the desired solution
$\bar{\delta}_{\bar{f}} \leq 10^{-3}$, $n_{\bar{f}}$	configurable; sub-iteration stopping criteria; linked to the accuracy of the desired solution
$\varrho = 0.9$, $\rho = \frac{1}{\\|\boldsymbol{\Phi}\\|_S^2}$	fixed; algorithm convergence parameters; need to satisfy (D1)
$\eta = \frac{1}{\\|\boldsymbol{\Psi}\\|_S^2}$	fixed; algorithm convergence parameter
Algorithm 2 (PD)	
$\kappa > 0$	configurable; influences the convergence speed
$\bar{\delta} \leq 10^{-3}$, $\bar{\epsilon}_j$	configurable; stopping criteria; linked to the accuracy of the desired solution
$p_{\mathcal{P}_i} > 0$, $p_{\mathcal{D}_j} > 0$	configurable; randomization probabilities; linked to the computing infrastructure
$\tau = 0.49$, $\varsigma = \frac{1}{\\|\boldsymbol{\Phi}\\|_S^2}$, $\sigma = \frac{1}{\\|\boldsymbol{\Psi}\\|_S^2}$	fixed; algorithm convergence parameters; need to satisfy (D3)

APPENDIX B: SDMM ALGORITHM

The structure of SDMM, solving the specific RI problem (16), is presented for completeness in Algorithm 3.

Algorithm 3 SDMM.

```
1:  given $\boldsymbol{x}^{(0)}, \tilde{\boldsymbol{r}}_j^{(0)}, \bar{\boldsymbol{r}}_j^{(0)}, \check{\boldsymbol{r}}_j^{(0)}, \tilde{\boldsymbol{s}}_j^{(0)}, \bar{\boldsymbol{s}}_j^{(0)}, \hat{\boldsymbol{s}}_i^{(0)}, \kappa$
2:  repeat for $t = 1, \ldots$
3:      $\tilde{\boldsymbol{b}}^{(t)} = \mathbf{FZ}\boldsymbol{x}^{(t-1)}$
4:      $\forall j \in \{1, \ldots, n_{\mathrm{d}}\}$ set
5:          $\boldsymbol{b}_j^{(t)} = \mathbf{M}_j \tilde{\boldsymbol{b}}^{(t)}$
6:      end
7:      run simultaneously
8:          $\forall j \in \{1, \ldots, n_{\mathrm{d}}\}$ distribute $\boldsymbol{b}_j^{(t)}$ and do in parallel
9:              $\tilde{\boldsymbol{r}}_j^{(t)} = \mathcal{P}_{\mathcal{B}_j}\left(\mathbf{G}_j \boldsymbol{b}_j^{(t)} + \tilde{\boldsymbol{s}}_j^{(t-1)}\right)$
10:             $\tilde{\boldsymbol{s}}_j^{(t)} = \tilde{\boldsymbol{s}}_j^{(t-1)} + \mathbf{G}_j \boldsymbol{b}_j^{(t)} - \tilde{\boldsymbol{r}}_j^{(t)}$
11:             $\tilde{\boldsymbol{q}}_j^{(t)} = \mathbf{G}_j^\dagger \left(\tilde{\boldsymbol{r}}_j^{(t)} - \tilde{\boldsymbol{s}}_j^{(t)}\right)$
12:         end and gather $\tilde{\boldsymbol{q}}_j^{(t)}$
13:         $\forall i \in \{1, \ldots, n_{\mathrm{b}}\}$ do in parallel
14:             $\bar{\boldsymbol{r}}_i^{(t)} = \mathcal{S}_{\kappa \|\boldsymbol{\Psi}\|_{\mathrm{S}}}\left(\boldsymbol{\Psi}_i^\dagger \boldsymbol{x}^{(t-1)} + \bar{\boldsymbol{s}}_i^{(t-1)}\right)$
15:             $\bar{\boldsymbol{s}}_i^{(t)} = \bar{\boldsymbol{s}}_i^{(t-1)} + \boldsymbol{\Psi}_i^\dagger \boldsymbol{x}^{(t-1)} - \bar{\boldsymbol{r}}_i^{(t)}$
16:             $\bar{\boldsymbol{q}}_i^{(t)} = \boldsymbol{\Psi}_i \left(\bar{\boldsymbol{r}}_i^{(t)} - \bar{\boldsymbol{s}}_i^{(t)}\right)$
17:         end
18:         do
19:             $\hat{\boldsymbol{r}}^{(t)} = \mathcal{P}_{\mathcal{C}}\left(\boldsymbol{x}^{(t-1)} + \hat{\boldsymbol{s}}^{(t-1)}\right)$
20:             $\hat{\boldsymbol{s}}^{(t)} = \hat{\boldsymbol{s}}^{(t-1)} + \boldsymbol{x}^{(t-1)} - \hat{\boldsymbol{r}}^{(t)}$
21:             $\hat{\boldsymbol{q}}^{(t)} = \hat{\boldsymbol{r}}^{(t)} - \hat{\boldsymbol{s}}^{(t)}$
22:         end
23:     end
24:     $\tilde{\boldsymbol{x}}^{(t)} = \hat{\boldsymbol{q}}^{(t)} + \frac{1}{\|\boldsymbol{\Phi}\|_{\mathrm{S}}^2} \mathbf{Z}^\dagger \mathbf{F}^\dagger \sum_{j=1}^{n_{\mathrm{d}}} \mathbf{M}_j^\dagger \tilde{\boldsymbol{q}}_j^{(t)} + \frac{1}{\|\boldsymbol{\Psi}\|_{\mathrm{S}}^2} \sum_{i=1}^{n_{\mathrm{b}}} \bar{\boldsymbol{q}}_i^{(t)}$
25:     $\boldsymbol{x}^{(t)} = \left(\frac{1}{\|\boldsymbol{\Phi}\|_{\mathrm{S}}^2} \sum_{j=1}^{n_{\mathrm{d}}} \boldsymbol{\Phi}_j^\dagger \boldsymbol{\Phi}_j + \frac{1}{\|\boldsymbol{\Psi}\|_{\mathrm{S}}^2} \sum_{i=1}^{n_{\mathrm{b}}} \boldsymbol{\Psi}_i \boldsymbol{\Psi}_i^\dagger + \boldsymbol{I}\right)^{-1} \tilde{\boldsymbol{x}}^{(t)}$
26: until convergence
```

APPENDIX C: CONVEX OPTIMIZATION TOOLS

Definition 1. The proximity operator (Moreau 1965) applied to any lower semicontinuous andproper convex function g is defined as

$$\operatorname{prox}_g(\boldsymbol{z}) \triangleq \underset{\bar{\boldsymbol{z}}}{\operatorname{argmin}}\, g(\bar{\boldsymbol{z}}) + \frac{1}{2}\|\boldsymbol{z} - \bar{\boldsymbol{z}}\|_2^2. \tag{C1}$$

Definition 2. The indicator function $\iota_{\mathcal{C}}$ of any set $\mathcal{C}$ is defined as

$$(\forall \boldsymbol{z}) \qquad \iota_{\mathcal{C}}(\boldsymbol{z}) \triangleq \begin{cases} 0 & \boldsymbol{z} \in \mathcal{C} \\ +\infty & \boldsymbol{z} \notin \mathcal{C}. \end{cases} \tag{C2}$$

In convex optimization, it allows the use of an equivalent formulation for constrained problems by replacing the explicit convex constraints with the indicator function of the convex set $\mathcal{C}$ defined by the constraints. Its use makes the minimization task easier to tackle by general convex optimization solvers.

Definition 3. The Legendre–Fenchel conjugate function g^* of a function g is

$$(\forall \boldsymbol{v}) \qquad g^*(\boldsymbol{v}) \triangleq \sup_{\boldsymbol{z}} \boldsymbol{z}^\dagger \boldsymbol{v} - g(\boldsymbol{z}). \tag{C3}$$

Property 1. (Moreau decomposition) The Moreau decomposition links the proximity operator of a lower semicontinuous and proper convex function g to that of its Legendre–Fenchel conjugate g^* as

$$(\forall \boldsymbol{z}) \;\; \boldsymbol{z} = \operatorname{prox}_{\alpha g}(\boldsymbol{z}) + \alpha \operatorname{prox}_{\alpha^{-1} g^*}(\alpha^{-1}\boldsymbol{z}), \;\; 0 < \alpha < \infty. \tag{C4}$$

APPENDIX D: ALGORITHM CONVERGENCE

D1 Alternating direction method of multipliers

The convergence of Algorithm 1 is achieved through a careful choice of the parameters ρ and ϱ. The algorithm converges for any choice of the Lagrange parameter μ satisfying $\mu > 0$. This imposes the same constraint on κ. For the convergence of the dual FB sub-iterations, the update parameter η should satisfy $0 < \eta < \frac{2}{\|\boldsymbol{\Psi}\|_{\mathrm{S}}^2}$.

Assuming that the measurement operator $\boldsymbol{\Phi}$ is full column rank and that convergence has been reached with the dual FB sub-iterations, the convergence for the whole algorithm is achieved in terms of both objective function $\bar{f}(\boldsymbol{x}) + \bar{h}(\boldsymbol{\Phi x})$ and iterates $\boldsymbol{x}^{(t)}$, $\boldsymbol{r}_j^{(t)}$ and, $\boldsymbol{s}_j^{(t)}$ (Boyd et al. 2011; Komodakis & Pesquet 2015). It requires that

$$\rho\|\boldsymbol{\Phi}\|_{\mathrm{S}}^2 + \varrho < 2, \tag{D1}$$

with $\|\boldsymbol{\Phi}\|_{\mathrm{S}}$ being the spectral norm of the measurement operator and the parameters ρ and ϱ being the update step used for the proximal splitting and the gradient ascent step, respectively.

In practice, however, the RI imaging problem is very ill-conditioned and the operator $\boldsymbol{\Phi}$ is typically not full rank. Under these relaxed conditions, the convergence is guaranteed only with respect to the objective function and the multipliers $\boldsymbol{s}_j^{(t)}$, without any guarantees for the iterates $\boldsymbol{x}^{(t)}$ and $\boldsymbol{r}^{(t)}$ (Boyd et al. 2011). A possible way to improve this is to replace $\bar{h}$ with an augmented function $\tilde{h}$,

$$\tilde{h}\left(\begin{bmatrix} \boldsymbol{\Phi} \\ \boldsymbol{\Gamma} \end{bmatrix} \boldsymbol{x}\right) = \bar{h}(\boldsymbol{\Phi x}) + 0(\boldsymbol{\Gamma x}), \tag{D2}$$

where 0 represents the null function, zero for any $\boldsymbol{x}$. Such a trick (Pesquet et al. 2012) replaces the measurement operator $\boldsymbol{\Phi}$ with the augmented operator representing the concatenation of both $\boldsymbol{\Phi}$ and $\boldsymbol{\Gamma}$. The new resulting operator is full rank for a proper choice of the matrix $\boldsymbol{\Gamma}$. In practice, Algorithm 1 produces reliable performance and we did not employ such a trick herein.

D2 PD algorithm

The variables $\boldsymbol{x}^{(t)}$, $\boldsymbol{v}_j^{(t)}$ and $\boldsymbol{u}_i^{(t)}$, $\forall i, j$, are guaranteed to converge to the solution of the PD problem (27) and (28) for a proper set of configuration parameters. The convergence, defined given two general preconditioning matrices $\mathbf{U}$ and $\mathbf{W}$, requires (Pesquet & Repetti 2015, Lemma 4.3) that

$$\|\mathbf{U}^{\frac{1}{2}}\mathbf{L}\mathbf{W}^{\frac{1}{2}}\|_{\mathrm{S}}^2 < 1, \tag{D3}$$

with the linear operator $\mathbf{L}$ being a concatenation of all the used operators, in our case a concatenation of both $\boldsymbol{\Psi}^\dagger$ and $\boldsymbol{\Phi}$. By choosing diagonal preconditioning matrices, with the config parameters τ, $\sigma_i = \sigma$ and $\varsigma_j = \varsigma$, $\forall i, j$, on the adequate diagonal locations, the conditions from (D3) can be restated explicitly for Algorithm 2 as

$$\left\| \begin{bmatrix} \sigma \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \varsigma \mathbf{I} \end{bmatrix}^{\frac{1}{2}} \begin{bmatrix} \boldsymbol{\Psi}^\dagger \\ \boldsymbol{\Phi} \end{bmatrix} \left[\tau \mathbf{I}\right]^{\frac{1}{2}} \right\|_{\mathrm{S}}^2 \leq \tau\sigma \left\|\boldsymbol{\Psi}^\dagger\right\|_{\mathrm{S}}^2 + \tau\varsigma \left\|\boldsymbol{\Phi}\right\|_{\mathrm{S}}^2 < 1, \tag{D4}$$

with the use of the triangle and Cauchy–Schwarz inequalities and with the diagonal matrices $\mathbf{I}$ of a proper dimension. It should be noted that this formulation does not limit the use to only two parameters σ and ς. However, having more independent update steps scales poorly due to the increasing difference between the resulting bound, computed similarly to (D4), and the requirements (D3). This translates to having increasingly small values for the update steps, the more independent parameters we employ, with the convergence speed slowing down considerably in such situation. It is also required that the relaxation parameter is chosen such that $0 < \lambda \leq 1$. The additional parameter $\gamma > 0$ imposes that $\kappa > 0$ as well.

For the randomized setup, the same parameters satisfying (D3) suffice, granted that the probabilities of update $p_{\mathcal{P}_i}$ and $p_{\mathcal{D}_j}$ are non-zero and the activated variables are drawn in an independent and identical manner along the iterations.

This paper has been typeset from a TEX/LATEX file prepared by the author.