Deep Unfolding of the DBFB Algorithm with Application to ROI CT Imaging with Limited Angular Density

Marion Savanier, Emilie Chouzenoux, Jean-Christophe Pesquet, Cyril Riddell

Deep Unfolding of the DBFB Algorithm with Application to ROI CT Imaging with Limited Angular Density

Marion Savanier, Emilie Chouzenoux, Jean-Christophe Pesquet, and Cyril Riddell

***Abstract*—This paper addresses the problem of image reconstruction for region-of-interest (ROI) computed tomography (CT). While model-based iterative methods can be used for such a problem, their practicability is often limited due to tedious parameterization and slow convergence. In addition, inadequate solutions can be obtained when the retained priors do not perfectly fit the solution space. Deep learning methods offer an alternative approach that is fast, leverages information from large data sets, and thus can reach high reconstruction quality. However, these methods usually rely on black boxes not accounting for the physics of the imaging system, and their lack of interpretability is often deplored. At the crossroads of both methods, unfolded deep learning techniques have been recently proposed. They incorporate the physics of the model and iterative optimization algorithms into a neural network design, leading to superior performance in various applications. This paper introduces a novel, unfolded deep learning approach called U-RDBFB designed for ROI CT reconstruction from limited data. Few-view truncated data are efficiently handled thanks to a robust non-convex data fidelity function combined with sparsity-inducing regularization functions. Iterations of a block dual forward-backward (DBFB) algorithm, embedded in an iterative reweighted scheme, are then unrolled over a neural network architecture, allowing the learning of various parameters in a supervised manner. Our experiments show an improvement over various state-of-the-art methods, including model-based iterative schemes, deep learning architectures, and deep unfolding methods.**

***Index Terms*—region-of-interest, computed tomography, angular sub-sampling, deep unfolding, forward-backward, iterative reweighted scheme**

I. Introduction

CT imaging is commonly used for diagnostic purposes and image guidance in interventional radiology and surgery. In most interventions, such as follow-up examinations of deployed stents and needle biopsies, only a small region of the patient is of interest. Irradiating only the ROI by X-rays involves focusing the X-ray beam with collimation techniques using radio-opaque blades before passing through the patient [1]. A focused irradiation results in a substantial reduction in patient dose [2]. The measurements (or projections) obtained from such acquisitions are truncated. The inverse problem of reconstructing an ROI from a set of truncated projections is an ill-posed problem often combined with angular sub-sampling for fast acquisitions. Previous research demonstrated the ability of model-based iterative reconstruction (MBIR) for CT reconstruction from few-view measurements [3], [4]. MBIR implements nonlinear iterative algorithms aiming at minimizing a penalized cost function. In CT, most works rely on total variation (TV) regularization [5]. Although they achieve a reduction in angular and cone-beam sub-sampling artifacts compared to analytical reconstruction methods, truncated projections still challenge MBIR [6].

To implement MBIR, one must choose the reconstruction grid, i.e., the support of the reconstructed area. When the grid matches the support of the ROI, the data will not agree with the reprojection of the ROI. When the reconstruction grid includes the support of the entire object [7], the reconstruction becomes computationally expensive and less stable due to the increase of unknowns for the same amount of data. Truncated data only allows a rough estimation of the exterior anatomical background, which can never be clinically acceptable. One, therefore, often considers an intermediate smaller grid size, bigger than the ROI but possibly less than the entire object [8]. This achieves a faster and more stable ROI reconstruction. An alternative is to reconstruct the entire object with large voxels and then subtract the reprojection of the exterior to the data before reconstructing only the ROI from the subtracted data [9], [10]. Both approaches produce a low-frequency approximation of the exterior of the ROI. When dense objects such as cables or needles are located outside of the ROI and, more importantly, outside of the chosen reconstruction grid, the approximation is poor, and reprojection of the objects increases the error in the data fidelity term. This means that the data should not be trusted equally but through a statistical analysis different from measurement noise. When angularly sub-sampled, they still produce streak artifacts over the ROI [6]. Thus, designing MBIR methods for ROI imaging with a low computational cost is still challenging. Preconditioning is often too restrictive for CT reconstruction when proximal algorithms are used [11]; hence heuristics have been proposed to accelerate TV-based methods. In [6], for example, substantial acceleration of the forward-backward algorithm is reached thanks to two modifications. The first modification uses the ramp filter to weigh their least-squares data fidelity term. Each gradient step then performs an approximate inversion of the forward

This work was supported by the European Research Council Starting Grant MAJORIS ERC-2019-STG-850925, the ANRT CIFRE Convention 2018/1587, and the ANR Research and Teaching Chair in Artificial Intelligence BRIDGEABLE.

E. Chouzenoux, J.-C. Pesquet and M. Savanier are with Univ. Paris-Saclay, CentraleSupélec, CVN, Inria, Gif-sur-Yvette, France (e-mail: first-name.lastname@centralesupelec.fr).

M. Savanier and C. Riddell are with GE Healthcare, Buc, France (e-mail: marion.savanier@gmail.com, cyril.riddell@ge.com).

projection, and each proximity step acts as a post-processing filter. The use of an approximate inverse of the measurement operator in the data fidelity term is also advocated in [12]–[14]. The second modification is the linear decrease of the regularization parameter's strength along with the iterations. Despite these modifications, the proximity operator of TV regularization does not have a closed form, so the method still requires many sub-iterations.

Convolutional neural networks (CNN) are an attractive alternative to MBIR due to their increased expressivity and fast inference. CNNs, and in particular the U-net [15], have already been used for removing sub-sampling streaks in CT reconstructions obtained using analytical methods [16], [17]. However, there are concerns about the lack of guarantees and capacity for generalization of post-processing CNNs, because these networks do not ensure data consistency [18]. Deep unfolding methods [19], [20] circumvent this issue by offering a way to include a priori information in a neural network. They have been applied to many fields of imaging, such as denoising [21], [22], deblurring [23], MRI reconstruction [24], and CT reconstruction from few-view data [25] Unfolding consists of untying each iteration of an optimization algorithm for MBIR, defining a set of learnable parameters, and training each iteration (or layer) in an end-to-end manner. Some authors allow the unfolding network to learn the optimization algorithm hyperparameters [23] as well as linear operators in the regularization, such as convolution kernels in ISTA-net [26] and in [27]. Others use CNNs to replace proximity operators, as in PD-net [25] and ADMM-net [24]. Deep unfolding networks automatically inherit from the feedback mechanism of MBIR for data consistency.

In this paper, we exploit the framework of deep unfolding for addressing ROI image reconstruction from few-view truncated measurements. We design an MBIR cost function, propose an iterative algorithm suitable for minimizing our cost function, and present an unfolding of this algorithm. Specifically:

- we formulate a penalized cost function combining a data fidelity term arising from the theory of M-estimators and a variant of total-variation regularization,
- we derive an iterative optimization algorithm combining an instance of the dual block forward-backward algorithm (DBFB) [28], also known as dual coordinate ascent technique, combined with an iterative reweighted scheme;
- we perform the unfolding of the above algorithm allowing the supervised learning of some parameters involved both in our cost function (e.g., adjoint of the linear operators) and in our optimization algorithm (e.g., step sizes);
- we show competitive performance against state-of-the-art for limiting sub-sampling artifacts from objects inside and outside the reconstruction grid.

Section II of this work introduces our notation, and Section III presents a mathematical formulation of ROI imaging. Section IV presents our choices for the data fidelity and regularization terms and introduces a convergent iterative algorithm to minimize the resulting cost function. We then explain how this algorithm is unfolded into a deep learning architecture, called U-RDBFB, in Section V. Section VI presents the training strategy for U-RDBFB architecture. Finally, Section VII shows and discusses the results of our experiments, which were conducted with both our unfolded networks and other approaches proposed in the literature.

II. Notation

Throughout the paper, we consider the Euclidean space $\mathbb{R}^L$ equipped with the standard scalar product $\langle \cdot, \cdot \rangle$, and the norm $\| \cdot \|$. Moreover, $|||\mathbf{L}|||$ denotes the spectral norm of squared matrix $\mathbf{L} \in \mathbb{R}^{L \times L}$. Let $\mathcal{S}_L^+$ be the set of symmetric positive definite matrices in $\mathbb{R}^{L \times L}$. For $\mathbf{Q} \in \mathcal{S}_L^+$, $\| \cdot \|_{\mathbf{Q}}$ denotes the $\mathbf{Q}$-weighted norm, i.e., for every $\mathbf{x} \in \mathbb{R}^L$, $\|\mathbf{x}\|_{\mathbf{Q}} = \sqrt{\langle \mathbf{x} \mid \mathbf{Q}\mathbf{x} \rangle}$. The diagonal matrix with diagonal entries equal to vector $\mathbf{z} = (z_l)_{\ell=1}^L \in \mathbb{R}^L$ is denoted $\mathrm{diag}((z_\ell)_{\ell=1}^L)$. The class of functions which are proper, convex, lower-semicontinuous, defined on $\mathbb{R}^L$ and taking values in $\mathbb{R} \cup \{+\infty\}$ is denoted by $\Gamma_0(\mathbb{R}^L)$. The proximity operator of $g \in \Gamma_0(\mathbb{R}^L)$ at $\mathbf{x} \in \mathbb{R}^L$ is uniquely defined as [29] $\mathrm{prox}_g(\mathbf{x}) = \underset{\mathbf{z} \in \mathbb{R}^L}{\mathrm{argmin}} \ \big(g(\mathbf{z}) + \frac{1}{2}\|\mathbf{x} - \mathbf{z}\|^2\big)$. The indicator function of a nonempty subset C of $\mathbb{R}^L$ is the function ι_C equal to 0 on C and $+\infty$ outside of C. If C is closed and convex, the projection onto C is denoted by proj_C.

III. ROI imaging

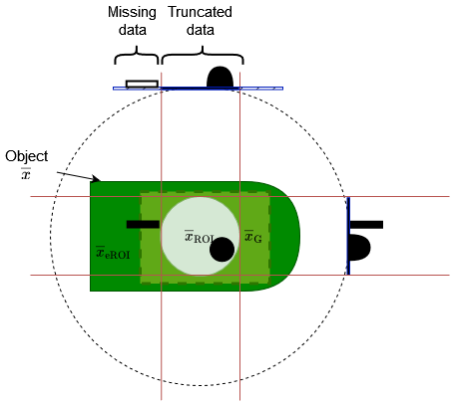


Fig. 1: ROI imaging in 2D. The exterior of the ROI contributes only to a subset of the projections

We consider a 1D detector array of B bins rotating around a patient. The detector is too short to image the entire object; its size defines a circular ROI we aim to reconstruct. Let S be the number of projection angles. The vector of truncated sub-sampled tomographic data is $\mathbf{y} \in \mathbb{R}^T$ with $T = B \times S$. A reconstruction of the patient attenuation map in the ROI can be obtained by considering a model of the form:

$$\mathbf{H}\overline{\mathbf{x}}_{\mathrm{G}} = \mathbf{y} + \mathbf{n}, \tag{1}$$

where $\mathbf{n} \in \mathbb{R}^M$ accounts for some acquisition noise, $\overline{\mathbf{x}}_{\mathrm{G}} \in \mathbb{R}^L$, and $\mathbf{H} \in \mathbb{R}^{L \times M}$ is the projector that models projection over an intermediary grid G that does not contain the support of the entire object. Operator $\mathbf{H}$ contains a subset of the columns of the projector on the entire space, or equivalently it corresponds to setting a subset of the columns of the complete projector, matching the pixels outside of the ROI, to zero.

When the grid corresponds to the ROI, (1) assumes that the

image values are 0 outside of the ROI. As this assumption is not necessarily true, (1) is not a good model for such a grid. A better idea is to extend the support of the reconstruction beyond the ROI so that no assumption is made about the values outside of the ROI. We now adopt this strategy.
We find an estimate of $\overline{\mathbf{x}}_{\mathrm{G}}$ by computing a minimizer of a penalized cost function consisting of the sum of a data fidelity term f involving $\mathbf{H}$ and $\mathbf{y}$, and a regularization term r, as

$$\underset{\mathbf{x}\in\mathbb{R}^L}{\operatorname{argmin}} \; f(\mathbf{x}) + r(\mathbf{x}). \tag{2}$$

The next section presents our choices for f and r, as well as a suitable algorithm to solve (2).

IV. Iterative reconstruction

A. Cost function

1) Cauchy data fidelity: When objects with high gradients (metallic wires or needles) do not belong to G, the error between the data and the reprojection of the estimate over G contains outliers at the projections of those high gradients. The effect of these outliers is amplified by the angular undersampling; it leads to streaks originating from these objects, i.e., from outside the grid. To avoid these outliers, we propose to decrease the influence of the largest errors between $\mathbf{y}$ and the reprojection of $\mathbf{H}\overline{\mathbf{x}}_{\mathrm{G}}$ using M-estimators. M-estimators have been used in ultrasound imaging [30] and in CT reconstruction [31], where the authors investigated their effect on ring artifacts that appear due to defective detector bins.
We focus on the Cauchy estimator ϕ, one redescending M-estimator [32], defined as

$$(\forall \zeta \in \mathbb{R}) \qquad \phi(\zeta) = \frac{\beta\kappa^2}{2} \ln\left(1 + \left(\frac{\zeta}{\kappa}\right)^2\right), \tag{3}$$

where $(\beta, \kappa) \in]0, +\infty[^2$.
We introduce the ramp filter [33] as $\mathbf{F} \in \mathcal{S}_L^+$, which acts as a metric matrix and a derivative operator, and we apply the Cauchy function to the ramp-filtered reprojection errors, i.e.,

$$(\forall \mathbf{x} \in \mathbb{R}^L) \quad f(\mathbf{x}) = g(\mathbf{F}(\mathbf{H}\mathbf{x} - \mathbf{y})) \tag{4}$$

with

$$(\forall \mathbf{z} = (z_t)_{1\leqslant t\leqslant T} \in \mathbb{R}^T) \qquad g(\mathbf{z}) = \sum_{t=1}^{T} \phi(z_t). \tag{5}$$

Note that the resulting cost function is non-convex. With such f, the reprojection error involving only the flat anatomical background is down-weighted with respect to the error due to outliers; the effect of the Cauchy term is more focused on outliers.

2) Semi-local total variation: Sub-sampling streaks due to a low angular density of the detector positions can be effectively handled by total variation (TV) regularization [3], [5], [6]. Semi-local variants (STV) [34] extend TV in a neighborhood of pixels indexed in $\Lambda_J = \{-J, \ldots, J\} \setminus \{0\}$:

$$\begin{aligned}(\forall \mathbf{x} \in \mathbb{R}^L)& \\ r_{\mathrm{STV}}(\mathbf{x}) &= \sum_{j=1}^{J}\sum_{\ell=1}^{L} \alpha_{j,\ell} \sqrt{(\mathbf{x} - \mathbf{V}_j\mathbf{x})_\ell^2 + (\mathbf{x} - \mathbf{V}_{-j}\mathbf{x})_\ell^2} \\ &= \sum_{j=1}^{J} r_j(\mathbf{L}_j\mathbf{x}).\end{aligned} \tag{6}$$

Hereinabove $\ell \in \{1, \ldots, L\}$ is the spatial index and $\mathbf{V}_j, \mathbf{V}_{-j} \in \mathbb{R}^{L\times L}$ are shift operators as shown in Figure 2 for $j \in \{1, \ldots, J\}$ and $J = 6$. Moreover, for every $j \in \{1, \cdots, J\}$, we define $\mathbf{L}_j = \begin{bmatrix}\mathbf{V}_j^\top & \mathbf{V}_{-j}^\top\end{bmatrix}^\top \in \mathbb{R}^{2L\times L}$ and, for every $\mathbf{z} = (\mathbf{z}_1, \mathbf{z}_2) \in \mathbb{R}^{2L}$, $h_j(\mathbf{z}) = \sum_{\ell=1}^{L} \alpha_{j,\ell}\sqrt{(\mathbf{z}_1)_\ell^2 + (\mathbf{z}_2)_\ell^2}$. Parameters $(\alpha_{j,\ell})_{1\leqslant j\leqslant J, 1\leqslant \ell\leqslant L}$ are nonnegative weights that can be chosen to vary spatially, so making STV adaptive to the spatial contents [35]. We recover the standard TV penalty for constant values of these parameters and $J = 1$.

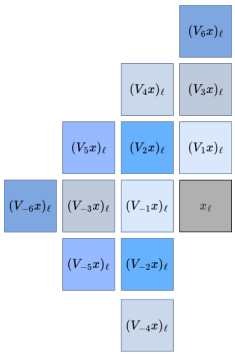


Fig. 2: Shift operators $(V_j)_{j\in\Lambda_6}$ applied to a given pixel position ℓ

We complement the above STV regularization function r_{STV} with a nonnegativity constraint on the pixel values, and a quadratic regularization $\frac{1}{2}\|\mathbf{x}\|_{\mathbf{M}}^2 = \mathbf{x}^\top\mathbf{M}\mathbf{x}$. Here, matrix $\mathbf{M} = \mathrm{diag}((m_\ell)_{\ell=1}^L) \in \mathcal{S}_L^+$ is such that, for every $\ell \in \{1, \ldots, L\}$, $m_\ell = 1$ if the ℓ-th entry x_ℓ of vector $\mathbf{x}$ belongs to the ROI, and $m_\ell = \xi > 1$ otherwise. In such a way, $\mathbf{M} \in \mathcal{S}_L^+$ acts as a mask allowing us to smooth the reconstruction outside of the ROI.
Altogether, our regularization function in (2) reads

$$(\forall \mathbf{x} \in \mathbb{R}^L) \quad r(\mathbf{x}) = \sum_{j=1}^{J} r_j(\mathbf{L}_j\mathbf{x}) + \frac{1}{2}\|\mathbf{x}\|_{\mathbf{M}}^2 + \iota_{[0,+\infty[^L}(\mathbf{x}). \tag{7}$$

B. Minimization of the cost function

1) Reweighting for non-convex criterion: Given our choices for r and f, the optimization problem (2) becomes

$$\underset{\mathbf{x}\in\mathbb{R}^L}{\operatorname{argmin}} \; g(\mathbf{F}(\mathbf{H}\mathbf{x}-\mathbf{y})) + \sum_{j=1}^{J} r_j(\mathbf{L}_j\mathbf{x}) + \frac{1}{2}\|\mathbf{x}\|_{\mathbf{M}}^2 + \iota_{[0,+\infty[^L}(\mathbf{x}). \tag{8}$$

Because of the non-convexity of the data fidelity term, we adopt an iterative reweighting strategy where Problem (8) is

replaced by a sequence of surrogate convex problems built following a majoration principle. As shown in Appendix A, the associated Majorization-Minimization strategy is given by Algorithm 1.

Algorithm 1 Iterative reweighting strategy for Problem (8)

$\mathbf{x}_0 \in \mathbb{R}^L$
for $k = 0$ to $K-1$ **do**
  Define majorant function $Q(\mathbf{x};\overline{\mathbf{x}})$
  $\mathbf{x}_{k+1} = \underset{\mathbf{x}\in\mathbb{R}^L}{\operatorname{argmin}}\ Q(\mathbf{x},\mathbf{x}_k)$
end for

According to (35), for every $\overline{\mathbf{x}} \in \mathbb{R}^L$, the surrogate function $Q(\cdot,\overline{\mathbf{x}})$ takes the form

$$Q(\mathbf{x},\overline{\mathbf{x}}) = \iota_{[0,+\infty[^L}(\mathbf{x}) + h(\mathbf{B}\mathbf{x};\mathbf{B}\overline{\mathbf{x}}) + \frac{1}{2}\|\mathbf{x}\|^2_{\mathbf{M}}, \quad (9)$$

where $\forall(\mathbf{x},\overline{\mathbf{x}}) \in (\mathbb{R}^L)^2$,

$$h(\mathbf{B}\mathbf{x};\mathbf{B}\overline{\mathbf{x}}) = h_0(\mathbf{B}_0\mathbf{x};\mathbf{B}_0\overline{\mathbf{x}}) + h_1(\mathbf{B}_1\mathbf{x})$$
$$h_1(\mathbf{B}_1\mathbf{x}) = \sum_{j=1}^{J} r_j(\mathbf{L}_j\mathbf{x}), \quad (10)$$

with

$$\mathbf{B} = \begin{bmatrix}\mathbf{B}_0^\top & \mathbf{B}_1^\top\end{bmatrix}^\top, \quad (11)$$
$$\mathbf{B}_0 = \mathbf{F}\mathbf{H}, \quad \mathbf{B}_1 = \begin{bmatrix}\mathbf{L}_1^\top & \cdots & \mathbf{L}_J^\top\end{bmatrix}^\top \quad (12)$$

and, for every $\overline{\mathbf{x}} \in \mathbb{R}^L$, $h_0(\cdot;\mathbf{B}_0\overline{\mathbf{x}}) = \tilde{g}(\cdot - \mathbf{F}\mathbf{y};\mathbf{B}_0\overline{\mathbf{x}} - \mathbf{F}\mathbf{y})$.

2) Dual block forward-backward algorithm: Since $\mathbf{M} \in \mathcal{S}_L^+$, function $Q(\cdot,\overline{\mathbf{x}})$ is strongly convex. Its minimization at each reweighted step is well-defined, with a unique solution that can be conveniently obtained using the dual forward-backward algorithm [22], [36]. An accelerated version of the dual forward-backward algorithm is its block coordinate version (DBFB) [37], which allows for splitting the terms of function (10) and accessing their proximity operators separately. We exploit the splitting of h into two blocks h_0 and h_1 to yield the form of the DBFB algorithm provided in (2). The estimate $\boldsymbol{x}_n$ generated by the algorithm at iteration n has been denoted in bold italic to distinguish it from the reweighting point $\overline{\mathbf{x}}$.

At each iteration of the DBFB algorithm, either the data fidelity term or the regularization term is processed, through the **data (D)** and **regularization (R)** steps, respectively. The implementation of each step is discussed in Appendix B.

V. Unfolded reconstruction

We now present our unfolding version for the reweighted DBFB, designated as U-RDBFB (Unfolded Reweighted DBFB).

A. Layering and relaxation of the DBFB algorithm

The unfolding principle recasts every algorithm iteration as one neural network layer to allow selected algorithm parameters to be learned in a supervised way. Each layer is thus made of costly and ill-conditioned linear operators ($\mathbf{B}_0$, $\mathbf{B}_1$ and their adjoints). Several works in iterative reconstruction [38]–[41] have shown the potential to replace adjoint operators with surrogates to accelerate convergence [42]–[44]. Therefore, in order to minimize the number of U-RDBFB layers, we propose relaxed steps **(D)** and **(R)** with surrogates, and thus mismatched adjoints.

Algorithm 2 DBFB algorithm to minimize (9) with $\overline{\mathbf{x}} \in \mathbb{R}^L$

$\mathbf{z}_0^0 \in \mathbb{R}^T, (\forall j \in \{1,\dots,J\})\ \mathbf{z}_0^j \in \mathbb{R}^{2L}, \mathbf{s}_0^1 = (\mathbf{z}_0^1,\dots,\mathbf{z}_0^J)$
$(\sigma,\tau_1,\dots,\tau_J) \in]0,+\infty[^{J+1}$
$\Sigma = \operatorname{diag}(\tau_1,\dots,\tau_J)$
$\mathbf{w}_0 = -\mathbf{M}^{-1}(\mathbf{B}_0^\top\mathbf{z}_0^0 + \mathbf{B}_1^\top\mathbf{s}_0^1)$ (13)
For $n = 0, 1\dots$
  $(\mathbf{z}_n^1,\dots,\mathbf{z}_n^J) \equiv \mathbf{s}_n^1$
  $\boldsymbol{x}_n = \operatorname{proj}_{[0,+\infty[^L}(\mathbf{w}_n)$
  Select $\varepsilon_n \in \{0,1\}$ and $\gamma_n \in]0,+\infty[$
  If $\varepsilon_n = 0$ **(D)**
    $\tilde{\mathbf{z}}_n^0 = \mathbf{z}_n^0 + \gamma_n\sigma^{-1}\mathbf{B}_0\boldsymbol{x}_n$
    $\mathbf{z}_{n+1}^0 = \tilde{\mathbf{z}}_n^0 - \gamma_n\sigma^{-1}\operatorname{prox}_{\gamma_n^{-1}\sigma h_0(\cdot;\mathbf{B}_0\overline{\mathbf{x}})}(\gamma_n^{-1}\sigma\tilde{\mathbf{z}}_n^0)$
    $\mathbf{w}_{n+1} = \mathbf{w}_n - \mathbf{M}^{-1}\mathbf{B}_0^\top(\mathbf{z}_{n+1}^0 - \mathbf{z}_n^0)$
    $\mathbf{s}_{n+1}^1 = \mathbf{s}_n^1$
  If $\varepsilon_n = 1$ **(R)**
    $\tilde{\mathbf{s}}_n^j = \mathbf{s}_n^j + \gamma_n\Sigma_j^{-1}\mathbf{B}_1\boldsymbol{x}_n$
    $\mathbf{s}_{n+1}^j = \tilde{\mathbf{s}}_n^j - \gamma_n\Sigma_j^{-1}\operatorname{prox}_{\gamma_n^{-1}\Sigma_j h_1}(\gamma_n^{-1}\Sigma_j\tilde{\mathbf{s}}_n^j)$
    $\mathbf{w}_{n+1} = \mathbf{w}_n - \mathbf{M}^{-1}\mathbf{B}_1^\top(\mathbf{s}_{n+1}^1 - \mathbf{s}_n^1)$
    $\mathbf{z}_{n+1}^0 = \mathbf{z}_n^0$

The transpose of $\mathbf{F}$ by an operator $\tilde{\mathbf{F}} \in \mathbb{R}^{L\times L}$. By setting $\nu_{n,0} = \gamma_n\sigma^{-1}$, $\mathbf{u}_n = \tilde{\mathbf{z}}_n^0 - \nu_{n,0}\mathbf{F}\mathbf{y}$, and by using the relation

$$\operatorname{prox}_{\nu_{n,0}h_0(\cdot;\mathbf{B}_0\overline{\mathbf{x}})} = \operatorname{prox}_{\nu_{n,0}\tilde{g}(\cdot;\mathbf{H}\overline{\mathbf{x}}-\mathbf{y})}(\cdot - \mathbf{F}\mathbf{y}) + \mathbf{F}\mathbf{y},$$

we modify step **(D)** to define layer $\mathcal{L}_{\mathrm{D}}$, mimicking the n-th iteration of DBFB for $\varepsilon_n = 0$ and a given $\overline{\mathbf{x}}$:

Data layer ($\mathcal{L}_{\mathrm{D}}$):

$$\begin{aligned}
&\boldsymbol{x}_n = \operatorname{proj}_{[0,+\infty[^L}(\mathbf{w}_n)\\
&\mathbf{u}_n = \mathbf{z}_n^0 + \nu_{n,0}\mathbf{F}(\mathbf{H}\boldsymbol{x}_n - \mathbf{y})\\
&\mathbf{z}_{n+1}^0 = \mathbf{u}_n - \nu_{n,0}^{-1}\operatorname{prox}_{\nu_{n,0}\tilde{g}(\cdot;\mathbf{H}\overline{\mathbf{x}}-\mathbf{y})}(\nu_{n,0}\mathbf{u}_n)\\
&\mathbf{w}_{n+1} = \mathbf{w}_n - \mathbf{M}^{-1}\mathbf{H}^\top\tilde{\mathbf{F}}(\mathbf{z}_{n+1}^0 - \mathbf{z}_n^0)\\
&\mathbf{z}_{n+1}^j = \mathbf{z}_n^j \quad (\forall j \in \{1,\dots,J\}).
\end{aligned} \quad (14)$$

Similarly, for step **(R)**, we unfold by replacing the TV-based operators $(\mathbf{L}_j^\top)_{j=1}^J$ by surrogate learnable operators $(\tilde{\mathbf{L}}_j)_{j=1}^J$. Using (38) and setting, for every $j \in \{1,\dots,K\}$, $\nu_{n,j} = \gamma_n\tau_j^{-1}$ yield the following regularization layer ($\mathcal{L}_{\mathrm{R}}$),

associated with the n-th DBFB iteration when $\varepsilon_n = 1$:

Regularization layer ($\mathcal{L}_{\mathrm{R}}$) :

$$\left\lfloor \begin{array}{l} \boldsymbol{x}_n = \mathrm{proj}_{[0,+\infty[^L}(\mathbf{w}_n) \\ \text{For } j \in \{1,\dots,J\} \\ \quad \left\lfloor \begin{array}{l} (\forall \ell \in \{1,\dots,L\}) \\ \quad (\mathbf{z}^j_{n+1})_\ell = \dfrac{\left(\mathbf{z}^j_n + \nu_{n,j}\mathbf{L}_j \boldsymbol{x}_n\right)_\ell}{\max\left\{1, \| \left(\mathbf{z}^j_n + \nu_{n,j}\mathbf{L}_j \boldsymbol{x}_n\right)_\ell \|_2/\alpha_{j,\ell}\right\}} \end{array} \right. \\ \mathbf{w}_{n+1} = \mathbf{w}_n - \mathbf{M}^{-1}\sum_{j=1}^J \tilde{\mathbf{L}}_j(\mathbf{z}^j_{n+1} - \mathbf{z}^j_n) \\ \mathbf{z}^0_{n+1} = \mathbf{z}^0_n. \end{array} \right. \tag{15}$$

We refer to the above regularization layer as $\mathcal{L}^*_R$ when using it without learning surrogates adjoints, i.e. when, for every $j \in \{1,\dots,J\}$, $\tilde{\mathbf{L}}_j = \mathbf{L}_j^\top$. Note that $\mathcal{L}^*_R$ does not involve $\overline{\mathbf{x}}$.

B. Total architecture

The K groups of layers of U-RDBFB architecture correspond to K reweighted steps, each consisting of N_k unfolded layers of DBFB. This yields an architecture with $\sum_{k=0}^{K-1} N_k$ layers in total. We investigate three variants of this architecture:

- **Architecture 1** ($\mathcal{A}_1$): the step sizes and the regularization parameters are learned and $\tilde{\mathbf{F}} = \mathbf{F}^\top$.
  The k-th group of layers with $k \in \{0,\dots,K-1\}$ implements the following update:

$$(\mathbf{x}_{k+1}, \mathbf{z}_{0,k+1}) = \mathcal{L}^k_1(\mathbf{x}_k, \Theta_{1,k}, \mathbf{z}_{0,k}). \tag{16}$$

  Hereinabove, $\mathcal{L}^k_1$ is a composition of N_k layers $\mathcal{L}_{\mathrm{D}}$ or $\mathcal{L}^*_{\mathrm{R}}$. $\mathbf{x}_k$ (corresponding to $\boldsymbol{x}_0$ in DBFB) is the initial value of the primal variable and $\mathbf{x}_{k+1}$ the generated one (corresponding to $\boldsymbol{x}_{N_k-1}$ in DBFB). $\mathbf{z}_{0,k} \in \mathbb{R}^T \times (\mathbb{R}^{2L})^J$ is the initial value of the dual variables (corresponding to $(\mathbf{z}^j_0)_{0\leqslant j\leqslant J}$ in DBFB) and $\mathbf{z}_{0,k+1}$ is the final value of these variables (corresponding to $(\mathbf{z}^j_{N_{k-1}})_{0\leqslant j\leqslant J}$ in DBFB). The variable $\mathbf{w}_0$ required in the unfolded version of DBFB is set as in (13). $\Theta_{1,k}$ is the vector of trainable parameters. As we allow all these parameters to be layer-dependent, we thus index them by k and n. If the n-th layer of $\mathcal{L}^k_1$ is a $\mathcal{L}_{\mathrm{D}}$ layer, its parameters are those of the Cauchy function $(\beta_{k,n}, \kappa_{k,n})$, the one of the quadratic regularization $\xi_{k,n}$, and a single step size $\nu_{k,n,0}$. If this is a $\mathcal{L}^*_{\mathrm{R}}$ layer, regularization parameters $(\alpha_{k,n,j,\ell})_{1\leqslant j\leqslant J, 1\leqslant \ell\leqslant L}$, $\kappa_{k,n}$, $\xi_{k,n}$, and step sizes $(\nu_{k,n,j})_{j=1}^J$ are learned.
- **Architecture 2** ($\mathcal{A}_2$): similar to $\mathcal{A}_1$ with $\tilde{\mathbf{F}} = \mathrm{Id}$.
- **Architecture 3** ($\mathcal{A}_3$): in addition to algorithm parameters (step sizes, Cauchy function and STV weights), the surrogates $(\tilde{\mathbf{L}}_j)_{j=1}^J$ to the adjoints of operators $(\mathbf{L}_j^\top)_{j=1}^J$ are also learned so that the k-th group of layers of Architecture 3 implements the update rule:

$$(\mathbf{x}_{k+1}, \mathbf{z}_{0,k+1}) = \mathcal{L}^k_2(\mathbf{x}_k, \Theta_{2,k}, \mathbf{z}_{0,k}), \tag{17}$$

  with $\tilde{\mathbf{F}} = \mathrm{Id}$. $\mathcal{L}^k_2$ is then a composition of N_k layers $\mathcal{L}_{\mathrm{D}}$ or $\mathcal{L}_{\mathrm{R}}$. Vector $\Theta_{2,k}$ contains the same parameters as $\Theta_{1,k}$ and the elements of matrices $(\tilde{\mathbf{L}}_j)_{j=1}^J$ for each $\mathcal{L}_{\mathrm{R}}$ layer.

Figure 3 illustrates one instance of Architecture 3, where $\mathcal{A}_3 = \mathcal{L}^1_2 \circ \mathcal{L}^0_2$ with $\mathcal{L}^0_2 = (\mathcal{L}_{\mathrm{R}})^2 \circ \mathcal{L}_{\mathrm{D}}$ and $\mathcal{L}^1_2 = \mathcal{L}_{\mathrm{D}}$ (i.e., $K = 2$, $N_0 = 3$ and $N_1 = 1$).

C. Parameter learning

To infer the parameters gathered in vector $\Theta_{i,k}$, in the k-th group of layers $\mathcal{L}^k_i$ of architecture $i \in \{1,2\}$, we introduce hidden layers. In the following, $(\mathcal{L}^{(\theta)}_{k,n})_{n=0}^{N_k-1}$ designates the hidden layers used to estimate a given parameter θ. We now discuss our choices for $(\mathcal{L}^{(\theta)}_{k,n})_{n=0}^{N_k-1}$.

- step size for $\mathcal{L}_{\mathrm{D}}$: $\nu_{k,n,0} = \mathcal{L}^{(\nu)}_{k,n} = \mathrm{softplus}(a_{k,n})$ where $a_{k,n}$ is a learnable real-valued parameter.
- Parameters of the Cauchy function for $\mathcal{L}_{\mathrm{D}}$:
  - $\kappa_{k,n} = \mathcal{L}^{(\kappa)}_{k,n} = W_\kappa\,\mathrm{softplus}(c_{k,n})$ where $c_{k,n}$ is inferred from a fully connected layer, whose weights are shared across the U-RDBFB network, applied on a histogram of the absolute value of the filtered reprojection error i.e. $\mathbf{F}(\mathbf{H}^\top\mathbf{x}_n - \mathbf{y})$. We implemented the learnable histogram layer proposed in [45], which is piecewise differentiable. More precisely, we built a cumulated histogram using 100 bins from 0 to the maximum value of the filtered reprojection error.
  - $\beta_{k,n} = \mathcal{L}^{(\beta)}_{k,n} = W_\beta\,\mathrm{softplus}(d_{k,n})$.
- Diagonal elements of $\mathbf{M}$ involved in (7) corresponding to the locations of pixels outside of the ROI for both $\mathcal{L}_{\mathrm{D}}$ and $\mathcal{L}_{\mathrm{R}}$/$\mathcal{L}^*_{\mathrm{R}}$: $\xi_{k,n} = \mathcal{L}^{(\xi)}_{k,n} = \mathrm{softplus}(e_{k,n})$ where $e_{k,n}$ is learned.
- step size for $\mathcal{L}_{\mathrm{R}}$/$\mathcal{L}^*_{\mathrm{R}}$: For every $j \in \{1,\dots,J\}$, $\nu_{k,n,j} = \mathcal{L}^{(\nu_j)}_{k,n} = W_\nu\,\mathrm{softplus}(b_{k,n,j})$.
- Parameters of the STV regularization for $\mathcal{L}_{\mathrm{R}}$/$\mathcal{L}^*_{\mathrm{R}}$: For every $j \in \{1,\dots,J\}$,

$$\begin{aligned} \alpha_{k,n,j} &= (\alpha_{k,n,j,l})_{\ell=1}^L = \mathcal{L}^{(\alpha_j)}_{k,n} \\ &= W_\alpha\,\mathrm{softplus}(A_{k,n} \circ \mathrm{ReLU} \circ B_{k,n}(\mathbf{L}_j\mathbf{x}_k)), \end{aligned}$$

  where $A_{k,n}$ and $B_{k,n}$ are two convolutional operators with size 3×3 kernels and 8 channels.

Finally, when using layer $\mathcal{L}_R$, we also learn operators $(\tilde{\mathbf{L}}^{k,n}_j)_{j=1}^J$ which have the same support as $(\mathbf{L}^{k,n}_j)_{j=1}^J$.

In $\mathcal{L}_D$, initial values of $a_{k,n}$, $d_{k,n}$ are set to 1. In $\mathcal{L}_R$ and $\mathcal{L}^*_R$, initial values for $b_{k,n,j}$, $e_{k,n}$ are 1. Normalization scalars W_κ and W_ν are set to 10^{-5} and 40 for $\mathcal{L}^*_R$ and to 10^{-5} and 10 for $\mathcal{L}_R$, respectively. When $\tilde{\mathbf{F}} = \mathrm{Id}$, W_β and W_α are set to 10 and 0.05 and, when $\tilde{\mathbf{F}} = \mathbf{F}^\top$, W_β and W_α are set to 2×10^5 and 5.

A schematic view of the data layer can be found in Figure 4, whereas Figure 5 shows an example of the regularization layer according to these settings.

VI. Experiments

A. Datasets

Two datasets of CT images were used in our experiments.

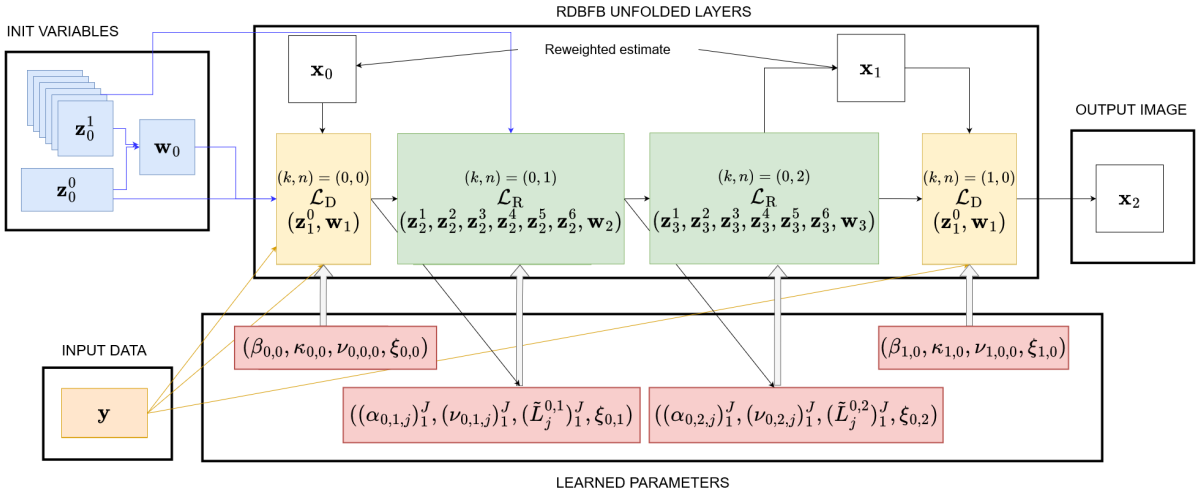


Fig. 3: Example of Architecture 3 of U-RDBFB in the case where $\mathcal{A}_3 = \mathcal{L}_2^1 \circ \mathcal{L}_2^0$ where $\mathcal{L}_2^0 = (\mathcal{L}_R)^2 \circ \mathcal{L}_D$ and $\mathcal{L}_2^1 = \mathcal{L}_D$ (i.e., $K = 2$, $N_0 = 3$, $N_1 = 1$). Red blocks represent the hidden structures to infer all the parameters $\theta \in \Theta_2$. When $k = 1$, the dual variables of DBFB are initialized with the values of the dual variables at the end of the previous N iterations of DBFB for $k = 0$.

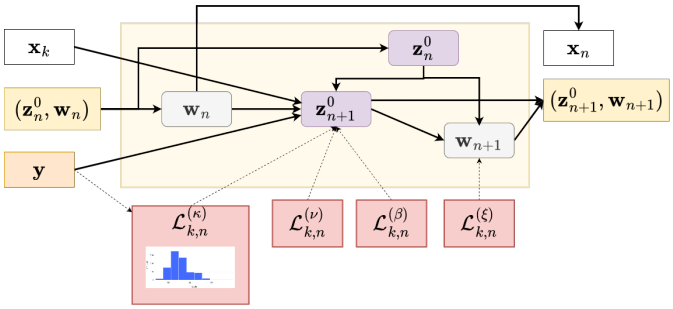


Fig. 4: Schematic view of layer $\mathcal{L}_D$, mimicking (14), for $\overline{\mathbf{x}} = \mathbf{x}_k$, i.e. k-th iterate of the reweighted algorithm. The layer takes as inputs $\mathbf{w}_n$, $\mathbf{z}_n^0$ from the previous layers. The projections $\mathbf{y}$ are also used as input.

1) Abdomen dataset: Our first dataset consists of 2D images obtained from 70 CT volumes of size $512 \times 512 \times 512$ from the lower lungs to the lower abdomen of 70 patients, which were extracted from the public dataset *CT Lymph Nodes* from https://www.cancerimagingarchive.net/. These volumes correspond to fully sampled CT reconstructions. They were made isotropic by interpolating the axial slices. A total of 50 out of 512 slices were kept per volume. We randomly added intense metallic wires between 3000 and 5000 Hounsfield units (HU) of varying sizes on the axial slices. We shifted the HU values of the images by 1000 so that air is 0 HU and water is 1000 HU, using $\boldsymbol{a}_{\text{tissue}} \mapsto (\boldsymbol{a}_{\text{tissue}} - \mu_{\text{water}}) \times (1000/\mu_{\text{water}})$, where μ_{water} is the value of the attenuation coefficient of water equal to 0.017 mm^{-1} and $\boldsymbol{a}_{\text{tissue}} \in \mathbb{R}^Q$ is the initial vector of attenuation values. Finally the 512×512 slices $\overline{\mathbf{x}}_{\text{P}}$ were normalized between $[0, 1]$ (a value of 1 corresponding to an object of HU intensity equal to 5000).
Projections were simulated for each slice of each volume in a 2D parallel geometry with a short detector of 300 bins (bin size equal to pixel size, i.e. 1 mm) and an angular density of 110 projections over 180°. Noisy projections $\mathbf{y} = (y_t)_{t=1}^T$ are computed as

$$(\forall t \in \{1, \dots, T\}) \quad y_t = \mu \log \left(\frac{I_0}{\mathcal{P}(I_0 \exp\left(-\mu(\mathbf{H}\overline{\mathbf{x}}_{\text{P}})_t\right))} \right), \tag{18}$$

where we set $\mu = \mu_{\text{water}}/1000$, $I_0 = 10^4$, and, for some $\delta > 0$, $\mathcal{P}(\delta)$ denotes a realization of a Poisson law with mean δ. In this context, the ROI was a centered disk of diameter 300. The resulting pairs of axial slice/projections $(\overline{\mathbf{x}}_{\text{P}}, \mathbf{y})$ were split into a training of 2500 pairs from a pool of 50 patients, a testing set of 500 pairs from 10 other patients, and a validation of 500 pairs from the 10 remaining patients.

2) Brain dataset: We used a second dataset containing 2D images extracted from 10 CT high-dose brain reconstructions. These volumes are from the public repository *2016 Low Dose CT Grand Challenge* from https://www.cancerimagingarchive.net/. We performed the same post-processing as above (with a pixel size of 0.5 mm) for generating a training set of 2500 pairs of axial slices/projections $(\overline{\mathbf{x}}_{\text{P}}, \mathbf{y})$ from 30 patients, validation and testing sets of 500 pairs from 10 different patients each.

B. Training strategy

Let $i = \{1, \dots, I\}$ be the index covering all I instances of the training set. The reconstruction grid (G) is a disk of diameter 400. Let $\mathbf{x}_{\text{G,i}}^* \in \mathbb{R}^L$ be the output of U-RDBFB for a given projection input $\mathbf{y}_i$. Our networks are designed to minimize $\sum_{i=1}^I \ell(\mathbf{C}_{\text{G}}\mathbf{x}_{\text{G},i}^*, \mathbf{C}_{\text{P}}\overline{\mathbf{x}}_{\text{P},i})$, where $\mathbf{C}_{\text{G}}$ is a cropping operator which extracts the ROI from the grid G, $\mathbf{C}_{\text{P}}$ is a cropping operator which extracts the ROI from the entire 512×512 grid, and $\ell(\cdot)$ is the loss retained for training the network.
We use the following similar compositions of layers for defining $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$:

$$\mathcal{A}_1 = \mathcal{A}_2 = \mathcal{L}_1^5 \circ \mathcal{L}_1^4 \circ \mathcal{L}_1^3 \circ \mathcal{L}_1^2 \circ \mathcal{L}_1^1 \circ \mathcal{L}_1^0, \tag{19}$$

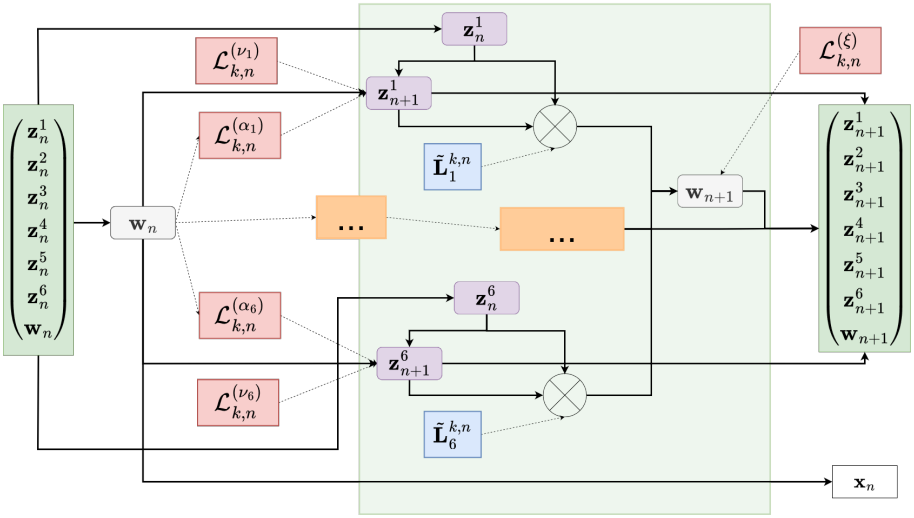


Fig. 5: Schematic view of layer $\mathcal{L}_R$ mimicking (15). The layer takes as inputs $\mathbf{w}_n$, $(\mathbf{z}_n^j)_1^J$ from the previous layer. The update of parameters for $j \in \{2, \dots, 5\}$ is hidden in the orange block for the sake of readability. When $\mathcal{L}_R$ reduces to $\mathcal{L}_R^*$, the blue blocks disappear. Only parameters α_j depend on the input.

where

$$\begin{aligned}\mathcal{L}_1^0 &= (\mathcal{L}_R^*)^5 \circ \mathcal{L}_D \\ \mathcal{L}_1^1 = \mathcal{L}_1^2 &= (\mathcal{L}_R^*)^4 \circ \mathcal{L}_D \\ \mathcal{L}_1^3 = \mathcal{L}_1^4 = \mathcal{L}_1^5 &= (\mathcal{L}_R^*)^3 \circ \mathcal{L}_D\end{aligned} \tag{20}$$

and

$$\mathcal{A}_3 = \mathcal{L}_2^5 \circ \mathcal{L}_2^4 \circ \mathcal{L}_2^3 \circ \mathcal{L}_2^2 \circ \mathcal{L}_2^1 \circ \mathcal{L}_2^0, \tag{21}$$

where

$$\begin{aligned}\mathcal{L}_2^0 &= (\mathcal{L}_R)^5 \circ \mathcal{L}_D \\ \mathcal{L}_2^1 = \mathcal{L}_2^2 &= (\mathcal{L}_R)^4 \circ \mathcal{L}_D \\ \mathcal{L}_2^3 = \mathcal{L}_2^4 = \mathcal{L}_2^5 &= (\mathcal{L}_R)^3 \circ \mathcal{L}_D.\end{aligned} \tag{22}$$

These architectures correspond to setting $K = 6$ and using a decreasing value of N_k for each $k \in \{0, \cdots, K-1\}$ ($N_0 = 6$, $N_1 = N_2 = 5$, $N_3 = N_4 = N_5 = 4$), resulting in a total of 28 layers per network.

For the first reweighted iteration ($k = 0$), the dual estimates $(\mathbf{z}_0^j)_{j=1}^J$ are initialized to zero while the dual estimate $\mathbf{z}_0^0$ is set to the negative filtered projections, $-\mathbf{F}\mathbf{y}$, leading to $\mathbf{x}_0 = \mathbf{H}^\top \mathbf{F}\mathbf{y}$. We implemented all networks in Pytorch, using a Tesla V100 32 Gb GPU. We employed the toolbox TorchRadon [46] to include Pytorch-compatible parallel-beam tomographic operators in all architectures. The training procedure of $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$ takes about one day and a half. Standard auto-differentiation tools can compute all necessary derivatives for backpropagation.

1) From layer-wise to greedy training: Our proposed initialization is a rough estimation of the parameters used for K reweighted iterations and N_k inner iterations of DBFB. We circumvent the initialization issue by using a layer-wise sequential training strategy, as sometimes advocated for when initializing the weights of recurrent neural networks [47], [48]. The learning in each layer (k, n) $(n \geqslant 1)$ starts by considering only its previous layers from $(0, 0)$ to $(k, n-1)$. As more and more layers are trained simultaneously, all 28 layers are eventually trained from end to end. We used ten epochs for training each layer $\mathcal{L}_\mathrm{R}$ and twenty epochs for training each layer $\mathcal{L}_\mathrm{D}$; the only exception was the last layer, for which we used 50 epochs. The learning rate is decreased with a step decay by a factor of 0.995 from 10^{-2} every 4 epochs. The batch size used for each epoch varied from 20 to 8 as the number of trained layers increased.

2) Choice for ℓ: For all instances of the training set, $\ell(\mathbf{C}_\mathrm{G}\mathbf{x}_{\mathrm{G,i}}^*, \mathbf{C}_\mathrm{P}\overline{\mathbf{x}}_{\mathrm{P,i}}) = \frac{1}{I}\|\mathbf{C}_\mathrm{G}\mathbf{x}_{\mathrm{G,i}}^* - \mathbf{C}_\mathrm{P}\overline{\mathbf{x}}_{\mathrm{P,i}})\|^2$, corresponding to the MSE loss for $\mathcal{A}_1$ and $\mathcal{A}_2$. For $\mathcal{A}_3$, we used a regularized MSE loss, which contains an additional term $-\lambda \sum_{k=0}^{K-1} \sum_{n=0}^{N_k-1} \sum_{j=1}^{J} \mathrm{trace}\left(\tilde{\mathbf{L}}_j^{k,n} \mathbf{L}_j^{k,n}\right)$ weighted by parameter $\lambda > 0$. To set λ, $\mathcal{A}_2$ was overfit to an example of our dataset. A grid search on λ was performed and we selected the value which minimized the MSE on this example.

C. Comparative strategy

The quantitative metric used to assess the reconstruction quality of $\mathbf{C}_\mathrm{G}\mathbf{x}_{\mathrm{G,i}}^*$ is the PSNR. We also evaluate the reconstruction performance using the structural similarity index (SSIM), the PieApp value [49], and the Mean Absolute Error (MAE) of the difference between $\mathbf{C}_\mathrm{G}\mathbf{x}_{\mathrm{G,i}}^*$ and $\mathbf{C}_\mathrm{P}\overline{\mathbf{x}}_{\mathrm{P,i}}$.

We compare our approach with five reconstruction methods, that we describe hereinafter.

1) Hierarchical reconstruction: The first competing method is the iterative hierarchical reconstruction algorithm of [6]. Inspired by homotopy strategies for solving ℓ_1-regularized least-squares problems [50], [51], this method amounts to solving, for $k \in \{1, \dots, K\}$, the following optimization problem parameterized by $\lambda_k \in [0, +\infty[$

$$\underset{\mathbf{x} \in \mathbb{R}^L}{\mathrm{argmin}} \ \frac{1}{2}\|\mathbf{H}\mathbf{x} - \mathbf{y}\|_\mathbf{F}^2 + \lambda_k r_1(\mathbf{L}_1\mathbf{x}), \tag{23}$$

where $r_1(\mathbf{L}_1 \cdot)$ is a special case of the DTV, reducing to the isotropic TV.

The authors perform $N = 1$ iteration of the forward-backward algorithm for each problem (23) with $K = 20$. The initial

value for the image with regularizing constant λ_k is chosen as the previous iterate obtained with λ_{k-1}, and the values $(\lambda_k)_{k=1}^K$ are linearly decreased from 100 to 0. This strategy reconstructs intense objects first and then anatomical details, hence its name "hierarchical reconstruction". Note that it does not address statistical noise but subsampling only. For our comparison, we use 100 inner iterations of the dual forward-backward algorithm to compute the proximity operator of TV regularization (with warm restart), within the forward-backward update.

2) Post-processing U-net: The second competing method is the CNN proposed in [16], [17]. This method is a post-processing of FBP, which consists of computing $\mathbf{H}_{\mathrm{ROI}}^\top \mathbf{F}\mathbf{y}$. It relies on a trained residual U-net, with a depth of 4 levels, filters of size 32, and a batch size of 15. Batch normalization is useful to improve the stability of training.

3) Preconditioned Neumann network (PNN): Our third competing method is a preconditioned Neumann network (PNN) initially introduced in [52] for MRI reconstruction. It builds on a method for solving problem (2) with $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{H}\mathbf{x} - \mathbf{y}\|^2$. For a differentiable function r, the resulting minimizer reads

$$(\mathbf{H}^\top\mathbf{H} + \nabla r)\mathbf{x} = \mathbf{H}^\top\mathbf{y}, \tag{24}$$

which can be rewritten as

$$(\mathbf{H}^\top\mathbf{H} + \lambda\,\mathrm{Id}\,)\mathbf{x} + (\nabla r - \lambda\,\mathrm{Id}\,)\mathbf{x} = \mathbf{H}^\top\mathbf{y}. \tag{25}$$

Setting $\mathbf{T}_\lambda = (\mathbf{H}^\top\mathbf{H} + \lambda\,\mathrm{Id}\,)^{-1}$ yields

$$(\mathrm{Id}\, - \lambda\mathbf{T}_\lambda + \mathbf{T}_\lambda\nabla r)\mathbf{x} = \mathbf{T}_\lambda\mathbf{H}^\top\mathbf{y}. \tag{26}$$

Using the Neumann identity $\mathbf{B}^{-1} = \sum_{n=0}^{\infty}(\mathrm{Id}\, - \mathbf{B})^n$, the authors derive the architecture of PNN with $N \in \mathbb{N}^*$ layers (see Figure 6)

$$(\lambda\mathbf{T}_\lambda - \mathbf{T}_\lambda\nabla r)^N \circ \mathbf{T}_\lambda(\mathbf{H}^\top\mathbf{y}). \tag{27}$$

All instances of $\mathbf{T}_\lambda$ are applied approximately using an unrolling of 10 iterations of the conjugate gradient algorithm. The operator $\mathbf{T}_\lambda\nabla r$ is replaced by a U-net, denoted by Ψ which has the same architecture as the aforementioned U-net without the residual connection. The weights of the U-net are shared for all layers. Following [52], no batch normalization is used.

We choose $N = 3$, the batch size is 6, the initial value for λ is 0.01, the inner U-net has a depth of 4, and the learning rate is set to 10^{-4}.

4) ISTA-net: Our fourth competing method is ISTA-net, which is derived from the work of [26]. ISTA-net is designed to solve Problem (2) for $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{H}\mathbf{x} - \mathbf{y}\|^2$, and $r(\mathbf{x}) = \lambda\|\mathbf{W}\mathbf{x}\|_1$ ($\lambda > 0$), where operator $\mathbf{W}$ is not known a priori but learned. $\mathbf{W}$ is an orthogonal linear operator in the initial ISTA algorithm, whose iteration reads

$$\mathbf{x}_{n+1} = \mathbf{W}^\top \mathrm{soft}\left(\mathbf{W}(\mathbf{x}_n - \tau\mathbf{H}^\top(\mathbf{H}\mathbf{x}_n - \mathbf{y}), \lambda\tau\right), \tag{28}$$

where soft is the soft-thresholding operation and $\tau > 0$ is the gradient step size. In ISTA-net, the authors replace $\mathbf{W}$ and $\mathbf{W}^\top$ by two decoupled nonlinear operators namely $\mathbf{A}_n \circ \mathrm{ReLU} \circ \mathbf{B}_n$ and $\mathbf{C}_n \circ \mathrm{ReLU} \circ \mathbf{D}_n$ (see Figure 7). The

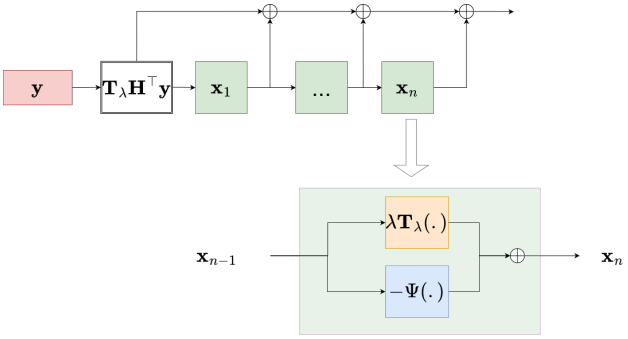


Fig. 6: Architecture of PNN [52]: The network maps a linear function of the measurements $\mathbf{T}_\lambda\mathbf{H}^\top\mathbf{y}$ to a reconstruction $\mathbf{x}_n$ by successive applications of an operator of the form $\lambda\mathbf{T}_\lambda - \Psi$, while summing the intermediate outputs of each block. All instances of $\mathbf{T}_\lambda$ are replaced by an unrolling of 10 iterations of the conjugate gradient algorithm. Ψ is a trained network and the scale parameter λ is also trained.

property of orthogonality of $\mathbf{W}$ is not imposed but favored during training by adding a term, weighted by $\chi \in]0, +\infty[$, penalizing the difference between $(\mathbf{C}_n \circ \mathrm{ReLU} \circ \mathbf{D}_n) \circ (\mathbf{A}_n \circ \mathrm{ReLU} \circ \mathbf{B}_n)\mathbf{x}_n$ and $\mathbf{x}_n$ in the loss function. Each $\mathbf{A}_n$, $\mathbf{B}_n$, $\mathbf{C}_n$ and $\mathbf{D}_n$ is a 2D convolutional operator. $\mathbf{B}_n$ and $\mathbf{C}_n$ are associated with a kernel of size 3×3 and 16 input and output channels; $\mathbf{A}_n$ has 1 input channel and 16 output channels and vice-versa for $\mathbf{D}_n$. As suggested by the authors, we learn these convolutional operators as well as λ and τ, which are allowed to vary at each iteration.

Experiments are carried out with 10 layers, $\chi = 0.1$, a batch size of 14 and the learning rate is initialized to 10^{-3} (step-wise decreased by a factor 0.9 every 20 epochs). The gradient norm is clipped to 1, and $\mathbf{x}_0$ is set at zero.

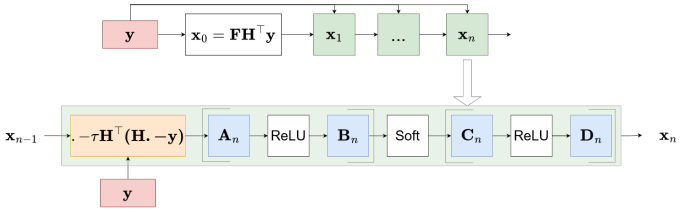


Fig. 7: Architecture of ISTA-net [26]: Each layer is composed of a gradient step followed by the application of a nonlinear operator which is the combination of two learnable linear convolutional operators ($\mathbf{A}_n$, $\mathbf{B}_n$) separated by a ReLU, a soft-thresholding operation and then two other learnable linear convolutional operators ($\mathbf{C}_n$, $\mathbf{D}_n$) separated by a ReLU. The property $(\mathbf{C}_n \circ \mathrm{ReLU} \circ \mathbf{D}_n) \circ (\mathbf{A}_n \circ \mathrm{ReLU} \circ \mathbf{B}_n) = \mathrm{Id}$ is favored during training.

5) PD-net: The last competing method is the learned Primal-Dual (PD-net) introduced in [53] by unrolling the Primal-Dual Hybrid Gradient (PDHG) optimization algorithm [54]. The authors consider problem (2) with a more generic data fidelity term $f(\mathbf{x}) = G(\mathbf{H}\mathbf{x}; \mathbf{y})$. They replace both the proximity operators of G and r in PDHG by residual CNN so that one layer $n \in \mathbb{N}$ of their network reads

$$\mathbf{z}_{n+1} = \mathrm{CNN}(\mathbf{z}_n + \sigma\mathbf{H}\tilde{\mathbf{x}}_n; \mathbf{y}) \tag{29}$$

$$\mathbf{x}_{n+1} = \mathrm{CNN}(\mathbf{x}_n - \tau\mathbf{H}^\top\mathbf{z}_{n+1}) \tag{30}$$

$$\tilde{\mathbf{x}}_{n+1} = \mathbf{x}_{n+1} + \gamma(\mathbf{x}_{n+1} - \mathbf{x}_n). \tag{31}$$

The CNNs act both in the image and projection domains. Furthermore, buffers of previous iterates of size $N_p \in \mathbb{N}$

	$\mathcal{A}_1$	$\mathcal{A}_2$	U-net	PNN	PD-net	ISTA-net
$\lvert\Theta\rvert$	21 242	31 311	483 153	447 214	117 393	48 980

TABLE I: Number of learnable parameters (Θ)

in the primal domain (image) and of size $N_d \in \mathbb{N}$ in the dual domain (projection) are kept to enable the network to learn an acceleration. For our implementation, we use a ReLU non-linearity instead of a PReLU. As advocated by the authors, the parameters used here are $N = 5$ layers, $N_d = N_p = 3$, and 32 filters in the convolutional layers. This network is illustrated in Figure 8. The batch size is 20, and the learning rate is initialized to 10^{-3} (step-wise decreased by a factor 0.9 every 20 epochs). The gradient norm is clipped to 1.

As advocated in their respective papers, the competing networks were also trained with the MSE loss for 100 epochs in a standard end-to-end manner. Note that codes are publicly available for these networks. We re-implemented them in Pytorch and re-tuned some of the hyper-parameters starting from the values proposed in the papers so that the number of parameters stays in the same order of magnitude for all networks (see table I).

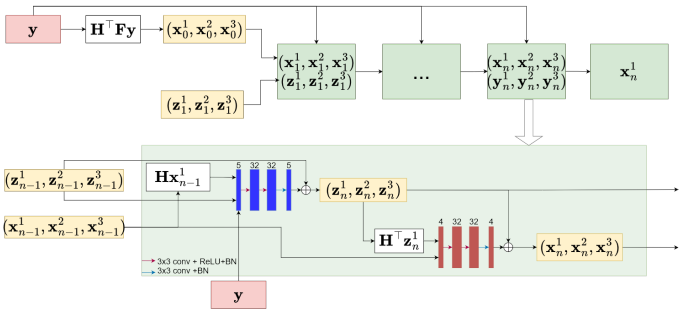


Fig. 8: Architecture of PD-net [53]: The red and blue boxes represent the primal and dual networks, respectively. Buffers of 3 primal $(\mathbf{x}_n^1, \mathbf{x}_n^2, \mathbf{x}_n^3)$ and dual $(\mathbf{z}_n^1, \mathbf{z}_n^2, \mathbf{z}_n^3)$ estimates are used at each iteration. The initial primal estimates are set to the FBP reconstruction given by $\mathbf{H}^\top\mathbf{F}\mathbf{y}$ and the initial dual estimates are set to zero.

VII. Results

A. Comparison with non-deep learning methods and use of an M-estimator

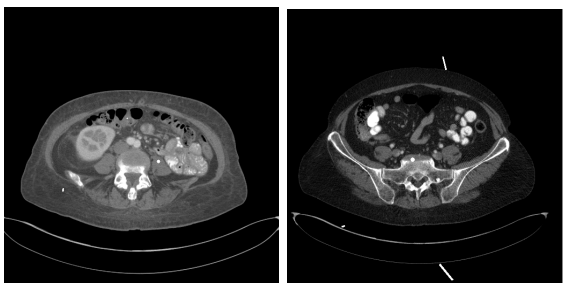


Fig. 9: Full ground truth $\overline{\mathbf{x}}_{\mathrm{P}}$: Example A (left) and Example B (right).

First, we compare architecture $\mathcal{A}_2$ with the hierarchical method and FBP. In $\mathcal{A}_2$, the mismatched adjoint is not learned; it is such that the ramp filter is applied only once between $\mathbf{H}$ and $\mathbf{H}^\top$, as is the case for the hierarchical method. However, unlike the latter, $\mathcal{A}_2$ takes into account the location of the ROI through its loss. For the comparison, we thus designed an overfitting network by training $\mathcal{A}_2$ on one instance of Dataset 1 at a time for two instances (Example A and Example B, see figure 9). These examples illustrate two phenomena. In Example A, the exterior of the ROI, which contributes to the truncated projections, is made of anatomical tissues only. In Example B, the projections contain the contribution of intense objects outside the reconstruction grid.

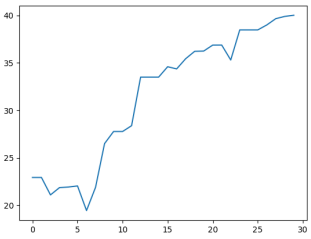
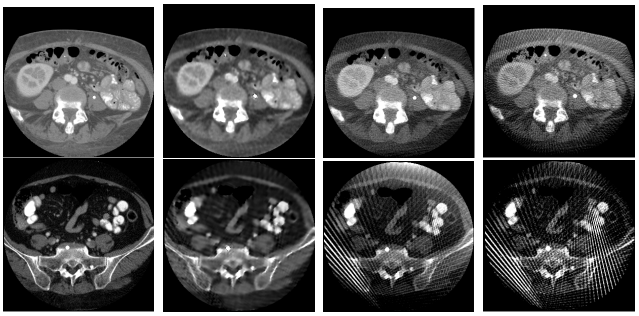


Fig. 10: Evolution of the PSNR across layers at the end of the end-to-end training phase of $\mathcal{A}_2$ trained on Example B.



(a) $\overline{\mathbf{x}}_{\mathrm{ROI}}$ (b) $\mathcal{A}_2$ (c) TV (d) FBP

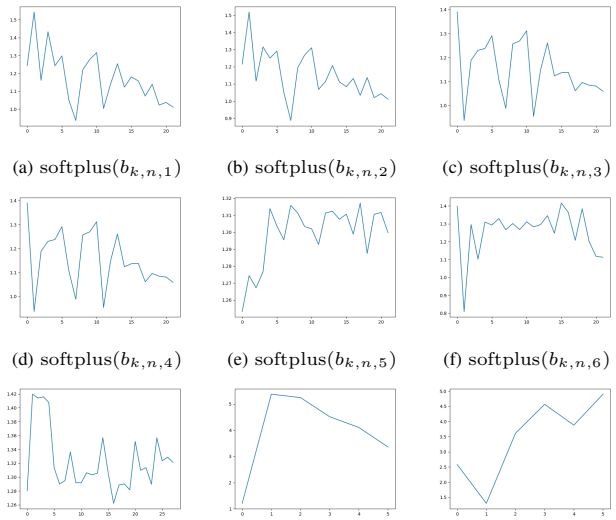
Fig. 11: Reconstructed ROIs of Example A (top) and Example B (bottom) obtained by two overfitting architectures of type $\mathcal{A}_2$, the hierarchical TV method and FBP.

Figure 10 shows the evolution of the PSNR for Example B along the 28 layers of the same architecture. These plots show that, for our choice of K and N, the network can learn an optimized set of parameters such that the PSNR gradually increases across each layer, though less rapidly in the last layers. Given that larger values for K and N imply a higher memory and training time, increasing the total number of layers here appeared unnecessary.

Figure 12 shows the learned scalar parameters in the data and regularization layers for Example B. Interestingly, the regularization step sizes ($b_{k,n,j}$) tend to decrease for the neighbors of x_ℓ (see Figure 2), except for its two furthest neighbors. The weight of the data term ($d_{k,n}$) consistently decreases after the second layer and vice-versa for the associated convergence parameter ($a_{k,n}$). The step sizes associated with the data layer thus follow an opposite trend to those associated with the regularization. This indicates that the number of regularization layers may be too low, forcing the network to accelerate the convergence of the variables $(\mathbf{z}_n^j)_{j=1}^J$ (high values for $\gamma_n \tau_j^{-1}$). Figure 13 displays the learned weights of the fully connected

layers, which infer the Cauchy function threshold from the reprojection error cumulative histogram. Figure 11 shows reconstructed ROIs obtained with $\mathcal{A}_2$ (overfitted to each image) (b), the hierarchical TV method (c), and FBP (d). In Example A, the sub-sampling streaks that appear in the FBP reconstruction are not present anymore in the images produced by $\mathcal{A}_2$ and the TV method. More precisely, our architecture $\mathcal{A}_2$ leads to similar results as the TV method in a very reduced number of iterations. If the hierarchical method uses few projection-backprojection operations, it requires 2000 sub-iterations in total to compute the proximity operator of the TV regularization. In contrast, U-RDBDB only uses 25 regularization layers.

In Example B, an intense wire exists in the data but not in the reconstruction grid, which produces high-intensity sub-sampling streaks in the FBP reconstruction. Most of the streaks remain in the TV solution, whereas our architecture $\mathcal{A}_2$ strongly attenuates them. In addition, the noise is reduced, but the resolution of the image is decreased. Because $\mathcal{A}_2$ is designed to minimize the ℓ_2 norm of the error with the noiseless ground truth and a very low number of regularization layers, it tends to selectively smooth some parts of the image to remove remaining streaks. On the contrary, the hierarchical method is data-agnostic; it is designed to reduce sub-sampling streaks only, regardless of the noise. Hence it is interesting to check whether an adjoint mismatch can help reduce this smoothing effect and compensate for the low number of regularization layers.



(a) softplus($b_{k,n,1}$) (b) softplus($b_{k,n,2}$) (c) softplus($b_{k,n,3}$)

(d) softplus($b_{k,n,4}$) (e) softplus($b_{k,n,5}$) (f) softplus($b_{k,n,6}$)

(g) softplus($e_{k,n}$) (h) softplus($d_{k,n}$) (i) softplus($a_{k,n}$)

Fig. 12: Evolution of learned scalar parameters of $\mathcal{A}_2$ along the layers, when the network is trained to overfit Example B.

B. Comparison with deep learning methods and impact of an adjoint mismatch

We now compare all deep learning methods trained on the union of the training sets of Dataset 1 and Dataset 2 ($I = 5000$). Qualitative and quantitative comparisons are made on test sets. Table II reports the performance of our

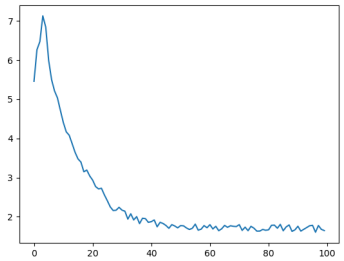


Fig. 13: Learned weights of the fully connected layer $c_{k,n}$, shared across the layers of $\mathcal{A}_2$, when the network is trained to overfit Example B.

three architectures, U-net, and other deep unfolding networks on the first dataset. The metrics obtained with the iterative hierarchical TV algorithm are also added as a reference. First, we see that all supervised methods can perform at least as well as the hierarchical method in a reduced number of iterations (i.e., layers). Then, in both cases, $\mathcal{A}_3$ outperforms $\mathcal{A}_1$ and $\mathcal{A}_2$. Moreover, both $\mathcal{A}_2$ and $\mathcal{A}_3$ perform on average better than the other unfolding networks (PNN, PD-net, and ISTA-net) and U-net for all considered metrics. The results obtained with $\mathcal{A}_1$ are characterized by lower metrics in all cases. This thus provides empirical evidence that performance can be optimized by introducing mismatched adjoints even without learning.

Figure 14 illustrates the reconstruction results for four examples from the test set of Dataset 1. The figure confirms that $\mathcal{A}_2$ and $\mathcal{A}_3$ reduce streaks more effectively than the other unfolding networks. In the last-row example, even though $\mathcal{A}_2$ removes streaks over the colon, it is done at the expense of over-smoothing, which does not appear with $\mathcal{A}_3$. Some images obtained with $\mathcal{A}_1$ contain artificial background patterns, translating into degradation for all metrics. At first sight, the images produced by U-net have fewer artifacts than most deep unfolding networks. However, in the second- and third-row images, U-net introduces an artificial background to the images; in the second image, the background appears darker than in the ground truth. This observation highlights that U-net can hallucinate structures under the sub-sampling streaks of the FBP input. Unfolding networks avoid these hallucinations; by simply alternating between U-net and several consistency layers, PNN already minimizes this effect.

In Table III, we report the performance of the trained networks when tested on Dataset 2. Both our architectures outperform the competing unfolding networks as soon as we include an adjoint mismatch in the data layer. U-net ranks third-best, while PD-net is fourth.

The four reconstructed images displayed in Figure 15 confirm this trend. U-net effectively removes streaks in the first, second, and last images, but it over-smoothes soft tissues, and an offset remains in some images. PNN manages to correct these offsets but at the price of additional streaks artifacts. Regarding our architectures, as for Dataset 1, the anatomical background is better recovered when an adjoint mismatch is considered in the data layer ($\mathcal{A}_2$ and $\mathcal{A}_3$). Still, the images

of the first and third rows obtained with $\mathcal{A}_2$ have a patchy look, often characteristic of TV regularization. This shows that $\mathcal{A}_2$ retains the characteristics of the original optimization problem, avoiding the generation of unexpected content as is possible with U-net. Learning the adjoint of the regularization operators in $\mathcal{A}_3$ allows for more flexibility. It modifies the piece-wise constant prior towards a prior that is learned from the database with a potential acceleration of the iterative reconstruction: the images do not display patchy artifacts. A visual inspection of the images in the last row shows that none of the methods can accurately recover the low contrasts in the brain. This suggests that the number of projections is still too low to enable the visualization of such tissues.

VIII. Conclusion

In this paper, we benefited from the structure of the dual block forward-backward algorithm to define two types of deep learning layers. We proposed U-RDBFB, an architecture built from a nonuniform distribution of these two layers, for reconstructing a region of interest from a limited number of truncated CT projections. Based on the Cauchy function and the ramp filter, a data consistency term and a non-local TV regularization are embedded to limit sub-sampling streaks from intense objects outside the reconstruction grid. Our data term allowed for fitting most data except for the projections of objects with high gradients outside the reconstruction grid. The patchy artifacts yielded by sparse priors on the non-sparse CT images for a very low number of iterations were shown to be mitigated by learning layer-wise adjoints of the regularization operators. Our experiments demonstrated that despite having fewer learnable parameters, U-RDBFB performs favorably compared to state-of-the-art deep learning methods. Our results suggest that including more a priori knowledge can bring more robustness and reliability to the current deep-learning methods. Future work will investigate the robustness of U-RDBFB against changes in the acquisition settings and the trade-off between the image quality in terms of resolution and the total number of layers of U-RDBFB.

Appendix A
Reweighting strategy

Let ϕ be given by (3). It was shown in [55] that, for every $\overline{\zeta} \in \mathbb{R}$, the following convex quadratic function $\tilde{\phi}(\cdot, \overline{\zeta})$, defined for every $\zeta \in \mathbb{R}$ as

$$\tilde{\phi}(\zeta, \overline{\zeta}) = \phi(\overline{\zeta}) + \beta \frac{(\zeta - \overline{\zeta})\overline{\zeta}}{1 + (\overline{\zeta}/\kappa)^2} + \frac{\beta}{2} \frac{(\zeta - \overline{\zeta})^2}{(1 + (\overline{\zeta}/\kappa)^2)}, \quad (32)$$

is a tangent majorant approximation to ϕ at $\overline{\zeta}$, that is

$$(\forall \zeta \in \mathbb{R}) \;\; \tilde{\phi}(\zeta, \overline{\zeta}) \geqslant \phi(\zeta) \quad \text{and} \quad \tilde{\phi}(\overline{\zeta}, \overline{\zeta}) = \phi(\overline{\zeta}). \quad (33)$$

This allows us to deduce a tangent majorant function $\tilde{g}$ of function g at any point $\overline{\mathbf{z}} \in \mathbb{R}^T$: $(\forall \mathbf{z} \in \mathbb{R}^T)$:

$$\begin{aligned}\tilde{g}(\mathbf{z}, \overline{\mathbf{z}}) &= \sum_{t=1}^{T} \phi(\mathbf{z}_t, \overline{\mathbf{z}}_t) \\ &= g(\overline{\mathbf{z}}) + \beta \operatorname{diag}\left(\left(\frac{\overline{\mathbf{z}}_t}{1 + (\overline{\mathbf{z}}_t/\kappa)^2} \right)_{t=1}^{T} \right) (\mathbf{z} - \overline{\mathbf{z}}) \\ &+ \frac{\beta}{2} (\mathbf{z} - \overline{\mathbf{z}})^\top \operatorname{diag}\left(\left(\frac{1}{1 + (\overline{\mathbf{z}}_t/\kappa)^2} \right)_{t=1}^{T} \right) (\mathbf{z} - \overline{\mathbf{z}}) \geqslant g(\mathbf{z}).\end{aligned}$$

Finally, for every $\overline{\mathbf{x}} \in \mathbb{R}^L$, we set

$$(\forall \mathbf{x} \in \mathbb{R}^L) \quad \tilde{f}(\mathbf{x}, \overline{\mathbf{x}}) = \tilde{g}(\mathbf{F}(\mathbf{H}\mathbf{x} - \mathbf{y}); \mathbf{F}(\mathbf{H}\overline{\mathbf{x}} - \mathbf{y})), \quad (34)$$

that satisfies $\tilde{f}(\mathbf{x}, \overline{\mathbf{x}}) \geqslant g(\mathbf{F}(\mathbf{H}\mathbf{x} - \mathbf{y})) = f(\mathbf{x})$. Given this majoration, the iterative reweighting strategy approximates the solution to (8) by the estimate produced by Algorithm 1, where

$$(\forall (\mathbf{x}, \overline{\mathbf{x}}) \in (\mathbb{R}^L)^2) \quad Q(\mathbf{x}, \overline{\mathbf{x}}) = \tilde{f}(\mathbf{x}, \overline{\mathbf{x}}) + r(\mathbf{x}). \quad (35)$$

Appendix B
Implementation of DBFB

The DBFB iterates $(\mathbf{x}_n)_{n \in \mathbb{N}}$ converge to the minimizer of (9) under the following assumptions on the algorithm parameters [37]:

$$\begin{cases} \sigma \geqslant |||\mathbf{B}_0 \mathbf{M}^{-1} \mathbf{B}_0^\top|||, \\ (\forall j \in \{1, \dots, J\}) \quad \tau_j \geqslant |||\mathbf{L}_j \mathbf{M}^{-1} \mathbf{L}_j^\top|||, \\ \gamma_n \in [\epsilon, 2 - \epsilon] \text{ with } \epsilon \in]0, 1] \\ (\exists N \in \mathbb{N} \setminus \{0, 1\})(\forall n \in \mathbb{N}) \quad 0 < \sum_{n'=n}^{n+N-1} \varepsilon_{n'} < N. \end{cases} \quad (36)$$

The first three assumptions are step size range conditions. The last one means that each step **(D)** and **(R)** is performed at least once, every N iterations. Step **(D)** involves the calculation of the proximity operator $\operatorname{prox}_{\gamma_n^{-1} \sigma h_0(\cdot, \mathbf{A}_0 \overline{\mathbf{x}})}$, which has a closed-form [29, Example 24.2], for $(\forall (\mathbf{z}, \overline{\mathbf{z}}) \in (\mathbb{R}^T)^2)$,

$$\begin{aligned}&\operatorname{prox}_{\gamma_n^{-1} \sigma h_0(\cdot, \mathbf{A}_0 \overline{\mathbf{z}})}(\mathbf{z}) = \operatorname{prox}_{\gamma_n^{-1} \sigma \tilde{g}(\cdot - \mathbf{F}\mathbf{y}, \overline{\mathbf{z}} - \mathbf{F}\mathbf{y})}(\mathbf{z}), \\ &= \mathbf{F}\mathbf{y} + \operatorname{prox}_{\gamma_n^{-1} \sigma \tilde{g}(\cdot, \overline{\mathbf{z}} - \mathbf{F}\mathbf{y})}(\mathbf{z} - \mathbf{F}\mathbf{y}) \\ &= \left((\mathbf{F}\mathbf{y})_t + \frac{\mathbf{z}_t - (\mathbf{F}\mathbf{y})_t}{1 + \beta \gamma_n^{-1} \sigma \left(1 + (\overline{\mathbf{z}}_t - (\mathbf{F}\mathbf{y})_t)^2 / \kappa^2\right)^{-1}} \right)_{t=1}^{T}.\end{aligned} \quad (37)$$

Step **(R)** requires calculating the proximity operator of h_j scaled by parameter $\gamma \in]0, +\infty[$. It also has a closed form: for $\mathbf{s} = (\mathbf{s}_1, \dots, \mathbf{s}_J) \in \mathbb{R}^{2JL}$, $\operatorname{prox}_{\gamma h_1}(\mathbf{s}) = \left(\operatorname{prox}_{\gamma r_j}(\mathbf{s}_j) \right)_{j=1}^{J}$, where, for every $\mathbf{z} = (\mathbf{z}_1, \mathbf{z}_2) \in \mathbb{R}^{2L}$,

$$\operatorname{prox}_{\gamma r_j}(\mathbf{z}) = \left(\max\left\{0, 1 - \frac{\gamma \alpha_{j,l}}{\|\mathbf{z}_\ell\|_2}\right\} \mathbf{z}_\ell \right)_{\ell=1}^{L}, \quad (38)$$

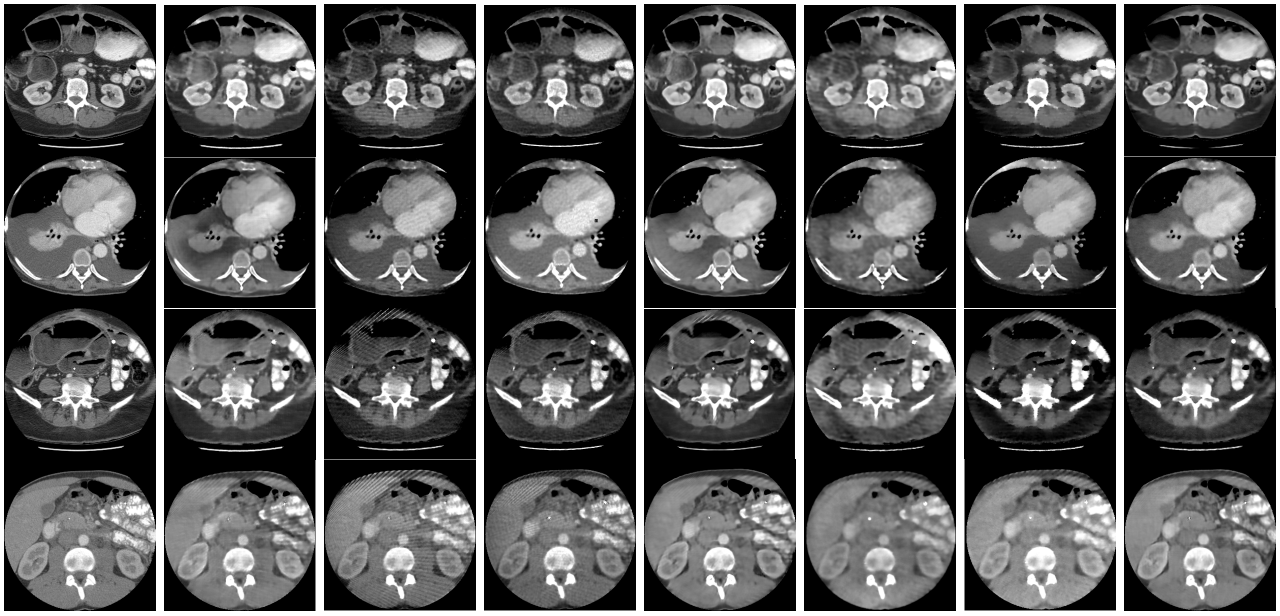
where, for every $\ell \in \{1, \dots, L\}$, $\mathbf{z}_\ell = ((\mathbf{z}_1)_\ell, (\mathbf{z}_2)_\ell) \in \mathbb{R}^2$.

Metrics	$\mathcal{A}_1$	$\mathcal{A}_2$	$\mathcal{A}_3$	U-net	PPN	PD-net	ISTA-net	TV
PSNR	38.5	40.5	**42.1**	40.0	40.3	39.7	35.3	33.9
SSIM	0.959	0.970	**0.974**	0.960	0.936	0.952	0.959	0.903
MAE ($\times 10^{-3}$)	6.65	4.32	**3.12**	5.34	7.80	5.68	5.62	12.5
PieApp	0.592	0.421	**0.389**	0.622	0.791	0.442	0.651	0.653

TABLE II: Quantitative assessment of the reconstructed ROIs. Mean values computed over the test set of Dataset 1.

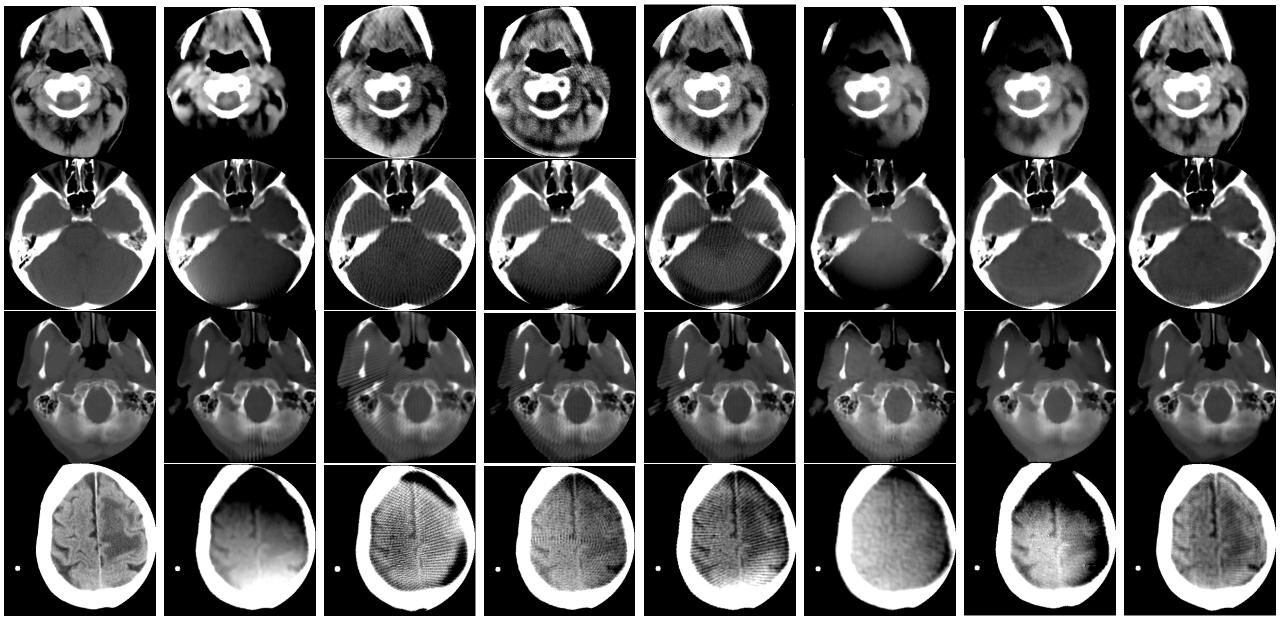
Metrics	$\mathcal{A}_1$	$\mathcal{A}_2$	$\mathcal{A}_3$	U-net	PPN	PD-net	ISTA-net	TV
PSNR	36.2	41.0	**42.8**	40.8	35.0	36.6	33.5	28.1
SSIM	0.967	0.971	**0.979**	0.978	0.972	0.977	0.958	0.948
MAE ($\times 10^{-3}$)	4.05	3.81	**3.12**	3.74	7.98	6.87	11.6	16.5
PieApp	0.362	0.289	**0.267**	0.310	0.403	0.269	0.438	0.476

TABLE III: Quantitative assessment of the reconstructed ROIs. Mean values computed over the test set of Dataset 2.



(a) $\overline{\mathbf{x}}_{\mathrm{ROI}}$ (b) U-net (c) ISTA-net (d) PD-net (e) PNN (f) $\mathcal{A}_1$ (g) $\mathcal{A}_2$ (h) $\mathcal{A}_3$

Fig. 14: Reconstructed ROIs using different deep learning methods on four examples in the test set of Dataset 1.



(a) $\overline{\mathbf{x}}_{\mathrm{ROI}}$ (b) U-net (c) ISTA-net (d) PD-net (e) PNN (f) $\mathcal{A}_1$ (g) $\mathcal{A}_2$ (h) $\mathcal{A}_3$

Fig. 15: Reconstructed ROIs using different deep learning methods on four examples in the test set of Dataset 2.

REFERENCES

[1] R. Chityala, K. Hoffmann, D. Bednarek, and S. Rudin. Region of interest (ROI) computed tomography. In *Medical Imaging 2004: Physics of Medical Imaging*, vol. 5368, pp. 534–541. SPIE, 2004.

[2] L. Yu, X. Liu, S. Leng, J. Kofler, J. Ramirez-Giraldo, M. Qu, J. Christner, J. Fletcher, and C. McCollough. Radiation dose reduction in computed tomography: techniques and future perspective. *Imaging in Medicine*, vol. 1, pp. 65–84, 2009.

[3] E. Sidky and X. Pan. Image reconstruction in circular cone-beam computed tomography by constrained, total-variation minimization. *Physics in Medicine & Biology*, vol. 53, pp. 4777–4807, 2008.

[4] J. Bian, J. Siewerdsen, X. Han, E. Sidky, J. Prince, C. Pelizzari, and X. Pan. Evaluation of sparse-view reconstruction from flat-panel-detector

cone-beam CT. *Physics in Medicine and Biology*, vol. 55, pp. 6575–6599, 2010.

[5] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, vol. 60, pp. 259–268, 1992.

[6] H. Langet, C. Riddell, A. Reshef, Y. Trousset, A. Tenenhaus, E. Lahalle, G. Fleury, and N. Paragios. Compressed-sensing-based content-driven hierarchical reconstruction: Theory and application to C-arm cone-beam tomography. *Medical Physics*, vol. 42, pp. 5222–5237, 2015.

[7] H. Yu and G. Wang. Compressed sensing based interior tomography. *Physics in Medicine & Biology*, vol. 54, pp. 2791, 2009.

[8] P. Paleo and A. Mirone. Efficient implementation of a local tomography reconstruction algorithm. *Advanced Structural and Chemical Imaging*, vol. 3, pp. 5, 2017.

[9] B. Hamelin, Y. Goussard, J.-P. Dussault, G. Cloutier, G. Beaudoin, and G. Soulez. Design of iterative ROI transmission tomography reconstruction procedures and image quality analysis. *Medical Physics*, vol. 37, pp. 4577–4589, 2010.

[10] A. Ziegler, T. Nielsen, and M. Grass. Iterative reconstruction of a region of interest for transmission tomography. *Medical Physics*, vol. 35, pp. 1317–1327, 2008.

[11] M. Savanier, E. Chouzenoux, J.-C. Pesquet, and C. Riddell. Unmatched preconditioning of the proximal gradient algorithm. *IEEE Signal Processing Letters*, vol. 29, pp. 1122–1126, 2022.

[12] R. Tirer and R. Giryes. On the convergence rate of projected gradient descent for a back-projection based objective. *SIAM Journal on Imaging Sciences*, vol. 14, pp. 1504–1531, 2021.

[13] T. Tirer and R. Giryes. Back-projection based fidelity term for ill-posed linear inverse problems. *IEEE Transactions on Image Processing*, vol. 29, pp. 6164–6179, 2020.

[14] M. Nilchian, C. Vonesch, P. Modregger, M. Stampanoni, and M. Unser. Iterative FBP for improved reconstruction of X-ray differential phase-contrast tomograms. In *Proceedings of the 2013 IEEE 10th International Symposium on Biomedical Imaging*, pp. 1260–1263, 7-11 April 2013, San Francisco, CA, USA.

[15] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Proceedings of the 18th International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI) 2015*, pp. 234–241, 5-9 October 2015, Munich, Germany.

[16] Y. Han, J. J. Yoo, and J. C. Ye. Deep residual learning for compressed sensing CT reconstruction via persistent homology analysis. Preprint 2016. Available: https://arxiv.org/abs/1611.06391.

[17] K. Jin, M. T. McCann, E. Froustey, and M. Unser. Deep convolutional neural network for inverse problems in imaging. *IEEE Transactions on Image Processing*, vol. 26, pp. 4509–4522, 2017.

[18] E. Sidky, I. Lorente, J. G. Brankov, and X. Pan. Do CNNs solve the CT inverse problem. *IEEE Transactions on Biomedical Engineering*, vol. 68, pp. 1799–1810, 2021.

[19] V. Monga, Y. Li, and Y. Eldar. Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Processing Magazine*, vol. 38, pp. 18–44, 2021.

[20] G. Wang, J. C. Ye, and B. De Man. Deep learning for tomographic image reconstruction. *Nature Machine Intelligence*, vol. 2, pp. 737–748, 2020.

[21] S. Wang, S. Fidler, and R. Urtasun. Proximal Deep Structured Models. In *Proceedings of the 30th Annual Conference on Neural Information Processing Systems, NIPS 2016*, pp. 865–873, 5-10 December 2016, Barcelona, Spain.

[22] H. T. V. Le, N. Pustelnik, and M. Foare. The faster proximal algorithm, the better unfolded deep learning architecture ? The study case of image denoising. In *Proceedings of the 30th European Signal Processing Conference, EUSIPCO 2022*, 29 August - 2 September 2022, Belgrade, Serbia.

[23] C. Bertocchi, E. Chouzenoux, M.-C. Corbineau, J.-C. Pesquet, and M. Prato. Deep unfolding of a proximal interior point method for image restoration. *Inverse Problems*, vol. 36, pp. 034005, 2020.

[24] Y. Yang, J. Sun, H. Li, and Z. Xu. Deep ADMM-Net for Compressive Sensing MRI. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pp. 10–18, 5-10 December 2016, Barcelona, Spain.

[25] J. Adler and O. Oktem. Solving ill-posed inverse problems using iterative deep neural networks. *Inverse Problems*, vol. 33, pp. 124007, 2017.

[26] J. Zhang and B. Ghanem. ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing. *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1828–1837, 18-23 June 2018, Salt Lake City, UT, USA.

[27] N. Pustelnik, A. Benazza-Benhayia, Y. Zheng, and J.-C. Pesquet. *Wavelet-Based Image Deconvolution and Reconstruction*, pages 1–34. American Cancer Society, 2016.

[28] F. Abboud, E. Chouzenoux, J.-C. Pesquet, J.-H. Chenot, and L. Laborelli. Dual Block Coordinate Forward-Backward Algorithm with Application to Deconvolution and Deinterlacing of Video Sequences. *Journal of Mathematical Imaging and Vision*, vol. 59, pp. 415–431, November 2017.

[29] H. H. Bauschke and P. L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. 2017.

[30] N. Ouzir, A. Basarab, O. Lairez, and J.-Y. Tourneret. Robust optical flow estimation in cardiac ultrasound images using a sparse representation. *IEEE Transactions on Medical Imaging*, vol. 38, pp. 741–752, 2019.

[31] D. Kazantsev, F. Bleichrodt, T. van Leeuwen, A. Kaestner, P. J. Withers, K. J. Batenburg, and P. D. Lee. A novel tomographic reconstruction method based on the robust student's t function for suppressing data outliers. *IEEE Transactions on Computational Imaging*, vol. 3, pp. 682–693, 2017.

[32] D. Andrews and F. Hampel. Robust estimates of location. In *Robust Estimates of Location*. Princeton University Press, 1972.

[33] G. L Zeng. Revisit of the ramp filter. In *Proceedings of the 2014 IEEE Nuclear Science Symposium and Medical Imaging Conference*, pp. 1–6, 8-15 November 2014, Seattle, WA USA.

[34] S. Kindermann, S. Osher, and P. W. Jones. Deblurring and denoising of images by nonlocal functionals. *Multiscale Modeling & Simulation*, vol. 4, pp. 1091–1115, 2006.

[35] G. Gilboa and S. Osher. Nonlocal operators with applications to image processing. *Multiscale Modeling & Simulation*, vol. 7, pp. 1005–1028, 2009.

[36] P. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering*, pp. 185–212. Springer, 2011.

[37] F. Abboud, E. Chouzenoux, J.-C. Pesquet, J.-H. Chenot, and L. Laborelli. Dual block-coordinate forward-backward algorithm with application to deconvolution and deinterlacing of video sequences. *Journal of Mathematical Imaging and Vision*, vol. 59, pp. 415–431, 2017.

[38] E. Chouzenoux, J.-C. Pesquet, C. Riddell, M. Savanier, and Y. Trousset. Convergence of proximal gradient algorithm in the presence of adjoint mismatch. *Inverse Problems*, vol. 37, pp. 065009, 2021.

[39] T. Elfving and P. C. Hansen. Unmatched projector/backprojector pairs: perturbation and convergence analysis. *SIAM Journal on Scientific Computing*, vol. 40, pp. A573–A591, 2018.

[40] D. Lorenz, S. Rose, and F. Schöpfer. The randomized Kaczmarz method with mismatched adjoint. *BIT Numerical Mathematics*, vol. 58, pp. 1079–1098, 2018.

[41] E. Chouzenoux, A. Contreras, J.-C. Pesquet, and M. Savanier. Convergence Results for Primal-Dual Algorithms in the Presence of Adjoint Mismatch. Preprint, 2022. Available: https://hal.archives-ouvertes.fr/hal-03654126/.

[42] G. Zeng. Counter examples for unmatched projector/backprojector in an iterative algorithm. *Chinese Journal of Academic Radiology*, 2019.

[43] G. Zeng and G. Gullberg. Unmatched projector/backprojector pairs in an iterative reconstruction algorithm. *IEEE Transactions on Medical Imaging*, 2000.

[44] E. Sidky, E. C. Hansen, J. Jorgensen, and X. Pan. Iterative image reconstruction for CT with unmatched projection matrices using the generalized minimal residual algorithm. Preprint, 2022. Available: https://arxiv.org/abs/2201.07408.

[45] Z. Wang, H. Li, W. Ouyang, and W. Wang. Learnable histogram: Statistical context features for deep neural networks. In *Proceedings of the 14th European Conference on Computer Vision (ECCV)*, Amsterdam, The Netherlands, 2016.

[46] M. Ronchetti. TorchRadon: Fast differentiable routines for computed tomography. Preprint, 2020. Available: https://arxiv.org/abs/2009.14788.

[47] I. Safran and O. Shamir. On the quality of the initial basin in overspecified neural networks. In *Proceedings of the 33rd International Conference on Machine Learning*, pp. 774–782, June 19 – June 24, 2016, New York City, NY, USA.

[48] J. Martens and I. Sutskever. Learning recurrent neural networks with hessian-free optimization. In *Proceedings of the 28th International Conference on Machine Learning*, pp. , 28 June to 2 July 2011, Bellevue, Washington, USA.

[49] E. Prashnani, H. Cai, Y. Mostofi, and P. Sen. Pieapp: Perceptual image-error assessment through pairwise preference. In *Proceedings of the 2018*

IEEE Conference on Computer Vision and Pattern Recognition, pp. 1808–1817, 18-23 June 2018, Salt Lake City, UT, USA.

[50] D. Donoho and Y. Tsaig. Fast solution of ℓ_1 -norm minimization problems when the solution may be sparse. *IEEE Transactions on Information Theory*, vol. 54, pp. 4789–4812, 2008.

[51] L. Xiao and T. Zhang. A proximal-gradient homotopy method for the sparse least-squares problem. *SIAM Journal on Optimization*, vol. 23, pp. 1062–1091, 2013.

[52] D. Gilton, G. Ongie, and R. Willett. Neumann networks for linear inverse problems in imaging. *IEEE Transactions on Computational Imaging*, vol. 6, pp. 328–343, 2020.

[53] J. Adler and O. Öktem. Learned primal-dual reconstruction. *IEEE Transactions on Medical Imaging*, vol. 37, pp. 1322–1332, 2018.

[54] J. Cheng, H. Wang, L. Ying, and D. Liang. Model learning: Primal dual networks for fast MR imaging. Preprint, 2019. Available: https://arxiv.org/abs/1908.02426.

[55] E. Chouzenoux, A. Jezierska, J.-C. Pesquet, and H. Talbot. A majorize-minimize subspace approach for ℓ_2-ℓ_0 image regularization. *SIAM Journal on Imaging Sciences*, vol. 6, pp. 563–591, 2013.