

The R2D2 deep neural network series paradigm for fast precision imaging in radio astronomy

AMIR AGHABIGLOU ¹, CHUNG SAN CHU ¹, ARWA DABBECH ¹ AND YVES WIAUX ¹

¹*Institute of Sensors, Signals and Systems, Heriot-Watt University, Edinburgh EH14 4AS, United Kingdom*

ABSTRACT

Radio-interferometric (RI) imaging entails solving high-resolution high-dynamic range inverse problems from large data volumes. Recent image reconstruction techniques grounded in optimization theory have demonstrated remarkable capability for imaging precision, well beyond CLEAN’s capability. These range from advanced proximal algorithms propelled by handcrafted regularization operators, such as the SARA family, to hybrid plug-and-play (PnP) algorithms propelled by learned regularization denoisers, such as AIRI. Optimization and PnP structures are however highly iterative, which hinders their ability to handle the extreme data sizes expected from future instruments. To address this scalability challenge, we introduce a novel deep-learning approach, dubbed “**R**esidual-to-**R**esidual **D**NN series for high-**D**ynamic range imaging”. R2D2’s reconstruction is formed as a series of residual images, iteratively estimated as outputs of Deep Neural Networks (DNNs) taking the previous iteration’s image estimate and associated data residual as inputs. It thus takes a hybrid structure between a PnP algorithm and a learned version of the matching pursuit algorithm that underpins CLEAN. We present a comprehensive study of our approach, featuring its multiple incarnations distinguished by their DNN architectures. We provide a detailed description of its training process, targeting a telescope-specific approach. R2D2’s capability to deliver high precision is demonstrated in simulation, across a variety of image and observation settings using the Very Large Array (VLA). Its reconstruction speed is also demonstrated: with only few iterations required to clean data residuals at dynamic ranges up to 10^5 , R2D2 opens the door to fast precision imaging. R2D2 codes are available in the [BASPLib](#) library on GitHub.

Keywords: Computational methods (1965) — Neural networks (1933) — Astronomy image processing (2306) — Aperture synthesis (53)

1. INTRODUCTION

Astronomical imaging is a cornerstone of our quest to deepen our understanding of the Universe. It allows us to decipher the origins of galaxies, explore planetary systems, and probe the fundamental laws governing the cosmos. Modern radio telescopes, such as the MeerKAT (Jonas 2016), the Low-Frequency Array (LOFAR; van Haarlem et al. 2013), and the Australian Square Kilometre Array Pathfinder (ASKAP; Hotan et al. 2021), have transformed the field, offering unprecedented resolution and sensitivity. However, these powerful instruments bring scalability and precision challenges. The data volumes they provide are vast and the expected

dynamic range of the radio images to be formed spans multiple orders of magnitude.

Aperture synthesis by radio interferometry captures incomplete Fourier information of the radio sky. Addressing the underlying ill-posed inverse problem calls for advanced image formation algorithms to deliver the science objectives. Five decades since its inception, the CLEAN algorithm (Högbom 1974) remains the standard in RI imaging, owing to its simplicity and computational efficiency. However, CLEAN reconstructions exhibit intrinsic limitations, particularly as their angular resolution is restricted to the nominal instrumental resolution. Complex emission with extreme dynamic ranges can also be challenging to CLEAN in spite of its advanced variants (e.g., Bhatnagar & Cornwell 2004; Cornwell 2008; Offringa et al. 2014). Furthermore, manual intervention is often required for stability, due to the greedy nature of the algorithm.

Computational imaging techniques from Bayesian inference (Junklewitz et al. 2016; Cai et al. 2018; Arras et al. 2019) and optimization theories (e.g., Wiaux et al. 2009; Carrillo et al. 2012; Garsden et al. 2015; Dabbech et al. 2015; Repetti & Wiaux 2020) were developed over the last decade or so. Bayesian methods provide quantification of the uncertainty about the image estimate. Optimization algorithms enable injecting handcrafted regularization into the data. One such example is the SARA family (Carrillo et al. 2012; Onose et al. 2016, 2017; Repetti & Wiaux 2020; Terris et al. 2022, 2023), promoting nonnegativity and average sparsity in a redundant wavelet dictionary. The SARA algorithm and its extension to unconstrained minimization, uSARA, demonstrated high imaging precision on real RI data from modern telescopes, outperforming CLEAN (Dabbech et al. 2018, 2022; Wilber et al. 2023a). However, all these methods are highly iterative, which hinders their scalability to the sheer data volumes expected from next-generation instruments such as the flagship Square Kilometre Array (SKA; Labate et al. 2022; Swart et al. 2022).

Recent advances in deep learning, from PnP algorithms to advanced end-to-end DNNs, have opened the door to a whole new paradigm in computational imaging owing to their modeling power and speed. PnP algorithms are at the intersection of optimization theory and deep learning. They entail training a denoising DNN, which is then deployed within an optimization algorithm as a substitute for a handcrafted, and often sub-iterative regularization operator (Venkatakrishnan et al. 2013; Chan et al. 2017; Zhang et al. 2017; Kamilov et al. 2023). One such example is AIRI, standing for “AI for Regularization in radio-interferometric Imaging” (Terris et al. 2022), underpinned by the Forward-Backward algorithmic structure from convex optimization theory. AIRI has been demonstrated to achieve slightly superior imaging precision to uSARA at a lower computational cost on real large-scale high-dynamic range RI data from SKA pathfinder and precursor instruments (Dabbech et al. 2022; Wilber et al. 2023b). Yet, PnP algorithms remain as highly iterative as their pure optimization counterparts. Fully data-driven end-to-end DNNs have been proposed (Connor et al. 2022). They are able to provide ultra-fast reconstruction, but simultaneously lose robustness (interpretability and generalizability) by failing to enforce fidelity to data. More advanced end-to-end unrolled DNN architectures have emerged (see Monga et al. 2021, and references within). Unrolled DNNs are model-based architectures designed to ensure the consistency of the reconstructed images with the measurements by unrolling the iteration struc-

ture of an optimization algorithm in its layers. They address the robustness issues of data-driven end-to-end approaches, and have been demonstrated to deliver a high degree of precision across a variety of applications (Kamilov et al. 2023). However, they reintroduce a lack of scalability, owing to the fact that embedding large-scale measurement operators in the network architecture is impractical both during training and inference. This constraint is particularly acute for RI imaging.

To address the scalability challenge of PnP approaches, while preserving their robustness, we introduce a novel deep-learning approach, dubbed “Residual-to-Residual DNN series for high-Dynamic range imaging”. R2D2’s reconstruction is formed as a series of residual images, iteratively estimated as outputs of DNNs taking the previous iteration’s image estimate and associated data residual as inputs. It thus takes a hybrid structure between a PnP algorithm and a learned version of the matching pursuit algorithm that underpins CLEAN. The main contribution of this paper is twofold. Firstly, we provide a detailed description of the R2D2 algorithm, generalizing from recent work (Aghabiglou et al. 2023; Hauptmann et al. 2018). We discuss two incarnations distinguished by their DNN architectures. We also present a telescope-specific training methodology for R2D2. Secondly, we present an in-depth analysis of the algorithm’s imaging precision and computational efficiency in simulation in comparison with the optimization algorithm uSARA, the PnP algorithm AIRI (Terris et al. 2022), and CLEAN (Offringa et al. 2014; Offringa & Smirnov 2017). In a sister article, we demonstrate the R2D2 algorithm trained for VLA on real high-dynamic range RI data (Dabbech et al. 2024).

The remainder of this paper is as follows. Section 2 describes the R2D2 algorithm featuring its two incarnations. Section 3 elaborates on the training methodology of the R2D2 algorithm, including the underpinning core DNN architecture, the diversified ground truth database, and resulting VLA-focused training datasets. It also presents implementation details of the training and its computational aspects. Section 4 provides a detailed analysis of the R2D2 algorithm performance, in comparison with state-of-the-art RI algorithms. Conclusions and future work are stated in Section 5.

2. R2D2 ALGORITHM

In this section, we recall the RI data model including its Fourier and image-domain formulations. We present the R2D2 iteration structure and explain the sequential training of its underpinning DNN series. We finally de-

scribe two different incarnations of the R2D2 algorithm relying on distinct DNN architectures.

2.1. Data model

In aperture synthesis by radio interferometry, at a given time instant, each pair of antennas acquires a noisy complex measurement, termed visibility, corresponding to a Fourier component of the sought radio emission (Thompson et al. 2017). The collection of the sensed Fourier modes accumulated over the total observation period forms the Fourier sampling pattern, describing an incomplete coverage of the 2-dimensional (2D) Fourier plane. With no loss of generality, let $\mathbf{x}^* \in \mathbb{R}_+^N$ represent the unknown radio image of interest. The RI data $\mathbf{y} \in \mathbb{C}^M$ can be modeled as:

$$\mathbf{y} = \Phi \mathbf{x}^* + \mathbf{n}, \quad (1)$$

where $\mathbf{n} \in \mathbb{C}^M$ represents a complex Gaussian random noise with a standard deviation $\tau > 0$ and mean 0. The linear operator $\Phi: \mathbb{R}^N \rightarrow \mathbb{C}^M$ is the measurement operator (i.e., describing the acquisition process), consisting in a nonuniform Fourier sampling. The operator Φ can also include a data-weighting scheme to improve the observation's effective resolution (e.g., Briggs weighting; Briggs 1995). Performing a discrete Fourier transform with the expected large amount of data is impractical. Often, the incomplete Fourier sampling is modeled using the nonuniform fast Fourier transform (NUFFT; Fessler & Sutton 2003; Thompson et al. 2017). In our proposed algorithm, we adopt the image-domain formulation of the RI data, obtained from (1) via a normalized back-projection such that:

$$\mathbf{x}_d = \kappa \text{Re}\{\Phi^\dagger \mathbf{y}\} = \mathbf{D} \mathbf{x}^* + \mathbf{b}, \quad (2)$$

where $\mathbf{x}_d \in \mathbb{R}^N$ stands for the back-projected data, known as the *dirty* image, with $(\cdot)^\dagger$ denoting the adjoint of its argument operator, and $\text{Re}\{\cdot\}$ the real part of its argument. Considering $\boldsymbol{\delta} \in \mathbb{R}^N$, the image with value 1 at its center and 0 elsewhere, the normalization factor $\kappa > 0$ is equal to $1/\max(\text{Re}\{\Phi^\dagger \Phi \boldsymbol{\delta}\})$, ensuring that the point spread function (PSF), defined as $\mathbf{h} = \kappa \text{Re}\{\Phi^\dagger \Phi \boldsymbol{\delta}\} \in \mathbb{R}^N$, has a peak value of 1. The linear operator $\mathbf{D}: \mathbb{R}^N \rightarrow \mathbb{R}^N$ maps the unknown image of interest to the dirty image space by encoding the Fourier de-gridding and gridding operations, and is defined as $\mathbf{D} \triangleq \kappa \text{Re}\{\Phi^\dagger \Phi\}$. Finally, the image-domain noise vector $\mathbf{b} = \kappa \text{Re}\{\Phi^\dagger \mathbf{n}\} \in \mathbb{R}^N$ represents the normalized back-projected noise.

2.2. Algorithmic structure

The R2D2 algorithm requires training a series of I DNNs represented as $(\mathbf{N}_{\hat{\boldsymbol{\theta}}^{(i)}})_{1 \leq i \leq I}$, and characterized by

their learned parameters $(\hat{\boldsymbol{\theta}}^{(i)} \in \mathbb{R}^Q)_{1 \leq i \leq I}$. Each network component takes two input images consisting of the previous image estimate $\mathbf{x}^{(i-1)}$ and its associated back-projected data residual $\mathbf{r}^{(i-1)}$, to which we refer as residual dirty image given by:

$$\mathbf{r}^{(i-1)} = \mathbf{x}_d - \mathbf{D} \mathbf{x}^{(i-1)}. \quad (3)$$

The current image estimate is updated from the output of the network component $\mathbf{N}_{\hat{\boldsymbol{\theta}}^{(i)}}(\mathbf{r}^{(i-1)}, \mathbf{x}^{(i-1)})$. The iteration structure of the R2D2 algorithm reads:

$$\mathbf{x}^{(i)} = \mathbf{x}^{(i-1)} + \mathbf{N}_{\hat{\boldsymbol{\theta}}^{(i)}}(\mathbf{r}^{(i-1)}, \mathbf{x}^{(i-1)}), \quad (4)$$

with the image estimate and the residual dirty image initialized to $\mathbf{x}^{(0)} = \mathbf{0}$ and $\mathbf{r}^{(0)} = \mathbf{x}_d$. R2D2's iteration structure enables progressive improvement of both the resolution and dynamic range of the image estimate. Illustration of the R2D2 algorithm is provided in Figure 1. One can see how the learned residual images depict smooth structure at early iterations, and progressively capture finer details together with fainter structure at later iterations. R2D2's reconstruction $\hat{\mathbf{x}}$ thus takes the simple series expression below:

$$\hat{\mathbf{x}} \triangleq \mathbf{x}^{(I)} = \sum_{i=1}^I \mathbf{N}_{\hat{\boldsymbol{\theta}}^{(i)}}(\mathbf{r}^{(i-1)}, \mathbf{x}^{(i-1)}). \quad (5)$$

Formally, I is an algorithmic parameter with an appropriate procedure necessary to determine its value.

The R2D2 algorithm features a hybrid structure between a learned version of matching pursuit, of which CLEAN is a well-known example, and a Forward-Backward PnP algorithm, such as AIRI. All three algorithmic structures are iterative, alternating at each iteration between the computation of a residual dirty image and a regularization step. They differ by the way their regularization operators are built and the variables they take as inputs. Firstly, as per (4), the R2D2 DNNs take the current image estimate and its associated residual dirty image as two separate input channels, and are trained to output a residual image serving as an additive update to the current image estimate. Secondly, at each iteration, CLEAN identifies model components by projecting the residual dirty image onto a sparsity dictionary, either the identity basis in the standard CLEAN (Högbom 1974; Schwab 1984), or bespoke multiscale kernels in multiscale CLEAN (Cornwell 2008). Its iterations follow the exact same structure as (4), with a minor cycle in lieu of R2D2's DNN. In contrast to an R2D2 DNN, CLEAN's minor cycle structure is iterative itself, taking information from the residual dirty image only (not explicitly the current image estimate). Thirdly, AIRI's image update is computed as the output of a

learned denoiser, taking as input a linear combination of the current image estimate and its associated residual dirty image (Terris et al. 2022).

2.3. DNN series training

The R2D2 algorithm is underpinned by a series of DNNs, trained sequentially. The first network in the series $\mathbf{N}_{\hat{\theta}^{(1)}}$ is trained using a dataset composed of L ground truth and dirty image pairs $(\mathbf{x}_l^*, \mathbf{x}_{dl})_{1 \leq l \leq L}$, with the input image estimates initialized such that $\mathbf{x}_l^{(0)} = \mathbf{0}$, for all $1 \leq l \leq L$. Subsequent networks $\mathbf{N}_{\hat{\theta}^{(i)}}$, at any given iteration $i > 1$, are trained using a dataset consisting of the image triplets $(\mathbf{x}_l^*, \mathbf{x}_l^{(i-1)}, \mathbf{r}_l^{(i-1)})_{1 \leq l \leq L}$. The learnable parameters $\hat{\theta}^{(i)}$ of each network are estimated by solving the loss function of the form:

$$\min_{\theta^{(i)} \in \mathbb{R}^Q} \frac{1}{L} \sum_{l=1}^L \|\mathbf{x}_l^* - [\mathbf{x}_l^{(i-1)} + \mathbf{N}_{\theta^{(i)}}(\mathbf{r}_l^{(i-1)}, \mathbf{x}_l^{(i-1)})]_+\|_1, \quad (6)$$

where $\|\cdot\|_1$ stands for the ℓ_1 -norm, and $[\cdot]_+$ denotes the projection onto the positive orthant $\mathbb{R}_+^N$ to enforce nonnegativity of the image estimates throughout the iterative process. The output of each trained DNN is used to update the image estimates $(\mathbf{x}_l^{(i)})_{1 \leq l \leq L}$ as specified in (4), and the associated residual dirty images $(\mathbf{r}_l^{(i)})_{1 \leq l \leq L}$ as per (3). The updated image pairs are then incorporated in the training dataset of the next DNN. The loss functions of the form (6) are solved using the Root Mean Square Propagation (RMSProp) algorithm, with the learnable parameters of each network initialized from the estimated parameters of the preceding trained network.

The sequential training concludes when the evaluation metrics of the reconstruction quality achieved by the validation dataset reach a point of stabilization. This offers a way to determine the number of required networks I at the training level, making R2D2 parameter-free when it comes to reconstructing an image from specific data.

2.4. Normalization procedures

To avoid generalizability issues arising from large variations in pixel value ranges between the training dataset and images of interest (e.g., test images), normalization procedures must be deployed. We consider an iteration-specific normalization at both training and inference stages. On the one hand, the training of the network component at any iteration $i > 1$ takes as input a normalized dataset where each image triplet $(\mathbf{x}_l^*, \mathbf{x}_l^{(i-1)}, \mathbf{r}_l^{(i-1)})$ is divided by $\alpha_l^{(i-1)}$, the mean value of the previous image estimate $\mathbf{x}_l^{(i-1)}$. The first network component ($i = 1$) is also trained using a normalized dataset, where each image pair $(\mathbf{x}_l^*, \mathbf{x}_{dl})$ is di-

vided by $\alpha_l^{(0)}$, the mean value of the dirty image $\mathbf{x}_{dl}$. On the other hand, at inference (see (4) and (5)), all network components are applied on normalized inputs, with their outputs de-normalized accordingly. Hence, formally: $\mathbf{N} \mapsto \alpha \mathbf{N}(\cdot/\alpha)$, where $\alpha \in \mathbb{R}$ is an iteration-specific normalization factor obtained via the exact same procedure as in the training stage.

2.5. R2D2 incarnations

We present two incarnations of the R2D2 algorithm distinguished by the architecture of their network components $(\mathbf{N}_{\hat{\theta}^{(i)}})_{1 \leq i \leq I}$. The first features a fully data-driven end-to-end DNN architecture. The second features a data model-informed DNN architecture unrolling the R2D2 algorithm itself, dubbed R2D2-Net. Nonetheless, both incarnations rely on the same DNN core architecture denoted by $\mathbf{C}_\beta$, with $\beta \in \mathbb{R}^P$ standing for its learnable parameters.

The first incarnation takes simply the DNN $\mathbf{C}_\beta$ as the architecture of its network components. Under this consideration, each network component takes the form $\mathbf{N}_{\hat{\theta}^{(i)}} \equiv \mathbf{C}_{\beta^{(i)}}$, and its learned parameters are given by $\hat{\theta}^{(i)} = \beta^{(i)} \in \mathbb{R}^{Q=P}$, for any $1 \leq i \leq I$. For simplicity, we refer to this incarnation as R2D2.

For the second incarnation, each of its R2D2-Net components comprises J layers of the core architecture $\mathbf{C}_\beta$ interleaved with $J - 1$ approximate data-fidelity layers. The learned parameters of each R2D2-Net component $\mathbf{N}_{\hat{\theta}^{(i)}}$ thus take the form $\hat{\theta}^{(i)} = [\beta_1^{(i)}, \dots, \beta_J^{(i)}] \in \mathbb{R}^{Q=P \times J}$. Specific to the data-fidelity layers, the residual dirty images are computed using the image-domain convolution with the PSF instead of the RI mapping operator $\mathbf{D}$. This mapping is fast and memory efficient, enabling the training of R2D2-Net on GPUs. In reference to its nesting structure, we dub this incarnation of the R2D2 algorithm as the “Russian Doll R2D2”, in short R3D3.

3. TRAINING APPROACH

This section introduces the training methodology for both incarnations of the R2D2 algorithm. It covers the underpinning core architecture and explains the procedure to build training datasets from a database of low-dynamic range images, using VLA sampling patterns. Implementation details including the computational cost of the training are also provided.

3.1. U-Net core architecture

We consider the well-known U-Net (Ronneberger et al. 2015) as $\mathbf{C}_\beta$, the core architecture of both incarnations of the R2D2 algorithm. R2D2 network components simply take the U-Net architecture. For R3D3, its R2D2-Net

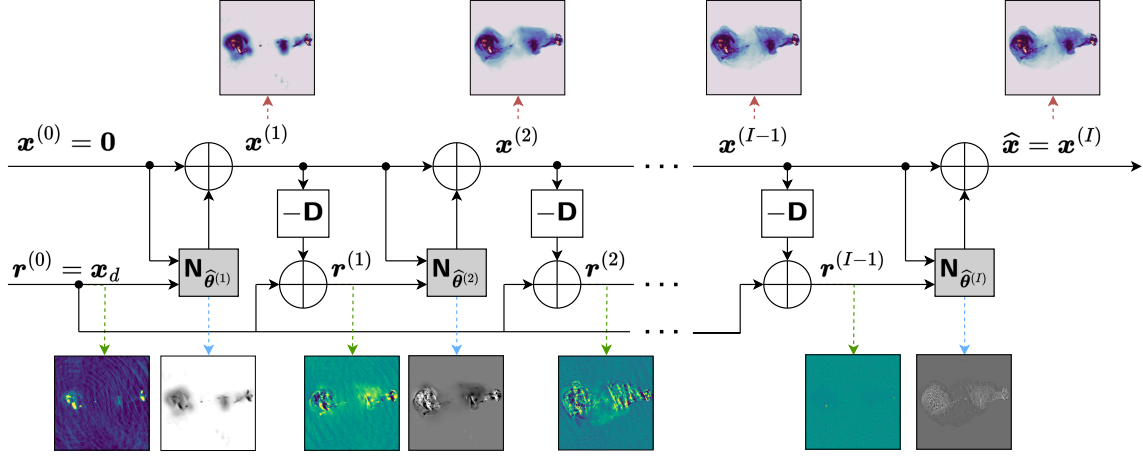


Figure 1. Illustration of the R2D2 algorithm. Both image iterates $\mathbf{x}^{(i-1)}$ and associated residual dirty images $\mathbf{r}^{(i-1)}$ are fed to R2D2 DNNs as input. R2D2 DNNs’ output are then used to update the next image iterates. The sequence of the image iterates and corresponding residual dirty images are indicated with dashed red and green arrows, respectively. The sequence of the learned residual images are indicated with blue arrows.

network components take the same U-Net in their J network layers.

Typically, a U-Net consists of two distinctive parts. The first is the contracting path which aims to capture the semantic aspects of the network’s input, encapsulated in the so-called feature maps (LeCun et al. 2015). To do so, it applies successive multi-channel convolutions with increasing number of channels, followed by pooling layers to reduce the spatial dimension. The second is the expanding path which mirrors the structure of the contracting path. It aims to restore the spatial resolution by applying successive multi-channel convolutions and up-sampling layers to the feature maps to increase their spatial dimension while integrating information from the contracting path.

The adopted U-Net architecture is illustrated in Figure 2. In the contracting path, each convolutional layer consists of two blocks of multi-channel 3×3 convolutions combined with the LeakyReLU activation function. Every two convolutional layers are followed by a 2D average-pooling layer with a stride of 2. After each down-sampling step, the number of feature channels is doubled. The first convolutional layer takes the network’s input corresponding to the previous image estimate and its associated back-projected data residual, both of size 512×512 , and produces a 3D feature map of size $512 \times 512 \times 64$, with 64 being the number of feature channels. At the end of the contracting path, the reached bottleneck layer produces low-resolution 3D feature maps of size $32 \times 32 \times 1024$. The expanding path substitutes the 2D average-pooling with the 2D transposed convolution to double the spatial dimension of the output feature maps and utilizes skip connections from

the down-sampling layers to restore the fine details. The final output layer employs a 1×1 convolution layer resulting in a single-channel output that is the learned residual image.

3.2. Ground truth database

Key to the training of our deep-learning model is a large database of ground-truth images with a wide variety of features and dynamic ranges, and free from noise and artifact structure. In the absence of a large physical RI database readily providing these characteristics, we build a database of curated ground-truth images from real low-dynamic range astronomical and medical images, sourced as follows. Radio astronomy images are gathered from the National Radio Astronomy Observatory (NRAO) Archives, and LOFAR surveys, namely, LOFAR HBA Virgo cluster survey (Edler et al. 2023) and LoTSS-DR2 survey (Shimwell et al. 2022). Optical astronomy images are gathered from the National Optical-Infrared Astronomy Research Laboratory. Medical images are selected from the NYU fastMRI Initiative Database (Zbontar et al. 2018; Knoll et al. 2020). Training using a curated database originating from other modalities and applications has shown to be effective for RI imaging (Terris et al. 2022, 2023).

Ground-truth images of size $N = 512 \times 512$, are generated using the pre-processing procedure proposed in Terris et al. (2023). More precisely, various operations including concatenation, rotation, translation, and edge smoothing, are applied specifically to the medical images to deconstruct their anatomical features. Denoising is applied to all images, of both medical and astronomical origins, to eliminate artifacts and noise, using a denoising DNN (Zhang et al. 2023) in combination

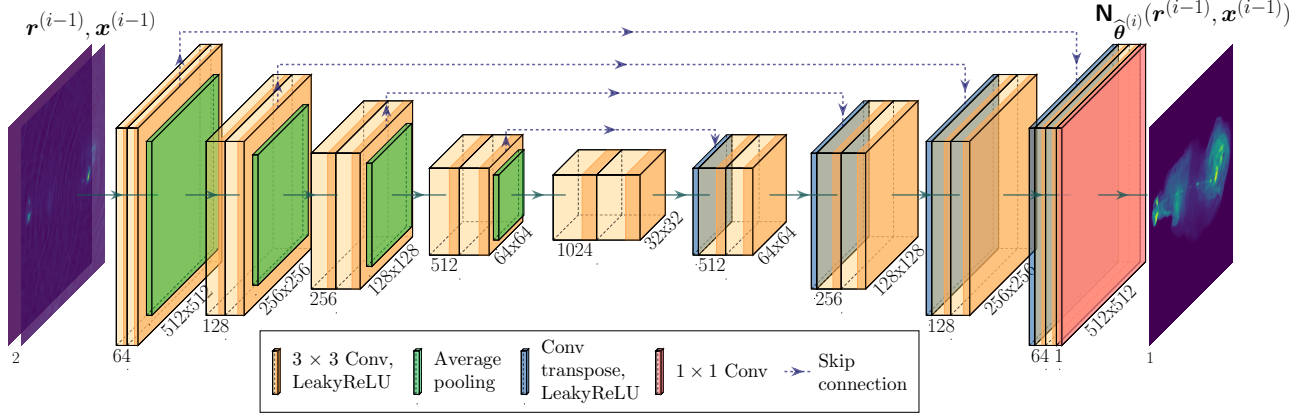


Figure 2. Illustration of the U-Net core architecture underpinning the different incarnations of the R2D2 algorithm. The convolutional layers of the network, represented as boxes, apply multi-channel convolutions, followed by down-sampling (contracting path) or up-sampling (expanding path). The outcome of these layers are 3D feature maps, whose dimensions are specified around the corresponding boxes: the 2D spatial dimension at the bottom center, and the number of feature channels at the outer edge.

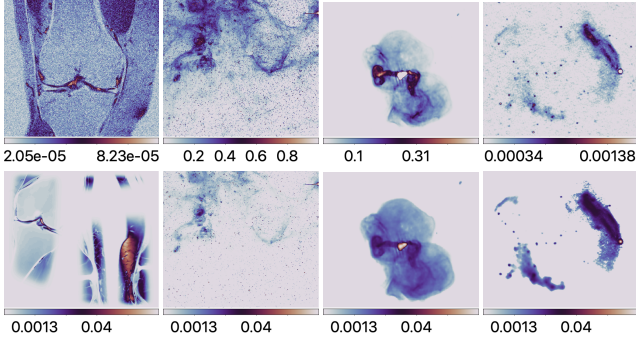


Figure 3. Selection of raw low-dynamic range images (linear scale; top row), and corresponding ground-truth images after pre-processing (logarithmic scale; bottom row). The training dataset includes medical images (first column), and optical astronomy images (second column). The validation dataset comprises images of giant radio galaxies (e.g., Messier 87; third column), and radio galaxy clusters (e.g., Abell 746; fourth column).

with soft-thresholding operations. Additionally, a pixel-wise exponentiation transform can be applied to the curated ground-truth images to emulate the characteristic high dynamic range of radio images (Terris et al. 2022). Examples of raw low-dynamic range images and their corresponding denoised and exponentiated ground-truth images are shown in Figure 3.

3.3. VLA-specific training

In training both incarnations of the R2D2 algorithm, we take a telescope-specific approach, encompassing all observation settings of a single telescope. While the methodology is implemented for the VLA, a similar process can be undertaken for other telescopes as well.

VLA antennas are re-configurable into four configurations A, B, C, and D. For a given observation frequency, configuration A provides the longest baselines, hence the highest angular resolution. In contrast, configuration D provides the shortest baselines, offering the best surface brightness sensitivity to extended emission. Depending on the science target, astronomers often probe the radio sky with multiple configurations of the VLA. Herein, we opt for the combination of configurations A and C for a balanced Fourier coverage. To obtain diversified datasets, we consider a wide variety of observation setting, by uniformly randomizing (i) the pointing direction with a declination in the range $[5, 60]$ degrees and a right ascension in the range $[0, 23]$ hr (J2000), (ii) the total observation time with configuration A $t_{\text{obs-A}}$ in the time interval $[5, 10]$ hr, and that of configuration C $t_{\text{obs-C}}$ in the time interval $[1, 3]$ hr, (iii) the frequency bandwidth such that the ratio between the highest and the lowest frequencies ρ_{freq} is in the range $[1, 2]$, and (iv) the number of frequencies n_{freq} between 1 and 4. Under these considerations, Fourier sampling patterns are created using the MeqTrees software (Noordam & Smirnov 2010). Additionally, a random rotation and flagging of up to 20% of their data points are applied. The total number of points in the resulting Fourier sampling patterns ranges from 2×10^5 to 2×10^6 .

The NUFFT measurement operators Φ are built from the generated sampling patterns, assuming a fixed ratio between the spatial Fourier bandwidth of the ground-truth images and the spatial bandwidth of the Fourier sampling that is set to 1.5. To enhance the effective resolution of the modeled visibilities, the measurement op-

erators incorporate the Briggs weighting scheme (Briggs 1995). The scheme is standard in RI imaging as it enables an adjustable trade-off between resolution and sensitivity. The weights are generated using the WSClean software with the Briggs parameter set to 0 (Offringa et al. 2014; Offringa & Smirnov 2017).

For the ground-truth images, their dynamic range ρ_{DR} is increased using the pixel-wise exponentiation transform described in Terris et al. (2022) and is varied in the interval $[10^3, 5 \times 10^5]$. More precisely, ρ_{DR} is uniformly randomised in the logarithmic scale such that $\log_{10}(\rho_{\text{DR}}) \in [3, 5.69]$. Using the data model (1), modeled visibilities are corrupted with additive random Gaussian noise, with a standard deviation τ set in an adaptive manner following a stipulation proposed in Terris et al. (2022) to ensure that the measurement noise adapts to the dynamic range of the radio image of interest. More precisely, for each ground truth image with a given dynamic range ρ_{DR} , τ is fixed such that:

$$\tau = \eta_{\text{Briggs}} (1/\rho_{\text{DR}}) \sqrt{2\|\text{Re}\{\Phi^\dagger \Phi\}\|_S}, \quad (7)$$

where $\|\cdot\|_S$ denotes the spectral norm of its argument operator, and $\eta_{\text{Briggs}} > 0$ is a correction factor accounting for the Briggs weights, which reduces to 1 otherwise (Wilber et al. 2023a). The dirty images are then obtained via back-projection of the RI data as per (2). The resulting images are of size $N = 512 \times 512$, same as the ground-truth images, with a pixel-size aligning with the super-resolution factor set to 1.5. Examples of simulated VLA sampling patterns and dirty images are provided in Figure 4.

Training and validation datasets are composed of pairs of high-dynamic range ground-truth images and associated dirty images. The training dataset incorporates ground-truth images generated from the sourced medical and optical astronomy images. The validation dataset incorporates ground-truth images obtained from the sourced radio astronomy images. The dichotomy in the nature of the images in both datasets aims to ensure the generalizability of the trained DNN series. The training and validation datasets consist of 20000 and 250 pairs of ground-truth images and associated dirty images, respectively.

3.4. Implementation

We train R2D2 and two realizations of R3D3 which are distinguished by the number of layers in their R2D2-Net components such that $J \in \{3, 6\}$. Hereafter, we refer to these R3D3 realizations as R3D3^{3L} and R3D3^{6L}. Similarly, we refer to their corresponding R2D2-Net realizations as R2D2-Net^{3L} and R2D2-Net^{6L}.

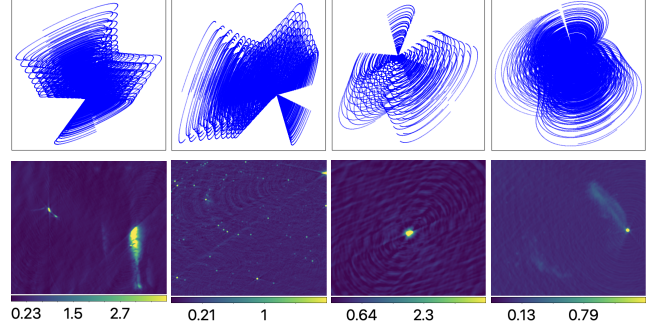


Figure 4. A selection of simulated 2D Fourier sampling patterns with the antenna configuration of the VLA (top row) and resulting dirty images (linear scale; bottom row). The depicted dirty images correspond to the ground-truth images shown in Figure 3.

In the sequential training of the different DNN series, we consider a pruning procedure which progressively reduces the size of the training dataset, initially set to $L = 20000$ image pairs. It relies on a data fidelity-based criterion to decide on the convergence of the inverse problem associated with each training image pair $(\mathbf{x}_l^*, \mathbf{x}_{dl})$ from the initial training dataset. More specifically, at any given iteration $i > 1$, if the updated residual dirty image $\mathbf{r}_l^{(i)}$ reaches the image-domain noise level by satisfying the condition $\|\mathbf{r}_l^{(i)}\|_2^2 \leq \|\mathbf{b}_l\|_2^2$, the pair $(\mathbf{x}_l^*, \mathbf{x}_{dl})$ is discarded from the training dataset of subsequent DNNs in the series. The validation dataset, typically significantly smaller in size, remains unchanged throughout the training process.

The dataset pruning procedure can accelerate the training process and potentially reduce its computational cost, especially the cost incurred for the update of the residual dirty images. Furthermore, by eliminating solved inverse problems, the training of subsequent DNNs becomes focused on relevant inverse problems. The procedure can therefore improve the imaging precision of the DNN series. Interestingly, it also informs an additional stopping criterion for the training sequence, specifically, when the training dataset is reduced down to an inappropriate size for efficient DNN training.

Training was conducted on Cirrus, a UK Tier 2 high-performance computing (HPC) service, equipped with both CPU and GPU compute nodes. The CPU nodes are composed of dual Intel 18-core Xeon E5-2695 processors with 256 GB of memory each. The GPU nodes are composed of two 20-core Intel Xeon Gold 6148 processors, four NVIDIA Tesla V100-SXM2-16GB GPUs, and 384 GB of DRAM memory. Computation of the dirty images and updates of the residual dirty images were run on the CPU nodes. DNNs' training relied on the PyTorch library in Python (Paszke et al. 2019), and was performed on the GPU nodes. The learning rate

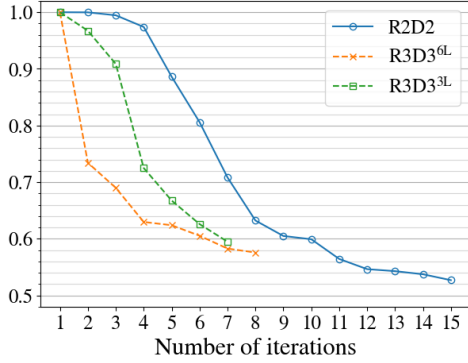


Figure 5. Results of the dataset pruning procedure. Evolution of the size of the training dataset shown as a fraction of the size of the initial training dataset, throughout the iterations of R2D2 and both R3D3 realizations (R3D3^{3L}, R3D3^{6L}).

was set to 10^{-4} . Due to GPU memory constraints, a batch size of 4 and 1 was used for R2D2 and R3D3 realizations, respectively. Note that given the manageable scale of both image and data sizes of the training datasets, the computation of the residual dirty images can be efficiently performed on GPUs.

R2D2 training concluded at iteration $I = 15$. For R3D3 realizations, training of R3D3^{3L} and R3D3^{6L} concluded at iterations $I = 7$ and $I = 8$, respectively. The VLA-trained R2D2 and R3D3 DNN series are available in the dataset [Aghabiglou et al. \(2024\)](#). Figure 5 shows the evolution of the training dataset size during the training of R2D2 and R3D3 realizations (R3D3^{3L}, R3D3^{6L}). For R2D2, the reduction of the training dataset was triggered from the third iteration. For R3D3^{3L} and R3D3^{6L}, the procedure was activated immediately after their first iteration, with up to 25% of the training dataset discarded with the deeper network component R2D2-Net^{6L}. This suggests the ability of data model-informed DNN architectures to accelerate the data fidelity of the image estimates, which aligns with the smaller number of terms required in their associated DNN series. At least 40% of the initial training dataset was discarded when training the last network component of all DNN series, suggesting the efficiency of the procedure. In principle, more network components can be trained with the remaining training datasets. However, in our current setting, the sequential training of all DNN series terminated based on the evaluation metrics of the validation dataset (see Section 2.3).

3.5. Computational cost

The computational cost of building the different DNN series in terms of CPU core time (CPU hr) incurred for the computation of the residual dirty images, and GPU time (GPU hr) incurred for the training of the DNNs, is

	I	$Q(\times 10^6)$	n_{epochs}	n_{gpu}	n_{cpu}	GPU hr	CPU hr
U-Net	1	31	264	4	6	82	336
R2D2-Net ^{3L}	1	93	405	12	6	873	336
R2D2-Net ^{6L}	1	186	192	12	6	414	336
R2D2	15	31	398	4	6	160	4757
R3D3 ^{3L}	7	93	605	12	6	1291	2165
R3D3 ^{6L}	8	186	591	12	6	1276	2244

Table 1. Training computational details of R2D2, R3D3 realizations (R3D3^{3L}, R3D3^{6L}), and the end-to-end DNNs corresponding to the first components in their series, U-Net and R2D2-Net (R2D2-Net^{3L}, R2D2-Net^{6L}), respectively. Results are reported in terms of: the number of iterations (I), the number of learnable parameters of their network components (Q), the cumulative number of epochs (n_{epochs}), the number of CPU cores (n_{cpu}) deployed for generating the dirty images and updating the residual dirty images, and the number of GPUs (n_{gpu}), deployed for DNN training and updating the image estimates. The training computational cost is provided in GPU hr and CPU hr.

provided in Table 1. The computational cost of the first network component of their series is also provided. On the one hand, R2D2 required approximately 80% less GPU hr than both realizations of R3D3, owing to the simpler architecture of its network components. On the other hand, it demanded about twice as many CPU hr for updating the residual dirty images, since it is trained with almost twice as many network components in its series. Interestingly, for all DNN series, 30 to 60% of the GPU hr cost was incurred in the training of their first network component. Their subsequent network components benefited from efficient initialization of their learnable parameters using their preceding trained component, and a reduced training dataset thanks to the data fidelity-driven pruning procedure. Both realizations of R3D3 required similar resources in both GPU hr and CPU hr, suggesting a balance between the depth of their network components and the required number of iterations in their series.

4. SIMULATIONS AND RESULTS

In this section, we study the performance of the VLA-trained incarnations of the R2D2 algorithm in terms of the imaging precision. We present the RI imaging algorithms used as a benchmark including their parameter choice and the adopted evaluation metrics. In our analysis, two experimental setups are considered. The first is generic, where the image and observation settings are fully randomized. The second is focused on exploring different regimes of key parameters of the image and observation settings. The computational efficiency of the proposed algorithm is studied using its implementations in Python and MATLAB.

4.1. Benchmark algorithms & parameter choice

The performance of the R2D2 algorithm is evaluated in comparison with the sparsity-based algorithm from optimization theory uSARA (Repetti & Wiaux 2020; Terris et al. 2022), the PnP algorithm AIRI (Terris et al. 2022, 2023), and multiscale CLEAN (Cornwell 2008) in the WSClean software (Offringa et al. 2014; Offringa & Smirnov 2017). All algorithms formally feature free parameters, whose values should be fixed following an appropriate procedure. Firstly, uSARA features a parameter that controls the trade-off between its handcrafted regularization function and data fidelity. AIRI features a parameter controlling the appropriate noise level of its underpinning DNN denoiser. The choice of uSARA and AIRI parameter is automated using noise-driven heuristics (Terris et al. 2022; Dabbech et al. 2022; Wilber et al. 2023a). Adjustments from the heuristic values can sometimes be required for optimized results. Secondly, in WSClean, multiscale CLEAN parameters are set according to nominal values, often requiring adjustments for improved results. In particular, the parameters controlling the cleaning depth can require tuning as they directly impact the data fidelity of the CLEAN reconstruction. For the proposed simulations, optimized values of the uSARA and AIRI heuristics, and the CLEANing depth were studied and fixed to the same value for batches of simulated inverse problems (assuming an automated/pipeline mode reconstruction) rather than for each reconstruction specifically (see Section 4.4). Finally, as discussed in Section 2.3, we note that the R2D2 algorithm is here seen as parameter-free with the number of its network components determined at the training stage (the number of iterations is set to the number of trained DNNs). This represents an interesting advantage over the benchmark in the perspective of the deployment of the imaging algorithms in automated mode.

4.2. Implementation

For the benchmark algorithms, we use multiscale CLEAN implemented in C++ as part of the WSClean software, and uSARA and AIRI MATLAB implementations in the BASPLib¹ code library. The R2D2 algorithm (in its two incarnations) comes in two distinct implementations in MATLAB and Python. The former shares the NUFFT implementation of AIRI and uSARA, and the latter uses a fast PyTorch-specific implementation (Muckley et al. 2020). Although both NUFFT implementations are based on the works of Fessler & Sutton (2003), the Python version uses a table-based inter-

polation, whereas the MATLAB version uses a more precise sparse matrix interpolation. The end-to-end U-Net and R2D2-Net, representing the first iterations of R2D2 and R3D3, benefit from both MATLAB and Python implementations.

All algorithms were executed on Cirrus, utilizing similar resources where relevant. Given the relatively small image and data dimensions considered in this work, minimal resources were allocated. CLEAN and uSARA were run using 1 CPU core each. AIRI, R2D2, and R3D3, in their MATLAB implementation, utilized 1 CPU core for data-fidelity specific operations and 1 GPU for the DNN inference. The MATLAB implementation of the end-to-end U-Net and R2D2-Net used a similar allocation, dedicating the CPU core for the computation of the dirty image. For the Python implementation of R2D2 and R3D3 (respectively, U-Net and R2D2-Net), two resource allocation settings are supported: (i) a hybrid setting that is consistent with their MATLAB version which used 1 CPU core for the update of the residual dirty images (respectively, computation of the dirty image) and 1 GPU for the DNN inference, (ii) and a full GPU implementation which used 1 GPU.

4.3. Evaluation metrics

A qualitative and quantitative evaluation of the reconstruction quality obtained by the different algorithms is provided through the visual examination of the image estimates, and the signal-to-noise ratio metric, both in linear scale (SNR) and logarithmic scale (logSNR). More precisely, considering a ground truth image $\mathbf{x}^*$ and an image estimate $\hat{\mathbf{x}}$, the SNR metric reads:

$$\text{SNR}(\hat{\mathbf{x}}, \mathbf{x}^*) = 20\log_{10}(\|\mathbf{x}^*\|_2 / \|\mathbf{x}^* - \hat{\mathbf{x}}\|_2). \quad (8)$$

The logSNR metric, a variation of SNR, is introduced to assess the reconstruction quality with emphasis on faint structure. For this purpose, we consider the following logarithmic mapping of the images of interest, parametrized by $a > 0$:

$$\text{rlog}(\mathbf{x}) = x_{\max} \log_a \left(\frac{a}{x_{\max}} \mathbf{x} + \mathbf{1} \right), \quad (9)$$

where $x_{\max}$ is the peak pixel value of the image $\mathbf{x}$ and $\mathbf{1} \in \mathbb{R}^N$ is a vector of values equal to 1. Setting the parameter a to ρ_{DR} , the dynamic range of the ground truth image, the logSNR metric reads:

$$\log\text{SNR}(\hat{\mathbf{x}}, \mathbf{x}^*) = \text{SNR}(\text{rlog}(\hat{\mathbf{x}}), \text{rlog}(\mathbf{x}^*)). \quad (10)$$

We also examine image-domain data fidelity obtained by all algorithms. Considering a dirty image $\mathbf{x}_d$ and an estimated residual dirty image $\hat{\mathbf{r}}$, the metric denoted by $\bar{\sigma}_{\text{res}}$ reads:

$$\bar{\sigma}_{\text{res}}(\hat{\mathbf{r}}, \mathbf{x}_d) = \|\hat{\mathbf{r}}\|_2 / \|\mathbf{x}_d\|_2. \quad (11)$$

¹ <https://basp-group.github.io/BASPLib/>

Additionally, we provide a qualitative assessment via the visual inspection of the estimated residual dirty images.

The computational performance of the different algorithms is analyzed using the total number of iterations I , the total computational time $t_{\text{tot.}}$, as well as the average computational time per iteration for the data-fidelity step $t_{\text{dat.}}$ and the regularization step $t_{\text{reg.}}$. Since all algorithms used a single CPU core and/or a single GPU, the computational time is reported in seconds.

4.4. Experiment in generic image & data settings

The experimental setup is generic in that all parameters characterizing the ground-truth images and the observations were uniformly randomized, as per the considerations of the training (see Section 3 for details). Test datasets were created using four real radio images: the giant radio galaxies 3C353 (NRAO Archives) and Messier 106 (Shimwell et al. 2022), and the radio galaxy clusters Abell 2034 and PSZ2 G165.68+44.01 (Botteon et al. 2022). From each radio image, 50 ground-truth images of size $N = 512 \times 512$ with varying dynamic range were generated following the pre-processing procedure used for the training datasets. Different Fourier sampling patterns were used to simulate the RI data associated with the ground-truth images. The resulting test dataset comprises 200 inverse problems.

The parameter choice of the benchmark algorithms is as follows. The uSARA parameter was set to two times the heuristic value. The AIRI parameter was fixed to the heuristic value for all RI data, except for those simulated from the radio image of 3C353, where 3 times the heuristic value was used. CLEAN parameters were set to ensure *deep* cleaning with minimal compromise on the algorithm stability². In particular, both auto-masking and threshold parameters were set to 1.5 and 0.5 times the estimate of the noise level, respectively.

Numerical reconstruction results of the different algorithms are summarized in Table 2. The reported values correspond to the average across the 200 inverse problems. SNR and logSNR metrics showcase the overall superior performance of R2D2, and both realizations of R3D3 (R3D3^{3L}, R3D3^{6L}), with more than 3 dB improvement over AIRI and uSARA. CLEAN results indicate sub-optimal performance with more than 10 dB lower values than uSARA and AIRI. This is somewhat expected since CLEAN reconstruction is limited in both resolution (due to the smoothing with the CLEAN beam) and dynamic range (due to the addition of the

residual dirty image). Furthermore, it does not inherently enforce the nonnegativity constraint in the context of intensity imaging. Focusing on R2D2 and R3D3, they both showcase comparable SNR and logSNR values, with incremental improvement achieved by the latter. Specific to R3D3, its deeper realization R3D3^{6L} provides an incremental improvement over R3D3^{3L} in SNR of about 0.2 dB on average, and exhibits similar results in logSNR. These results showcase the consistency of the reconstruction quality of R3D3 across variations of the depth of its network components, and more generally the consistency of the imaging precision of the R2D2 algorithm in its two incarnations.

For the end-to-end DNNs, the fully data-driven U-Net provides a sub-optimal reconstruction, where both SNR and logSNR are at least 10 dB lower than R2D2, demonstrating the interest of the series structure underpinning the proposed deep-learning approach to enable high imaging precision. Interestingly, both realizations of R2D2-Net (R2D2-Net^{3L}, R2D2-Net^{6L}) are able to achieve good reconstruction. On the one hand, R2D2-Net^{3L} delivers a 2 dB higher SNR value than both AIRI and uSARA, and a lower logSNR value by the same amount. On the other hand, R2D2-Net^{6L} outperforms both AIRI and uSARA by more than 2 dB in both SNR and logSNR, showcasing the benefit of increasing the number of layers in the data model-informed R2D2-Net to enhance the dynamic range of the estimated images. Although the deeper realization of R2D2-Net reduces the gap in the reconstruction quality with R3D3, the latter provides 1 dB improvement in logSNR, confirming the interest of the series structure of the proposed algorithm.

Numerical analysis of the image-domain data fidelity obtained by the imaging algorithms (see Table 2) indicates that CLEAN delivers the highest data fidelity, enabled by the thorough cleaning functionality in WSClean. Both realizations of R3D3, uSARA and AIRI obtain low $\bar{\sigma}_{\text{res.}}$ values that are comparable to CLEAN. R2D2, however, provides a value nearly twice as high as R3D3 realizations, suggesting lower data fidelity. These results showcase the importance of the architecture of the network components of the R2D2 algorithm to enhance the data fidelity of its reconstruction.

The Python and MATLAB implementations of R2D2 and R3D3 yield consistent results, with marginal difference in logSNR values induced by their different NUFFT implementations. We note that sparse matrix interpolation NUFFT implementation is also available in Python.

² WSClean command: `wsclean -niter 2000000 -weight briggs 0 -j 1 -multiscale -mgain 0.8 -gain 0.1 -auto-threshold 0.5 -auto-mask 1.5 -padding 2`

Table 2. Results of the experiment in generic image and data settings obtained by the imaging algorithms. Reconstruction quality metrics are SNR, logSNR, and $\bar{\sigma}_{\text{res.}}$. Computational details are presented in terms of: the total number of iterations (I), the total reconstruction time ($t_{\text{tot.}}$), the average time per iteration of the data fidelity step ($t_{\text{dat.}}$) and the regularization step ($t_{\text{reg.}}$), and the allocated resources in GPUs (n_{gpu}) and CPU cores (n_{cpu}). Reported values are computed as averages over 200 inverse problems. Information on the programming languages underlying the algorithms implementations is provided.

	SNR	logSNR	$\bar{\sigma}_{\text{res.}}$	I	$t_{\text{tot.}}$	$t_{\text{dat.}}$	$t_{\text{reg.}}$	n_{cpu}	n_{gpu}	Programming language
	$\pm$ std (dB)	$\pm$ std (dB)	$\pm$ std ($\times 1\text{E-}4$)	$\pm$ std	$\pm$ std (s)	$\pm$ std (s)	$\pm$ std (s)			
CLEAN	13.6 $\pm$ 3.6	10.3 $\pm$ 3.5	5.1 $\pm$ 5.2	9 $\pm$ 1	65.9 $\pm$ 18.8	3.8 $\pm$ 1.1	3.8 $\pm$ 1.0	1	-	C++
uSARA	30.8 $\pm$ 1.9	21.9 $\pm$ 3.3	6.5 $\pm$ 8.2	1103 $\pm$ 373	4184.2 $\pm$ 1548.9	1.4 $\pm$ 0.7	2.3 $\pm$ 1.1	1	-	MATLAB
AIRI	31.3 $\pm$ 2.3	21.9 $\pm$ 4.4	6.4 $\pm$ 8.0	5000 $\pm$ 0.0	3478.8 $\pm$ 1531.4	0.66 $\pm$ 0.3	0.03 $\pm$ 0.2	1	1	MATLAB
U-Net	20.5 $\pm$ 2.7	6.6 $\pm$ 3.3	777.5 $\pm$ 467.4	1	2.2 $\pm$ 0.6	-	2.2 $\pm$ 0.6	-	1	MATLAB
	20.5 $\pm$ 2.7	6.6 $\pm$ 3.3	777.6 $\pm$ 467.4	1	1.1 $\pm$ 0.1	-	1.1 $\pm$ 0.1	-	1	Python
R2D2-Net ^{3L}	32.6 $\pm$ 1.5	19.6 $\pm$ 5.5	21.5 $\pm$ 13.8	1	2.7 $\pm$ 0.4	-	2.7 $\pm$ 0.4	-	1	MATLAB
	32.6 $\pm$ 1.5	19.6 $\pm$ 5.4	21.5 $\pm$ 13.8	1	1.1 $\pm$ 0.1	-	1.1 $\pm$ 0.1	-	1	Python
R2D2-Net ^{6L}	33.7 $\pm$ 1.7	24.0 $\pm$ 4.7	9.3 $\pm$ 6.9	1	3.8 $\pm$ 1.5	-	3.8 $\pm$ 1.5	-	1	MATLAB
	33.7 $\pm$ 1.7	24.0 $\pm$ 4.7	9.3 $\pm$ 7.0	1	1.1 $\pm$ 0.1	-	1.1 $\pm$ 0.1	-	1	Python
R2D2	33.7 $\pm$ 1.5	25.1 $\pm$ 4.9	13.5 $\pm$ 46.9	15	12.2 $\pm$ 2.8	0.4 $\pm$ 0.2	0.2 $\pm$ 0.07	1	1	MATLAB
	33.7 $\pm$ 1.5	25.0 $\pm$ 4.9	13.5 $\pm$ 46.9	15	18.6 $\pm$ 5.9	1.1 $\pm$ 0.4	0.1 $\pm$ 0.1	1	1	Python
					2.9 $\pm$ 0.3	0.05 $\pm$ 0.01	0.1 $\pm$ 0.2	-	1	Python
R3D3 ^{3L}	33.8 $\pm$ 1.4	25.3 $\pm$ 4.6	7.6 $\pm$ 7.6	7	9.4 $\pm$ 1.5	0.5 $\pm$ 0.2	0.5 $\pm$ 0.1	1	1	MATLAB
	33.8 $\pm$ 1.4	25.3 $\pm$ 4.6	7.6 $\pm$ 7.6	7	9.8 $\pm$ 4.3	1.1 $\pm$ 0.4	0.2 $\pm$ 0.3	1	1	Python
					1.9 $\pm$ 0.5	0.05 $\pm$ 0.02	0.2 $\pm$ 0.3	-	1	Python
R3D3 ^{6L}	34.0 $\pm$ 1.6	25.3 $\pm$ 4.7	7.9 $\pm$ 7.8	8	15.2 $\pm$ 2.1	0.5 $\pm$ 0.2	1.3 $\pm$ 0.6	1	1	MATLAB
	34.0 $\pm$ 1.6	25.3 $\pm$ 4.7	7.9 $\pm$ 7.8	8	11.3 $\pm$ 3.9	1.1 $\pm$ 0.4	0.2 $\pm$ 0.3	1	1	Python
					2.2 $\pm$ 0.3	0.05 $\pm$ 0.02	0.2 $\pm$ 0.3	-	1	Python

Note. For enhanced readability, a color-coding is considered to categorize the performance of the imaging algorithms: high in green, sub-optimal in red. Specific to CLEAN, the reported number of iterations corresponds to the number of “major cycles” reached at convergence. Two inverse problems from the test dataset diverged and are therefore excluded from the reported results. These instances of instability in the CLEAN implementation could potentially stem from the *deep* cleaning.

Identifier	ρ_{DR}	$t_{\text{obs-A}}$ (hr)	$t_{\text{obs-C}}$ (hr)	n_{freq}	ρ_{freq}
I	10^5	5.5	1.1	1	1
II	5×10^3	5.5	1.1	1	1
III	10^5	5.5	1.1	4	2
IV	10^5	9.0	2.7	1	1

Table 3. Details of Experiments I–IV in terms of: the value of the dynamic range of the ground-truth images (ρ_{DR}), the total duration of the observations with VLA configurations A and C ($t_{\text{obs-A}}, t_{\text{obs-C}}$), the number of frequency channels (n_{freq}), and the ratio between the highest and lowest frequency channels (ρ_{freq}). With Experiment I taken as the reference, the different settings were designed to study the impact of each parameter in two different regimes.

We note that sparse matrix interpolation NUFFT implementation is also available in Python. This was validated on the same experiments and confirmed to provide identical results to the MATLAB version. It is however slower than its table-based interpolation counterpart.

4.5. Experiments in specific image & data settings

Considering the four radio images used in the previous experiment, we designed four experiments (I–IV) to assess the performance of both incarnations of the R2D2 algorithm (i) in contrasting regimes of the dynamic range of the sought radio images, and (ii) in varying observation settings, in terms of the total observation time and the bandwidth of the frequency channels (under the assumption of radio emission with flat spectra). The simulation parameter choice in the four experiments is listed in Table 3. In all experiments, the dynamic range of the ground-truth images is fixed. Therefore, 1 ground truth image per radio image was generated. For each experiment, we simulated 25 RI datasets with varying Fourier sampling patterns per ground truth image, resulting in a total of 100 inverse problems. Imaging with the benchmark algorithms, in particular their parameter choice, followed the same considerations of the generic experiment.

Numerical results of Experiments I–IV are provided in Figure 6, comprising graphs of the evolution of the SNR, logSNR, and $\bar{\sigma}_{\text{res.}}$ metrics across the iterations of R2D2 and R3D3. In these graphs, the numerical values at the first iteration of R2D2, R3D3^{3L}, and R3D3^{6L} correspond to the results of the end-to-end DNNs, U-Net, R2D2-Net^{3L}, and R2D2-Net^{6L}, respectively. Final results of CLEAN, AIRI, and uSARA are indicated with horizontal lines.

In Experiment I, we consider a single frequency-channel observation, a relatively short observation time totaling 6.6 hr with both VLA configurations, and an extremely high-dynamic range regime with $\rho_{\text{DR}} = 10^5$.

This scenario will serve as a reference in our analysis. We first study the performance of R2D2 and R3D3 under different regimes of the dynamic range of the ground-truth images. To this aim, we consider Experiment II, characterized by a relatively low-dynamic range regime with $\rho_{\text{DR}} = 5 \times 10^3$. In terms of SNR and logSNR, both R3D3^{3L} and R3D3^{6L} consistently outperform all imaging algorithms, showing a wider gap in both SNR and logSNR in the high-dynamic range regime. Although trailing behind AIRI in the low-dynamic range regime, R2D2 outperforms the benchmark algorithms in the high-dynamic range regime. Generally, both realizations of R3D3 converge faster than R2D2, with the SNR and logSNR metrics saturating more promptly in the low-dynamic range regime compared to the high-dynamic range regime.

We study the impact of multi-frequency acquisition and longer observation duration through Experiments III and IV, both enabling more Fourier information of the sought images than Experiment I, by combining wideband observations in the former, and increasing the combined total observation time to 11.7 hr in the latter. When compared to Experiment I, both incarnations of the R2D2 algorithm obtain higher SNR and logSNR values, outperforming the benchmark algorithms. Moreover, their metrics saturate more promptly. These experiments showcase consistency in the results of the R2D2 algorithm.

Concerning data fidelity, in the low-dynamic range regime (Experiment II), all algorithms obtain similar $\bar{\sigma}_{\text{res.}}$ values, suggesting comparable data fidelity. In the high-dynamic range regime (Experiments I, III–IV), AIRI and uSARA consistently exhibit the lowest $\bar{\sigma}_{\text{res.}}$ values, thus superior data fidelity, whereas CLEAN obtains 2 to 4 times higher values. Both R3D3 realizations deliver similar results, which are 3 to 5 times higher than AIRI and uSARA. R2D2, however, delivers the highest $\bar{\sigma}_{\text{res.}}$ values among all algorithms, indicating the lowest data fidelity. Yet, when compared to R3D3, R2D2 results are only 20% to 60% higher.

Reconstructed images of selected test sets from Experiments I–IV are displayed in Figures 7–10, respectively. The scrutinized radio images exhibit different morphology, from extended to compact and point-like structure. Since both realizations of R3D3 yield comparable results, we consider reconstructed images R3D3^{6L} in this visual examination. One can observe that R2D2, R3D3, AIRI and uSARA reconstructions are comparable. CLEAN reconstruction, by design, provides a smoother depiction of the radio emission and a noise-like background. Focusing on the end-to-end DNNs, U-Net delivers an overly-smooth representation, and in some

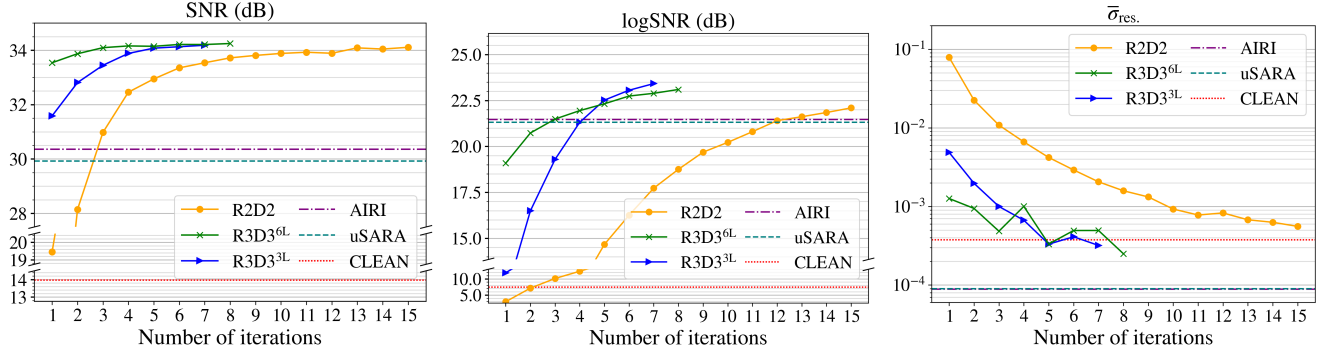
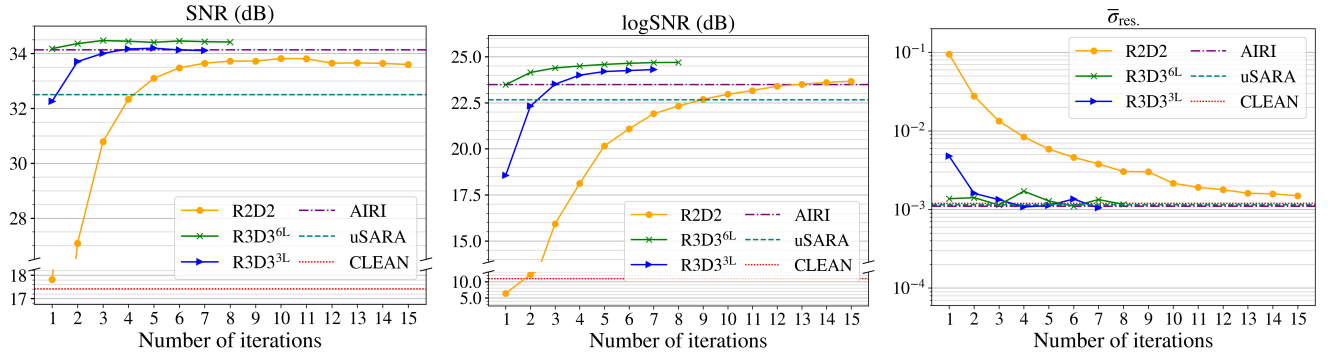
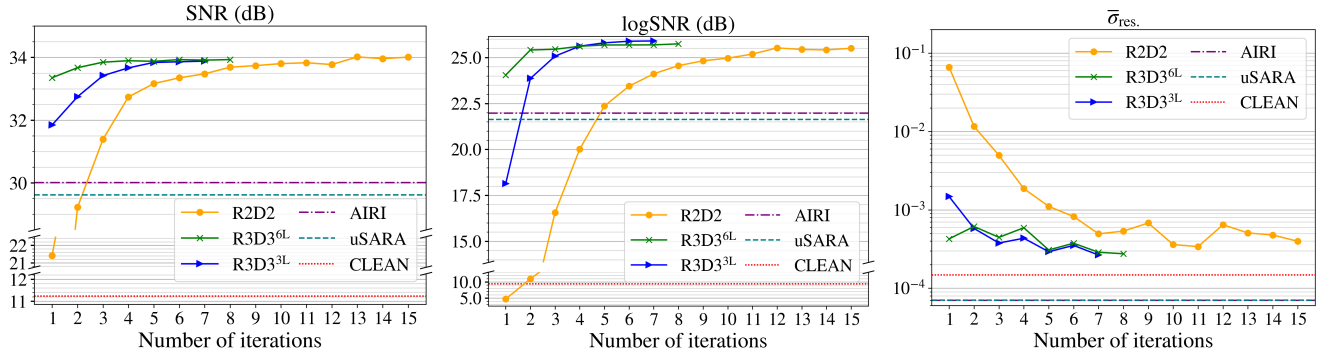
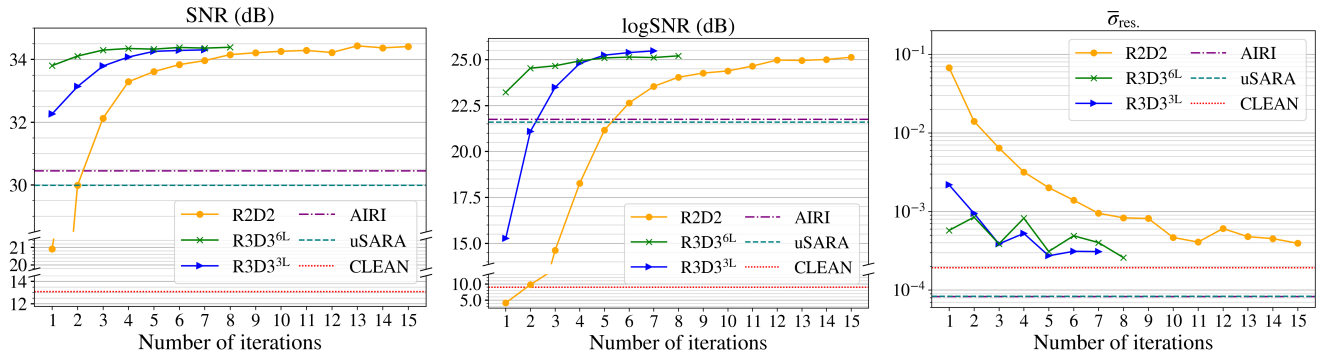
Experiment I: $\rho_{\text{DR}} = 10^5$, $(t_{\text{obs-A}}, t_{\text{obs-C}}) = (5.5, 1.1)$ hr, $n_{\text{freq}} = 1$ Experiment II: $\rho_{\text{DR}} = 5 \times 10^3$, $(t_{\text{obs-A}}, t_{\text{obs-C}}) = (5.5, 1.1)$ hr, $n_{\text{freq}} = 1$ Experiment III: $\rho_{\text{DR}} = 10^5$, $(t_{\text{obs-A}}, t_{\text{obs-C}}) = (5.5, 1.1)$ hr, $\rho_{\text{freq}} = 2$, $n_{\text{freq}} = 4$ Experiment IV: $\rho_{\text{DR}} = 10^5$, $(t_{\text{obs-A}}, t_{\text{obs-C}}) = (9, 2.7)$ hr, $n_{\text{freq}} = 1$

Figure 6. Progress of the reconstruction quality (left column: SNR, middle column: logSNR) and data fidelity (right column: $\bar{\sigma}_{\text{res.}}$) across the iterations of R2D2 and both realizations of R3D3. Values of the different metrics, achieved by convergence by the benchmark algorithms uSARA, AIRI and CLEAN, are reported via horizontal lines. From top to bottom: results of Experiments I-IV. In each scenario, the reported values are averages over a test dataset composed of 100 inverse problems.

cases, hallucination structure (e.g. the extended emission in the bottom right quadrant in the reconstruction of the cluster PSZ2 G165.68+44.0 in Figure 9, and the compact source near Messier 106 in Figure 10). Such artifacts are completely removed in the R2D2 reconstruction, highlighting its robustness, induced by its iterative nature. R2D2-Net achieves a high-quality reconstruction, confirming the numerical analysis. In contrast with U-Net, R2D2-Net does not exhibit hallucination structure, owing to its integrated data consistency layers. The examination of the residual dirty images shows that R2D2 and R3D3 consistently present discernible structures around the pixel positions of the brightest emission in the radio images, particularly noticeable in the high-dynamic range regime (see bottom row of Figures 8–10). Both AIRI and uSARA achieve high data fidelity with noise-like residual dirty images. CLEAN obtains residual dirty images comparable to AIRI and uSARA, except for the selected test set from Experiment I. Fine-tuning manually the cleaning depth for this specific test set could improve the results. These findings are in general agreement with the numerical analysis.

4.6. Computational performance

Computational details of the different algorithms in terms of the number of iterations and the imaging computational time are presented in Table 2. A summary of the allocated resources and the programming languages underlying their implementations are also provided. Both incarnations of the proposed algorithm call for a small number of iterations. In contrast, uSARA and AIRI required orders of magnitude more iterations, as expected. As such, the MATLAB implementation of R2D2 and R3D3 delivers reconstructions in seconds, when AIRI and uSARA take around an hour or more. In comparison with CLEAN, both R2D2 and R3D3 call for similar number of passes through the data. Yet, multiscale CLEAN takes over a minute on average, due to its iterative minor cycles, and the slow NUFFT implementation for the relatively small data and image sizes considered in this work. In its hybrid resource allocation setting, the Python version of R2D2 and R3D3 delivers comparable computational time to their MATLAB version. Remarkably, when fully executed on a GPU, it enables faster reconstructions, with an average reconstruction time below 1 second.

Acknowledging the differences in the programming languages and computing hardware of the different algorithms, the reported numbers are indicative only. It is worth noting that both uSARA and AIRI could benefit from additional computational resources for the parallelization of their denoising operators, hence a faster

reconstruction. Furthermore, the widely-used WSClean is optimized for large image and data sizes. Also, in the high-dimensional regime of the modern telescopes, uSARA and AIRI demonstrated a substantially reduced computational gap with CLEAN (Dabbech et al. 2022; Wilber et al. 2023a,b). Last but not least, an in-depth investigation of the practical scalability of R2D2 and R3D3 is warranted, particularly given the GPU memory limitations, possibly requiring advanced data and image decomposition approaches at large scale. However, the conclusion stands that the imaging time of R2D2 and R3D3 is significantly lower than AIRI and uSARA thanks to their inherent small number of iterations, and is possibly faster than CLEAN thanks to their DNN inference that is faster than CLEAN’s iterative minor cycles.

5. CONCLUSIONS

We have presented the R2D2 algorithm, a novel deep-learning technique for high-dynamic range imaging in radio astronomy. The R2D2 algorithm is underpinned by a series of end-to-end DNNs trained sequentially, each taking as input the image estimate from the previous iteration alongside its corresponding residual dirty image. The R2D2 reconstruction is formed as the sum of iteratively learned residual images which are outputs of its underpinning DNN series.

We have provided an in-depth description of the R2D2 paradigm, featuring its two incarnations distinguished by their distinct network components. The first incarnation uses the well-known U-Net architecture, and is simply referred to as R2D2. The second uses a more advanced architecture dubbed R2D2-Net, obtained by unrolling the R2D2 iterative scheme itself. Given its nesting structure, we refer to this incarnation as R3D3. Taking a telescope-specific training approach, R2D2 and R3D3 have been implemented for the VLA. Their imaging precision capability in simulation, across a variety of image and data settings, has been demonstrated against uSARA and AIRI. Owing to their small number of iterations, and fast DNN inference, R2D2 and R3D3 have demonstrated a significant acceleration over uSARA and AIRI. Compared to CLEAN, they require similar number of passes through the data, suggesting comparable computational time independently of their implementation. Propelled by its advanced data model-informed networks, R3D3 has shown similar or higher imaging precision than R2D2 with fewer iterations, potentially enabling further scalability to large data sizes. Of note, the R2D2/R3D3 structure is seen here as parameter-free with the number of required networks set at the training stage, offering an automation advantage over algo-

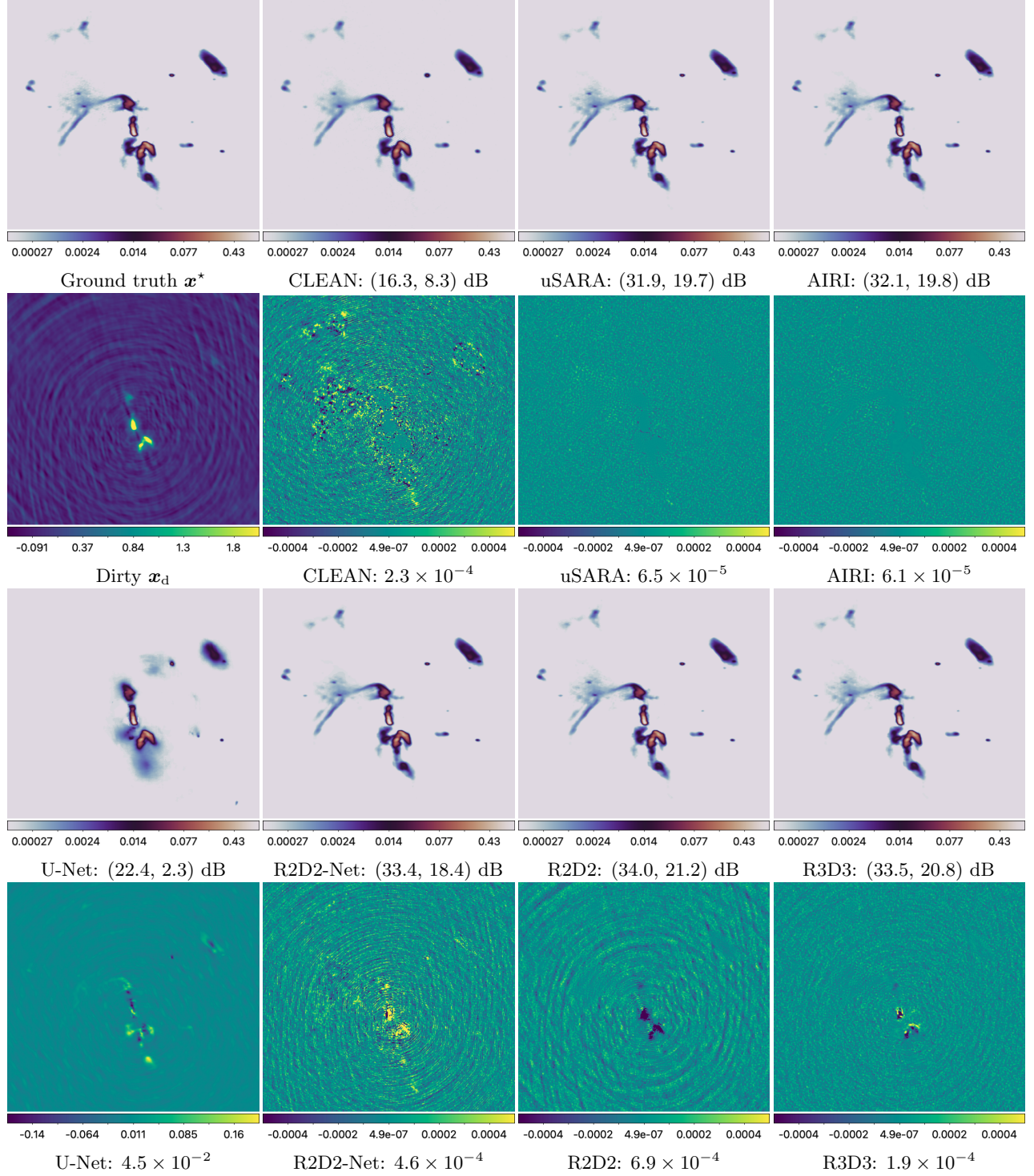


Figure 7. Experiment I: reconstructions results of the galaxy cluster Abell 2034. The first and third rows show the ground truth image and the estimated images obtained by the imaging algorithms (logarithmic scale). The second and fourth row show the dirty image and the corresponding residual dirty images (linear scale). For R3D3, we showcase the results of its realization with $J = 6$ layers in its R2D2-Net components. Values of (SNR, logSNR) are reported below estimated images. $\bar{\sigma}_{\text{res}}$ values are indicated below the residual dirty images.

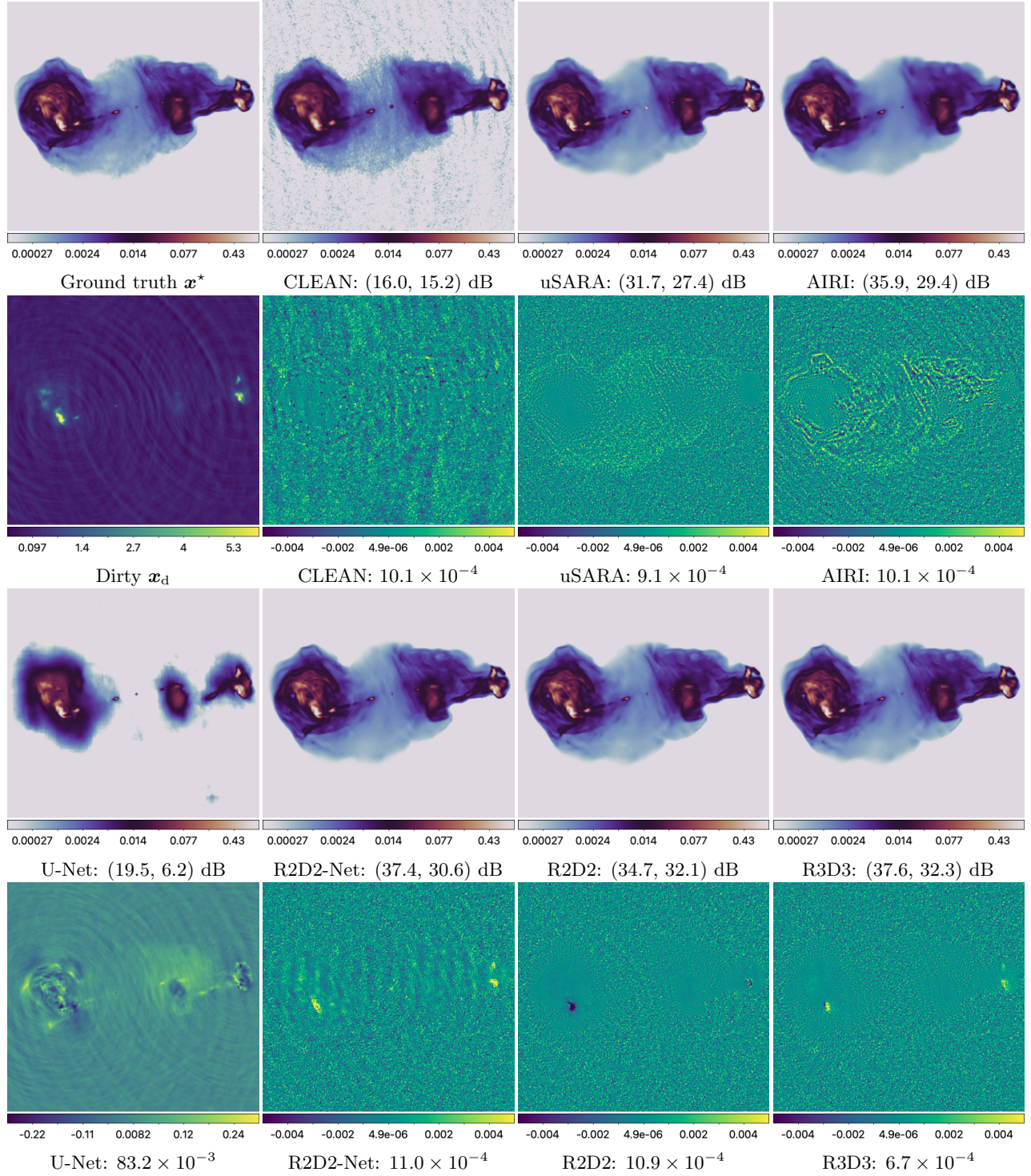


Figure 8. Experiment II: reconstructions results of the radio galaxy 3C353. The first and third rows show the ground truth image and the estimated images obtained by the imaging algorithms (logarithmic scale). The second and fourth row shows the dirty image and the corresponding residual dirty images (linear scale). For R3D3, we showcase the results of its realization with $J = 6$ layers in its R2D2-Net components. Values of (SNR, logSNR) are reported below estimated images. The σ_{res} values are indicated below the residual dirty images.

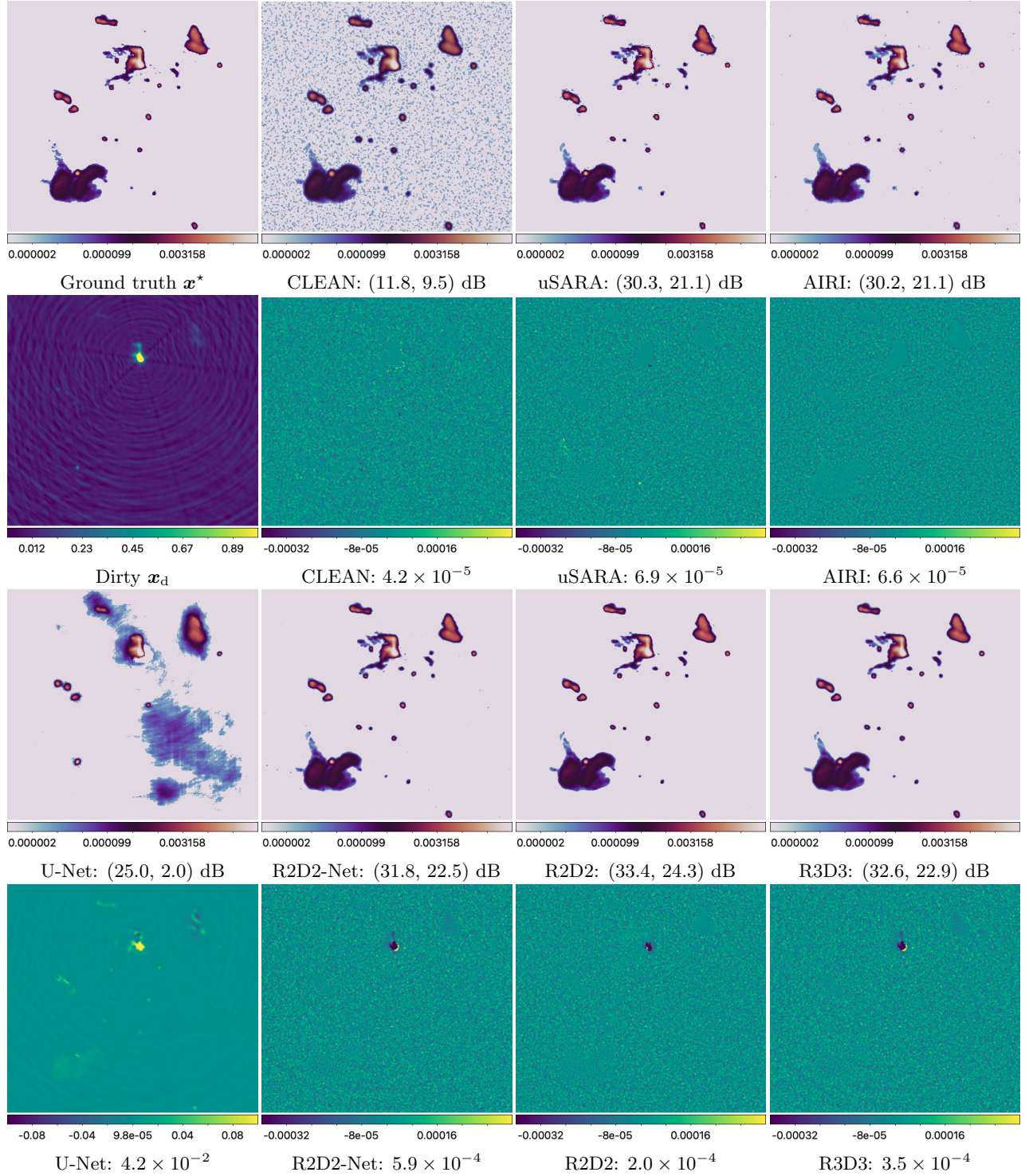


Figure 9. Experiment III: reconstructions results of the galaxy cluster PSZ2 G165.68+44.01. The first and third rows show the ground truth image and the estimated images obtained by the imaging algorithms (logarithmic scale). The second and fourth row shows the dirty image and the corresponding residual dirty images (linear scale). For R3D3, we showcase the results of its realization with $J = 6$ layers in its R2D2-Net components. Values of (SNR, logSNR) are reported below estimated images. The $\bar{\sigma}_{\text{res}}$ values are indicated below the residual dirty images.

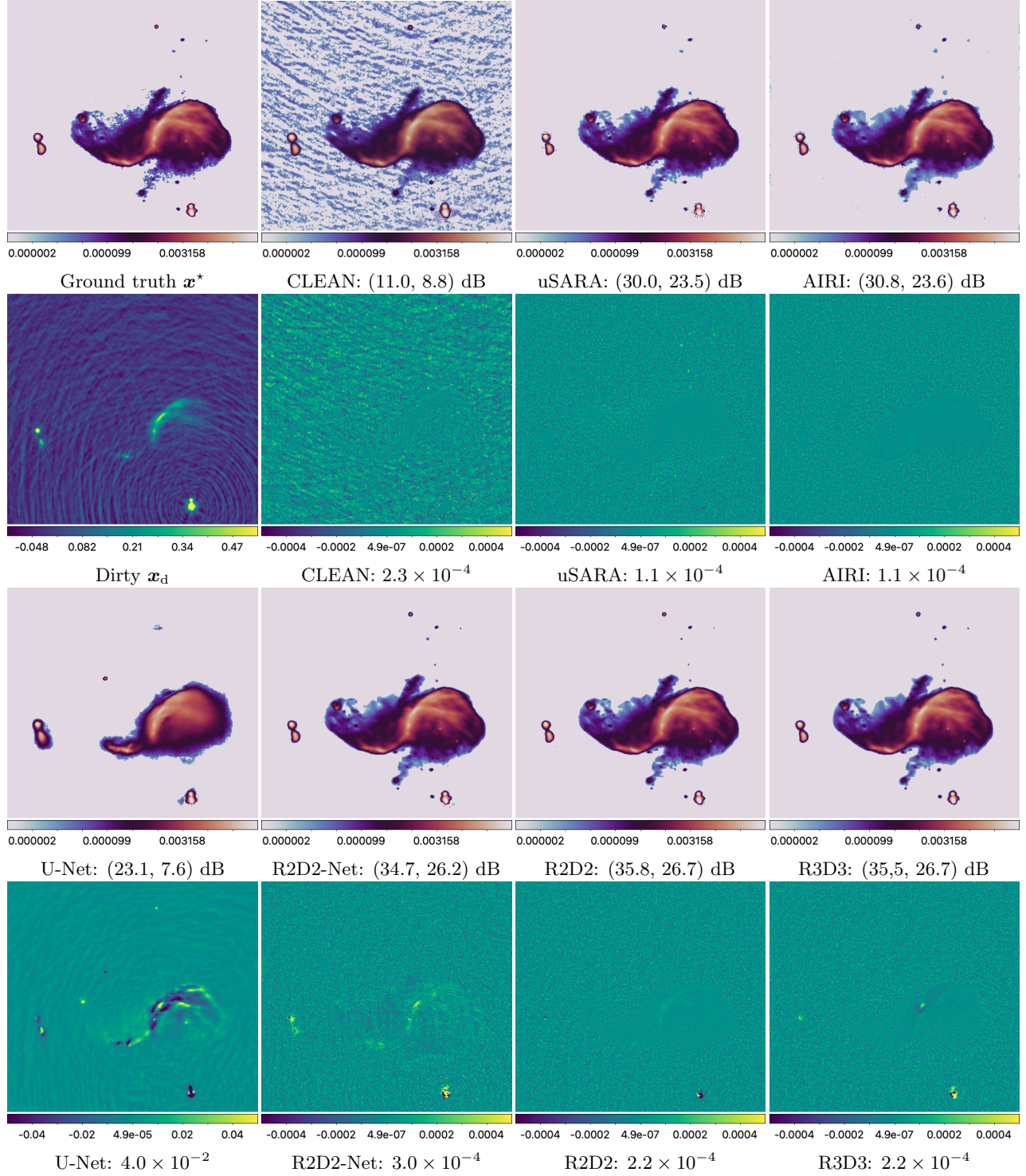


Figure 10. Experiment IV: reconstructions results of the radio galaxy Messier 106. The first and third rows show the ground truth image and the estimated images obtained by the imaging algorithms (logarithmic scale). The second and fourth row shows the dirty image and the corresponding residual dirty images (linear scale). For R3D3, we showcase the results of its realization with $J = 6$ layers in its R2D2-Net components. Values of (SNR, logSNR) are reported below estimated images. The $\bar{\sigma}_{\text{res}}$ values are indicated below the residual dirty images.

rithms requiring parameter fine-tuning. Thus, the R2D2 paradigm is opening the door to fast, possibly real-time, high-resolution high-dynamic range RI imaging.

Future developments are warranted to improve both imaging precision and scalability of the R2D2 algorithm. Firstly, novel core DNN architectures need to be explored to mitigate the residual structure observed in the back-projected residual data. Secondly, a generalized training approach, accommodating yet a wider variety of imaging settings (e.g., varying pixel resolutions and data-weighting schemes, an all-telescopes-encompassing setting) should be investigated. Thirdly, image-faceting procedures (e.g., [Thouvenin et al. 2023](#)) should be deployed at both training and inference, to handle the large image dimensions of interest to SKA and its precursor instruments, especially given the memory limitations of currently available GPUs. The iterative nature of the R2D2 algorithm, in particular its accurate computation of the back-projected data residuals, presents an opportunity for the self-correction of any faceting artifacts. Finally, advanced data decomposition approaches should be investigated at large scale, for an in-depth study of the practical scalability of the R2D2 algorithm.

DATA AVAILABILITY

R2D2 codes are available alongside the AIRI and uSARA codes in the [BASPLib](#) code library on GitHub. BASPLib is developed and maintained by the Biomedical and Astronomical Signal Processing Laboratory

([BASP](#)). R2D2 is available in both Python and MATLAB implementations. The VLA-trained R2D2 and R3D3 DNN series are available in both PyTorch and ONNX formats in the dataset [Aghabiglou et al. \(2024\)](#).

Images used to generate training, validation, and testing datasets are sourced as follows. Optical astronomy images are gathered from NOIRLab/NSF/AURA/H.Schweiker/WIYN/T.A.Rector (University of Alaska Anchorage). Medical images are obtained from the NYU fastMRI Initiative database ([Zbontar et al. 2018](#); [Knoll et al. 2020](#)). Radio astronomy images are obtained from the NRAO Archives, LOFAR HBA Virgo cluster survey ([Edler et al. 2023](#)), and LoTSS-DR2 survey ([Shimwell et al. 2022](#)).

Software: WSClean ([Offringa et al. 2014](#); [Offringa & Smirnov 2017](#)), Meqtrees ([Noordam & Smirnov 2010](#)), PyTorch ([Paszke et al. 2019](#)), TorchKbNufft ([Muckley et al. 2020](#));

- 1 The authors thank Yiwei Chen for the insightful discus-
- 2 sions on DNN architectures, and Chao Tang for his assis-
- 3 tance in building training datasets. The research of AA,
- 4 AD and YW was supported by the UK Research and
- 5 Innovation under the EPSRC grant EP/T028270/1 and
- 6 the STFC grant ST/W000970/1. The research was con-
- 7 ducted using Cirrus, a UK National Tier-2 HPC Service
- 8 at EPCC funded by the University of Edinburgh and
- 9 EPSRC (EP/P020267/1). The authors thank Adrian
- 10 Jackson for related support.

REFERENCES

- Aghabiglou, A., Chu, C. S., A., Dabbech, A., & Wiaux, Y. 2024, R2D2 deep neural network series for radio-interferometric imaging [Data set], doi: [10.17861/99cbe654-5071-4625-b59d-a26c790cbeb4](#)
- Aghabiglou, A., Terris, M., Jackson, A., & Wiaux, Y. 2023, in *Proc. IEEE ICASSP 2023*, 1–5
- Arras, P., Frank, P., Leike, R., Westermann, R., & Enßlin, T. A. 2019, *A&A*, 627, A134
- Bhatnagar, S., & Cornwell, T. 2004, *A&A*, 426, 747
- Botteon, A., Shimwell, T., Cassano, R., et al. 2022, *A&A*, 660, A78
- Briggs, D. 1995, in *AAS Meeting Abstracts*, Vol. 187, 112–02
- Cai, X., Pereyra, M., & McEwen, J. D. 2018, *MNRAS*, 480, 4154
- Carrillo, R. E., McEwen, J. D., & Wiaux, Y. 2012, *MNRAS*, 426, 1223
- Chan, S. H., Wang, X., & Elgendy, O. A. 2017, *IEEE Trans. Comput. Imaging*, 3, 84
- Connor, L., Bouman, K. L., Ravi, V., & Hallinan, G. 2022, *MNRAS*, 514, 2614
- Cornwell, T. J. 2008, *IEEE J. Sel. Top. Signal Process.*, 2, 793
- Dabbech, A., Aghabiglou, A., Chu, C. S., & Wiaux, Y. 2024, preprint [arXiv:2309.03291](#)
- Dabbech, A., Ferrari, C., Mary, D., et al. 2015, *A&A*, 576, A7
- Dabbech, A., Onose, A., Abdulaziz, A., et al. 2018, *MNRAS*, 476, 2853
- Dabbech, A., Terris, M., Jackson, A., et al. 2022, *ApJL*, 939
- Edler, H., de Gasperin, F., Shimwell, T., et al. 2023, *A&A*, 676, A24
- Fessler, J., & Sutton, B. 2003, *IEEE Trans. Signal Process.*, 51, 560

- Garsden, H., Girard, J., Starck, J.-L., et al. 2015, *MNRAS Letters*, 575, A90
- Hauptmann, A., Lucka, F., Betcke, M., et al. 2018, *IEEE Trans. Med. Imaging*, 37, 1382
- Högbom, J. 1974, *A&AS*, 15, 417
- Hotan, A., Bunton, J., Chippendale, A., et al. 2021, *PASA*, 38, e009
- Jonas, J. 2016, *Proc. Sci.*, MeerKAT Science: On the Pathway to the SKA, Sissa Trieste
- Junklewitz, H., Bell, M., Selig, M., & Enßlin, T. 2016, *A&A*, 586, A76
- Kamilov, U. S., Bouman, C. A., Buzzard, G. T., & Wohlberg, B. 2023, *IEEE Signal Process. Mag.*, 40, 85
- Knoll, F., Zbontar, J., Sriram, A., et al. 2020, *Radiol. Artif. Intell.*, 2, e190007
- Labate, M. G., Waterson, M., Alachkar, B., et al. 2022, *JATIS*, 8, 011024
- LeCun, Y., Bengio, Y., & Hinton, G. 2015, *Nature*, 521, 436
- Monga, V., Li, Y., & Eldar, Y. C. 2021, *IEEE Signal Process. Mag.*, 38, 18
- Muckley, M. J., Stern, R., Murrell, T., & Knoll, F. 2020, in *ISMRM Workshop on Data Sampling & Image Reconstruction*
- Noordam, J. E., & Smirnov, O. M. 2010, *A&A*, 524, A61
- Offringa, A., McKinley, B., Hurley-Walker, N., et al. 2014, *MNRAS*, 444, 606
- Offringa, A. R., & Smirnov, O. 2017, *MNRAS*, 471, 301
- Onose, A., Carrillo, R. E., Repetti, A., et al. 2016, *MNRAS*, 462, 4314
- Onose, A., Dabbech, A., & Wiaux, Y. 2017, *MNRAS*, 469, 938
- Paszke, A., Gross, S., Massa, F., et al. 2019, preprint [arXiv:1912.01703](https://arxiv.org/abs/1912.01703)
- Repetti, A., & Wiaux, Y. 2020, in *Proc. IEEE ICASSP 2020*, 1434–1438
- Ronneberger, O., Fischer, P., & Brox, T. 2015, in *MICCAI 2015*, 234–241
- Schwab, F. R. 1984, *AJ*, 89, 1076
- Shimwell, T., Hardcastle, M., Tasse, C., et al. 2022, *A&A*, 659, A1
- Swart, G. P., Dewdney, P. E., & Cremonini, A. 2022, *JATIS*, 8, 011021
- Terris, M., Dabbech, A., Tang, C., & Wiaux, Y. 2022, *MNRAS*, 518, 604
- Terris, M., Tang, C., Jackson, A., & Wiaux, Y. 2023, preprint [arXiv:2312.07137](https://arxiv.org/abs/2312.07137)
- Thompson, A. R., Moran, J. M., & Swenson, G. W. 2017, *Interferometry and synthesis in radio astronomy* (Springer Nature)
- Thouvenin, P.-A., Abdulaziz, A., Dabbech, A., Repetti, A., & Wiaux, Y. 2023, *MNRAS*, 521, 1
- van Haarlem, M. P., Wise, M. W., Gunst, A., et al. 2013, *A&A*, 556, A2
- Venkatakrishnan, S. V., Bouman, C. A., & Wohlberg, B. 2013, in *Proc. IEEE GlobalSIP 2013*, 945–948
- Wiaux, Y., Jacques, L., Puy, G., Scaife, A. M., & Vanderghynst, P. 2009, *MNRAS*, 395, 1733
- Wilber, A. G., Dabbech, A., Jackson, A., & Wiaux, Y. 2023a, *MNRAS*, 522, 5558
- Wilber, A. G., Dabbech, A., Terris, M., Jackson, A., & Wiaux, Y. 2023b, *MNRAS*, 522, 5576
- Zbontar, J., Knoll, F., Sriram, A., et al. 2018, preprint [arXiv:1811.08839](https://arxiv.org/abs/1811.08839)
- Zhang, K., Zuo, W., Chen, Y., Meng, D., & Zhang, L. 2017, *IEEE Trans. Image Process.*, 26, 3142
- Zhang, K., Li, Y., Liang, J., et al. 2023, *Mach. Intell. Res.*, 20, 822–836