HAL open science

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

To cite this version:

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. Inria Saclay. 2022. hal-03881278v3

HAL Id: hal-03881278

https://hal.science/hal-03881278v3

Submitted on 17 May 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

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 presents a new method for reconstructing regions of interest (ROI) from a limited number of computed tomography (CT) measurements. Classical model-based iterative reconstruction methods lead to images with predictable features. Still, they often suffer from tedious parameterization and slow convergence. On the contrary, deep learning methods are fast, and they can reach high reconstruction quality by leveraging information from large datasets, but they lack interpretability. At the crossroads of both methods, deep unfolding networks have been recently proposed. Their design includes the physics of the imaging system and the steps of an iterative optimization algorithm. Motivated by the success of these networks for various applications, we introduce an unfolding neural network called U-RDBFB designed for ROI CT reconstruction from limited data. Few-view truncated data are effectively handled thanks to a robust non-convex data fidelity term combined with a sparsity-inducing regularization function. We unfold the Dual Block coordinate Forward-Backward (DBFB) algorithm, embedded in an iterative reweighted scheme, allowing the learning of key parameters in a supervised manner. Our experiments show an improvement over several state-of-the-art methods, including a model-based iterative scheme, a multi-scale deep learning architecture, and other 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 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] and truncated measurements (or projections). The inverse problem of reconstructing an ROI from a set of truncated projections is ill-posed. In practice, it is often combined with angular sub-sampling for fast or further dose-saving acquisitions.

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).

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 these works achieve a reduction in angular sub-sampling artifacts compared to analytical reconstruction methods, truncated projections still challenge MBIR [6], especially in terms of computation time.

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 holds no clinical value. Thus, one more practical solution is to consider an intermediate smaller grid size with a "margin" outside of the ROI [8]. This achieves a faster and more stable ROI reconstruction in general. Yet, when dense objects such as metallic cables or needles are outside the reconstruction grid, the reprojection of the extended ROI contains high-frequency errors. Moreover, when too few projections are used for reconstruction, such objects suffer from aliasing, and additional streak artifacts can degrade the reconstructed ROI [6]. Another approach is to reconstruct the entire object with large voxels and then subtract the reprojection of the exterior from the data before reconstructing only the ROI from the subtracted data [9], [10]. This approach produces a low-frequency approximation of the exterior of the ROI. However, such an approximation is poor in the presence of dense objects in the exterior of the ROI, and unwanted high-frequency content remains after subtraction that must again be dealt with.

Apart from the choice for the reconstruction grid, a way of reducing reconstruction time in MBIR is through preconditioning techniques. However, these techniques are often too restrictive for CT reconstruction when proximal algorithms are used [11]; hence the emergence of heuristics 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 involves weighing a least-squares data fidelity term by the ramp filter. Each gradient step then performs an approximate inversion of the forward 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 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. Thus, designing MBIR methods with a low computational cost for CT imaging is still challenging.

An increasingly popular alternative to MBIR is convolutional neural networks (CNN) due to their increased expressivity and fast inference. CNNs, and in particular 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]. The deep unfolding paradigm [19], [20] circumvents this issue by offering a way to include a priori information in a neural network. Unfolding networks have been applied to many inverse problems, 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 learning of 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 design a deep unfolding network for addressing the problem of ROI image reconstruction from few-view truncated measurements. Specifically:

- We first introduce an original cost function involving a Cauchy fidelity term and a semi-local total-variation regularization to limit sub-sampling streaks from objects inside and outside the reconstruction grid.
- We then advocate for an iterative optimization algorithm combining for the first time an instance of the dual block forward-backward algorithm (DBFB) [28] with an iterative reweighted scheme.
- We propose a neural network architecture inspired by this algorithm, allowing supervised parameter learning and fast reconstruction on the GPU.
- We evaluate the performance and generalization of our network on three datasets against state-of-the-art methods such as MBIR, other deep unfolding networks, and classical deep learning post-processing.

This paper is organized as follows: Section II introduces our notation, and Section III provides a mathematical formulation of our reconstruction problem. Section IV presents our cost function and introduces a convergent iterative algorithm to minimize it. Section V then explains how this algorithm is unfolded into a deep learning architecture and discusses our training strategy. This is followed by experiments (Section VI), results (Section VII), and discussions (Section VIII).

II. Notation

Throughout the paper, the underlying image space is 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] $\mathrm{prox}_g(\mathbf{x}) = \underset{\mathbf{z} \in \mathbb{R}^L}{\mathrm{argmin}} \quad \big(g(\mathbf{z}) + \frac{1}{2}\|\mathbf{x} - \mathbf{z}\|^2\big)$.

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. Problem formulation

We consider a 1D detector array of $B \in \mathbb{N}$ bins rotating around an object. The detector is too short to measure the projections of the entire object; its size defines a circular ROI we aim to reconstruct. Let $S \in \mathbb{N}$ be the number of projection angles. The vector of truncated sub-sampled tomographic data is $\mathbf{y} \in \mathbb{R}^T$ with $T = B\,S$.

A reconstruction of the object 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 $\mathbf{n} \in \mathbb{R}^T$ accounts for some acquisition noise, $\overline{\mathbf{x}}_{\mathrm{G}} \in \mathbb{R}^L$ is the scanned image restricted to a grid G with support larger than the ROI but smaller than that of the entire object, and $\mathbf{H} \in \mathbb{R}^{T \times L}$ is the projector that models projection over this intermediary grid G. 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 full projector, corresponding to the pixels outside of G, to zero.

When the grid G corresponds to the ROI, (1) assumes that the image values are 0 outside of the ROI. This assumption is not necessarily valid for truncated data, so (1) does not hold. Hereafter, we suppose that the grid is extended beyond the ROI so that no assumption is made about the values outside the ROI.

MBIR finds an estimate of $\overline{\mathbf{x}}_{\mathrm{G}}$ by computing a minimizer of a penalized cost function that can, for instance, be written as the sum of a data fidelity term f involving $\mathbf{H}$ and $\mathbf{y}$, and a regularization term r, as

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

Our proposed neural network will unfold a baseline algorithm for solving the MBIR problem (2) for a specific choice of f and r. The next section presents the two terms of the cost function and the baseline nested-loop minimization algorithm.

IV. Model-based iterative reconstruction

A. Cost function

1) Cauchy data fidelity: When objects with high gradients (metallic wires or needles) do not belong to the reconstruction grid G, the error between the data and the reprojection of the estimate over G contains outliers at the projections of those high gradients. Angular sub-sampling of these outliers leads to streaks originating from these objects, i.e., from outside the grid. This means the data should not be trusted equally but through a statistical analysis different from measurement noise. To avoid the streaks, 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 an M-estimator.

Here, we focus on the Cauchy estimator ϕ, which is a redescending M-estimator [32] i.e., its derivative decreases to zero on $]-\infty,\kappa]\cap[\kappa,+\infty[$. It is 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 $\beta > 0$ is a weighting term and $\kappa > 0$ monitors the sensitivity to outliers: the lower κ, the lower the influence of the outliers.

A graphical comparison of the Cauchy function (3) and the quadratic function $\phi(\cdot) = \frac{\beta}{2}(\cdot)^2$ is displayed in Figure 1 for $\beta = 1$ and various values of κ.

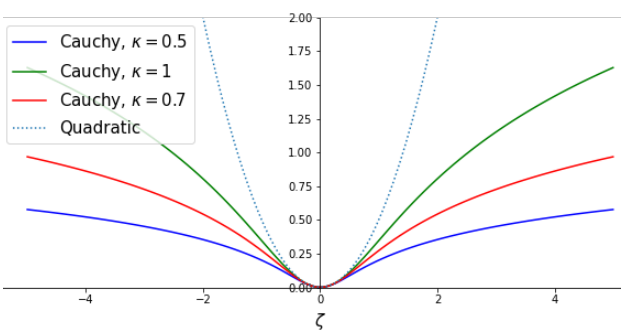


Fig. 1: Comparison between the Cauchy and the quadratic functions.

Our data fidelity term f reads

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

with

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

2) Semi-local total variation: Sub-sampling streaks are commonly handled by total variation (TV) regularization [3], [5], [6]. Semi-local variants (STV) [34] extend TV in a neighborhood of pixels indexed in $\Lambda_J = \{-J,\dots,J\}\setminus\{0\}$:

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

Hereinabove $\ell \in \{1,\dots,L\}$ 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 $j \in \{1,\dots,J\}$ and $J = 6$. Moreover, for every $j \in \{1,\cdots,J\}$, we define $\nabla_j = \left[\mathbf{V}_j^\top \ \mathbf{V}_{-j}^\top\right]^\top \in \mathbb{R}^{2L\times L}$ and, for every $\mathbf{z} = (\mathbf{z}_1,\mathbf{z}_2) \in \mathbb{R}^{2L}$, $r_j(\mathbf{z}) = \sum_{\ell=1}^{L}\alpha_{j,\ell}\sqrt{(\mathbf{z}_1)_\ell^2+(\mathbf{z}_2)_\ell^2}$. 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 regularization for constant values of these parameters and $J = 1$.

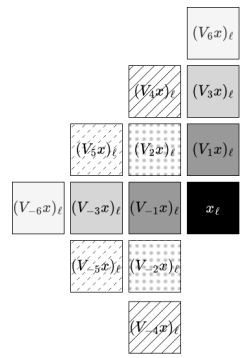


Fig. 2: Shift operators $(V_j)_{j\in\Lambda_6}$ applied to a given pixel position ℓ

We add a nonnegativity constraint on the pixel values and a quadratic term $\frac{1}{2}\|\mathbf{x}\|_{\mathbf{M}}^2 = \mathbf{x}^\top\mathbf{M}\mathbf{x}$ to the STV regularization. Here, matrix $\mathbf{M} = \mathrm{diag}((m_\ell)_{\ell=1}^L) \in \mathcal{S}_L^+$ is such that, for every $\ell \in \{1,\dots,L\}$, $m_\ell = 1$ if the ℓ-th entry x_ℓ of vector $\mathbf{x}$ belongs to the ROI, and $m_\ell = \xi > 1$ otherwise. Thus $\mathbf{M} \in \mathcal{S}_L^+$ acts as a mask, limiting high values outside of the ROI.

Altogether, our regularization function in (2) reads

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

To our knowledge, STV regularization has not been used in CT for handling sub-sampling streaks. In this work, STV provides extra capacity compared to TV for learning. The Cauchy fidelity term has been used in ultrasound imaging [30] and in CT imaging [31] for mitigating the ring artifacts that appear due to defective detector bins only.

B. Minimization algorithm

1) Reweighting for non-convex data fidelity: Given our choices for r and f, Problem (2) becomes

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

Because of the non-convexity of g, 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,

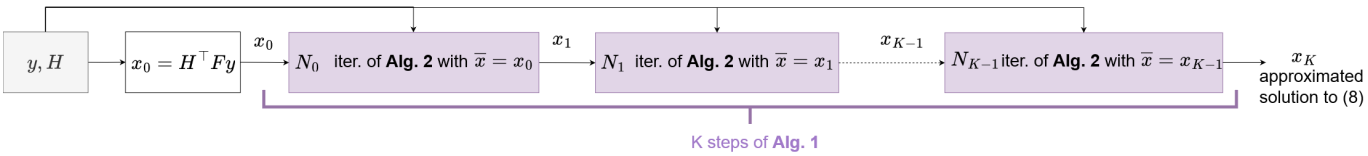


Fig. 3: Baseline RDBFB algorithm for solving (8): K steps of Alg. 1, where each step k consists of applying N_k iterations of Alg. 2.

Algorithm 1 Iterative reweighting strategy for Problem (8)

Input: Number of iterations $K \in \mathbb{N}^*$, $\mathbf{x}_0 \in \mathbb{R}^L$
for $k = 0$ to $K-1$ **do**
Define majorant function $Q(\mathbf{x}; \overline{\mathbf{x}})$ using (10)

$$\mathbf{x}_{k+1} = \underset{\mathbf{x} \in \mathbb{R}^L}{\operatorname{argmin}} \; Q(\mathbf{x}, \mathbf{x}_k) \tag{9}$$

end for
Output: $\mathbf{x}_K$ approximating the solution to (8)

where, for every $\overline{\mathbf{x}} \in \mathbb{R}^L$, the surrogate function $Q(\cdot, \overline{\mathbf{x}})$ is

$$Q(\mathbf{x}, \overline{\mathbf{x}}) = \iota_{[0,+\infty[^L}(\mathbf{x}) + h_0(\mathbf{B}_0\mathbf{x}; \mathbf{B}_0\overline{\mathbf{x}}) + h_1(\mathbf{B}_1\mathbf{x}) + \frac{1}{2}\|\mathbf{x}\|^2_{\mathbf{M}}, \tag{10}$$

where $\forall (\mathbf{x}, \overline{\mathbf{x}}) \in (\mathbb{R}^L)^2$, with $\mathbf{B}_0 = \mathbf{H} \in \mathbb{R}^{T \times L}$,

$$\begin{aligned} \mathbf{B}_1 &= \begin{bmatrix} \nabla_1^\top & \cdots & \nabla_J^\top \end{bmatrix}^\top \in \mathbb{R}^{2JL \times L} \\ h_0(\cdot; \mathbf{B}_0\overline{\mathbf{x}}) &= \tilde{g}(\cdot - \mathbf{y}; \mathbf{B}_0\overline{\mathbf{x}} - \mathbf{y}) \\ h_1(\mathbf{B}_1\mathbf{x}) &= \sum_{j=1}^{J} r_j(\nabla_j \mathbf{x}). \end{aligned} \tag{11}$$

Let $(\mathbf{x}_k)_{k \in \mathbb{N}}$ be generated by Algorithm 1. The cost sequence value (defined from (8)) monotonically converges.

2) Dual block coordinate forward-backward algorithm: The k-th iteration of Algorithm 1 requires to solve (9), which amounts to minimizing the function $Q(\cdot, \overline{\mathbf{x}})$, with $\overline{\mathbf{x}}$ equals to the current iterate $\mathbf{x}_k$. Since $\mathbf{M} \in \mathcal{S}_L^+$, $Q(\cdot, \overline{\mathbf{x}})$ is strongly convex for every $\overline{\mathbf{x}} \in \mathbb{R}^L$. The minimization (9) is hence well-defined, with a unique solution that can be conveniently obtained using the dual forward-backward algorithm [22], [36]. An accelerated and light version of this algorithm is its block coordinate version (DBFB) [37], which allows for accessing the proximity operators of h_0 and h_1 separately.
Algorithm 2 describes $N \in \mathbb{N}^*$ iterations of DBFB. The output $\boldsymbol{x}_N$ generated by DBFB with input $\mathbf{x}_k$ then defines $\mathbf{x}_{K+1}$ in Algorithm 1. For every $n \in \{1, \dots, N\}$, DBFB updates the main primal variable $\boldsymbol{x}_n$ as well as two dual variables $\mathbf{z}_n^0 \in \mathbb{R}^T$ and $\mathbf{s}_n^1 \in \mathbb{R}^{2JL}$, associated with the data fidelity (**data step (D)**) or the regularization (**regularization step (R)**) terms, respectively. Each dual variable is activated (or not) at iteration n according to a binary variable ε_n.
When $N \to \infty$, the DBFB sequence $(\boldsymbol{x}_n)_{n \in \mathbb{N}}$ converge to the solution to (9) under the following assumptions on the algorithm parameters [37]:

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

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

Input: Number of iterations $N \in \mathbb{N}^*$, tangent point $\overline{\mathbf{x}} \in \mathbb{R}^L$, initial dual variables $\mathbf{z}_0^0 \in \mathbb{R}^T, (\forall j \in \{1, \dots, J\})\ \mathbf{z}_0^j \in \mathbb{R}^{2L}$ with constant J defined in (6), operators $\mathbf{B}_0$ and $\mathbf{B}_1$ defined in (11), stepsizes $(\sigma, \tau_1, \dots, \tau_J) \in]0, +\infty[^{J+1}$.

$\mathbf{s}_0^1 = (\mathbf{z}_0^1, \dots, \mathbf{z}_0^J)$
$\boldsymbol{\Sigma} = \operatorname{diag}(\tau_1, \dots, \tau_J)$
$\mathbf{w}_0 = -\mathbf{M}^{-1}(\mathbf{B}_0^\top \mathbf{z}_0^0 + \mathbf{B}_1^\top \mathbf{s}_0^1)$
For $n = 0, 1 \dots, N$
- $(\mathbf{z}_n^1, \dots, \mathbf{z}_n^J) \equiv \mathbf{s}_n^1$
- $\boldsymbol{x}_n = \operatorname{proj}_{[0,+\infty[^L}(\mathbf{w}_n)$
- 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)**
  - $\tilde{\mathbf{s}}_n^1 = \mathbf{s}_n^1 + \gamma_n \boldsymbol{\Sigma}^{-1} \mathbf{B}_1 \boldsymbol{x}_n$
  - $\mathbf{s}_{n+1}^1 = \tilde{\mathbf{s}}_n^1 - \gamma_n \boldsymbol{\Sigma}^{-1} \operatorname{prox}_{\gamma_n^{-1}\boldsymbol{\Sigma} h_1}(\gamma_n^{-1}\boldsymbol{\Sigma}\tilde{\mathbf{s}}_n^1)$
  - $\mathbf{w}_{n+1} = \mathbf{w}_n - \mathbf{M}^{-1}\mathbf{B}_1^\top(\mathbf{s}_{n+1}^1 - \mathbf{s}_n^1)$
  - $\mathbf{z}_{n+1}^0 = \mathbf{z}_n^0$

Output: $\boldsymbol{x}_N$ approximating the minimizer of $Q(\cdot, \overline{\mathbf{x}})$.

The first three assumptions are stepsize range conditions. The last one means that each step **(D)** and **(R)** is performed at least once every M iterations. The practical implementation of the proximal operators involved in DBFB for our choices for h_0 and h_1 is discussed in Appendix B.

The overall iterative strategy for approximating the solution to (8) consists of applying Algorithm 1, where, for every $k \in \mathbb{N}^*$, $N_k \in \mathbb{N}^*$ iterations of Algorithm 2 are used as an inner solver with $\overline{\mathbf{x}} = \mathbf{x}_k$ to compute $\mathbf{x}_{k+1}$ in (9). We call the resulting iterations *reweighted DBFB (RDBFB) algorithm* (see Figure 3). Such a combination of an iterative reweighted algorithm with dual ascent steps is original to our knowledge. In the context of CT, iterative reweighted algorithms usually involve surrogates to the regularization term [57], [58] rather than to the data fidelity, as done here.

V. Unfolded reconstruction

Hereinafter, we present a deep neural network, designated as U-RDBFB (Unfolded Reweighted DBFB), by unfolding all the steps of RDBFB. Specifically, the network mimics the application of K iterations of Algorithm 1, as K main layers, each of them grouping $N_k \in \mathbb{N}$ iterations of Algorithm 2. This yields an architecture with $\sum_{k=0}^{K-1} N_k$ layers in total.

A. From RDBFB iterations to U-RDBFB layers

The deep unfolding paradigm recasts every step of Algorithm 2 as one neural network layer: step **(D)** becomes $\mathcal{L}_D$ ($\varepsilon_n = 0$) and step **(R)** becomes $\mathcal{L}_R$ ($\varepsilon_n = 1$). It requires truncating the number of layers drastically. To optimize the depth of our network, we propose two modifications of the steps **(D)** and **(R)** to construct the corresponding layers. These modifications stem from recent works [38]–[41] that have shown the potential to replace adjoint operators with surrogates to accelerate the convergence of MBIR methods [42]–[44]. In CT, a frequently encountered operator is the ramp filter $\mathbf{F}$, which satisfies $\mathbf{F}\mathbf{H}\mathbf{H}^\top \approx \mathrm{Id}$. Thus, to improve conditioning and allow larger values for the step sizes associated with layer $\mathcal{L}_D$, hence a lower number of such a layer, we replace $\mathbf{B}_0$ in Algorithm 2 with $\mathbf{FH}$.
By setting $\nu_{n,0} = \gamma_n \sigma^{-1}$ and by using the relation

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

we define layer $\mathcal{L}_{\mathrm{D}}$ as

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

$$\left|\begin{array}{l} \boldsymbol{x}_n = \mathrm{proj}_{[0,+\infty[^L}(\mathbf{w}_n) \\ \mathbf{u}_n = \mathbf{z}_n^0 + \nu_{n,0}\mathbf{F}(\mathbf{H}\boldsymbol{x}_n - \mathbf{y}) \\ \mathbf{z}_{n+1}^0 = \mathbf{u}_n - \nu_{n,0}^{-1}\,\mathrm{prox}_{\nu_{n,0}\tilde{g}(\cdot;\mathbf{FH}\overline{\mathbf{x}}-\mathbf{y})}(\nu_{n,0}\mathbf{u}_n) \\ \mathbf{w}_{n+1} = \mathbf{w}_n - \mathbf{M}^{-1}\mathbf{H}^\top(\mathbf{z}_{n+1}^0 - \mathbf{z}_n^0) \\ \mathbf{z}_{n+1}^j = \mathbf{z}_n^j \quad (\forall j \in \{1,\dots,J\}). \end{array}\right. \tag{12}$$

Similarly, for step **(R)**, we unfold by replacing the adjoint of the regularization operator $\mathbf{B}_1$ with $\tilde{\mathbf{B}}_1 = \begin{bmatrix}\tilde{\nabla}_1^\top & \cdots & \tilde{\nabla}_j^\top\end{bmatrix}^\top$. Setting, for every $j \in \{1,\dots,K\}$, $\nu_{n,j} = \gamma_n \tau_j^{-1}$ yield the following regularization layer $\mathcal{L}_{\mathrm{R}}$:

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

$$\left|\begin{array}{l} \boldsymbol{x}_n = \mathrm{proj}_{[0,+\infty[^L}(\mathbf{w}_n) \\ \text{For } j \in \{1,\dots,J\} \\ \quad \left|\begin{array}{l} (\forall \ell \in \{1,\dots,L\}) \\ (\mathbf{z}_{n+1}^j)_\ell = \dfrac{\left(\mathbf{z}_n^j + \nu_{n,j}\nabla_j \boldsymbol{x}_n\right)_\ell}{\max\left\{1, \| \left(\mathbf{z}_n^j + \nu_{n,j}\nabla_j \boldsymbol{x}_n\right)_\ell \|_2/\alpha_{j,\ell}\right\}} \end{array}\right. \\ \mathbf{w}_{n+1} = \mathbf{w}_n - \mathbf{M}^{-1}\sum_{j=1}^J \tilde{\nabla}_j(\mathbf{z}_{n+1}^j - \mathbf{z}_n^j) \\ \mathbf{z}_{n+1}^0 = \mathbf{z}_n^0. \end{array}\right. \tag{13}$$

Note that $\mathcal{L}_R$ does not involve $\overline{\mathbf{x}}$.

B. Total architecture

The total architecture of U-RDBFB, denoted $\mathcal{A}$, can be summarized as

$$\mathcal{A} = \mathcal{L}^{K-1} \circ \cdots \circ \mathcal{L}^0. \tag{14}$$

For $0 \leqslant k \leqslant K-1$, $\mathcal{L}^k$ corresponds to a sequence of N_k layers $\mathcal{L}_D$ or $\mathcal{L}_R$ and implements the following update:

$$(\mathbf{z}_{0,k+1}, \mathbf{x}_{k+1}) = \mathcal{L}^k(\mathbf{z}_{0,k}, \mathbf{x}_k; \Theta_k), \tag{15}$$

where

- $\mathbf{x}_k$ is the current reweighted estimate $\overline{\mathbf{x}}$ in $\mathcal{L}_{\mathrm{R}}$-$\mathcal{L}_{\mathrm{D}}$ ($\mathbf{x}_0 = \mathbf{H}^\top\mathbf{F}\mathbf{y}$).
- $\mathbf{x}_{k+1}$ is the next reweighted estimate; it is equal to $\boldsymbol{x}_{N_k-1}$ given by the N_k-th layer $\mathcal{L}_{\mathrm{R}}$-$\mathcal{L}_{\mathrm{D}}$.
- $\mathbf{z}_{0,k} \in \mathbb{R}^T \times (\mathbb{R}^{2L})^J$ is the initial value of the variables $(\mathbf{z}_0^j)_{0\leqslant j\leqslant J}$ for layers $\mathcal{L}_{\mathrm{R}}$-$\mathcal{L}_{\mathrm{D}}$ (for $k=0$, $(\mathbf{z}_0^j)_{j=1}^J$ are initialized to zero while $\mathbf{z}_0^0$ is set to $-\mathbf{F}\mathbf{y}$).
- $\mathbf{z}_{0,k+1} \in \mathbb{R}^T \times (\mathbb{R}^{2L})^J$ is equal to $(\mathbf{z}_{N_{k-1}}^j)_{0\leqslant j\leqslant J}$ given by the N_k-th layer $\mathcal{L}_{\mathrm{R}}$-$\mathcal{L}_{\mathrm{D}}$
- Θ_k is the vector of trainable parameters. The parameters are layer-dependent, so we index them by k and n.
  For layer $\mathcal{L}_{\mathrm{D}}$, the 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 $\nu_{k,n,0}$.
  For layer $\mathcal{L}_{\mathrm{R}}$, the 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 as well as the surrogates $(\tilde{\nabla}_j^{k,n})_{j=1}^J$ to the adjoints of operators $(\nabla_j^\top)_{j=1}^J$.
  To infer all these parameters, we introduce learning modules in each layer $(\mathcal{L}_{k,n}^{(\theta)})_{n=0}^{N_k-1}$ for $\theta \in \Theta_k$ (implementation discussed in Appendix C).

Schematic views of layers $\mathcal{L}_{\mathrm{D}}$ and $\mathcal{L}_{\mathrm{R}}$ can be found in Figure 9a and Figure 9b, and a composition $\mathcal{A}$ of such layers is displayed in Figure 9c.

Here we propose using $K = 7$ in (14) with $N_k = 4$ for each $k \in \{0,\cdots,K-1\}$, resulting in a total of 28 layers:

$$\mathcal{L}^0 = \cdots = \mathcal{L}^6 = (\mathcal{L}_R \circ \mathcal{L}_D)^2. \tag{16}$$

C. Incremental training strategy

For each learning module, we circumvent the issue of optimizing the initial values of their parameters using an incremental 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 all the previous layers from $(0,0)$ to $(k,n-1)$ with their past trained parameters as an initialization. This means that an increasing number of layers are trained simultaneously. In the last step, all layers are trained end-to-end.

VI. Experiments

We now describe our experimental setup. First, we illustrate the benefits of using a Cauchy-based data fidelity function. We compare the results of our RDBFB iterative approach to a simpler DBFB scheme minimizing the same cost function but with g in (8) replaced with the ℓ_2 norm.
Second, we comment on the improved performance brought by our unfolding strategy (learning of adjoints and parameters as well as the use of the ramp filter), and so U-RDBFB (Section V) is compared to the original RDBFB algorithm.

Third, we compare U-RDBFB to several state-of-the-art reconstruction methods and comment on the transfer of performance over different synthetic datasets.

A. Datasets

We used three datasets for evaluation.

1) Abdomen dataset: Our first dataset consists of 2D images obtained from 60 CT volumes of size $512 \times 512 \times 512$ from the lower lungs to the lower abdomen of 60 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).

To eliminate bias with respect to model discretization, projections were simulated for each slice of each volume in a 2D parallel geometry with a short detector of 600 bins (bin size equal to half a pixel size, i.e., 0.5 mm) and an angular density of 110 projections over 180° through to the forward projector $\mathbf{H}_{\text{P}} \in \mathbb{R}^{512^2 \times (110 \times 600)}$. The projections were then rebinned by a factor 2 (operator $\mathbf{R} \in \mathbb{R}^{300 \times 600}$). Noisy projections $\mathbf{y} = (y_t)_{t=1}^T$ are computed as

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

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 and a testing set of 500 pairs from 10 other patients.

2) Head 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/. After extracting 50 slices of size 512×512 per volume (pixel size of 0.5 mm), we performed the same processing as for the Abdomen dataset (addition of intense wires, normalization, projection, rebinning) for generating a testing set of 500 pairs of axial slices/projections $(\overline{\mathbf{x}}_{\text{P}}, \mathbf{y})$.

3) Geometrical dataset: Our third dataset was created using the toolbox TomoPhantom [56]. 500 geometrical 2D piecewise-constant phantoms were randomly generated on a 512×512 grid and normalized between 0 and 1. Again, we performed the same processing as for the Abdomen dataset for generating a testing set of 500 pairs of axial slices/projections $(\overline{\mathbf{x}}_{\text{P}}, \mathbf{y})$.

B. Training details for U-RDBFB

Let $i \in \{1, \dots, I\}$ be the index covering all $I = 2500$ 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 network is thus 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 ℓ is the loss retained for training the network. For all instances of the training set,

$$\ell(\mathbf{C}_{\text{G}}\cdot, \mathbf{C}_{\text{P}}\overline{\mathbf{x}}_{\text{P,i}}) = \frac{1}{I}\|\mathbf{C}_{\text{G}} \cdot - \mathbf{C}_{\text{P}}\overline{\mathbf{x}}_{\text{P,i}}\|^2, \tag{17}$$

corresponding to the MSE loss.

We implemented U-RDBFB following (16) in Pytorch, using a Tesla V100 32 Gb GPU. We used six epochs for training each layer $\mathcal{L}_{\text{R}}$ and ten epochs for training each layer $\mathcal{L}_{\text{D}}$; the only exception was the last layer, for which we used 20 epochs. The learning rate is decreased with a step decay by a factor of 0.99 from 10^{-2} every four epochs. The batch size for each epoch varied from 20 to 8 as the number of trained layers increased. We employed the toolbox TorchRadon [46] to include Pytorch-compatible parallel-beam tomographic operators in all architectures. Standard auto-differentiation tools can compute all necessary derivatives for backpropagation. The training procedure takes about one day and a half.

C. Competing methods

The quantitative metric used to assess the reconstruction quality of $\mathbf{C}_{\text{G}}\mathbf{x}^*_{\text{G,i}}$ 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 $\mathbf{C}_{\text{G}}\mathbf{x}^*_{\text{G,i}}$ and $\mathbf{C}_{\text{P}}\overline{\mathbf{x}}_{\text{P,i}}$.

We compare U-RDBFB with FBP, an iterative method, and four deep-learning methods that we describe hereinafter.

1) FBP: This analytical method consists of computing $\mathbf{H}_{\text{ROI}}^\top \mathbf{F}\mathbf{y}$, where $\mathbf{H}_{\text{ROI}} \in \mathbb{R}^{300^2 \times (110 \times 300)}$. As is commonly the case when applying FBP on truncated data, we extrapolated the projections prior to ramp filtering (using anti-symmetric padding).

2) RDBFB algorithm: For completeness, we perform comparisons with the iterative method proposed in Section IV. For each reweighted iteration k, we used $N_k = 10$ DBFB iterations alternating between data and regularization steps (1:1 correspondence). For an easier manual tuning of the hyperparameters, instead of using $J = 6$ as in U-RDBFB, we set $J = 1$ so that STV reduces to TV and $\alpha_{1,\ell} \equiv \alpha_1$, for all $\ell \in \{1, \cdots, L\}$. The remaining cost function parameters (ξ, κ, β) are selected by optimizing PSNR on the training set via a grid search.

3) Post-processing U-net: The third competing method is the CNN proposed in [16], [17], which is a post-processing of FBP. It relies on a trained residual U-net, with a depth of 4 levels, filters of size 32, and batch normalization to improve the stability of training.

4) Preconditioned Neumann Network (PNN): Our fourth 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

$$(\mathbf{H}^\top\mathbf{H} + \nabla r)\mathbf{x} = \mathbf{H}^\top\mathbf{y}, \tag{18}$$

which can be rewritten as

$$(\mathbf{H}^\top\mathbf{H} + \lambda\,\mathrm{Id}\,)\mathbf{x} + (\nabla r - \lambda\,\mathrm{Id}\,)\mathbf{x} = \mathbf{H}^\top\mathbf{y}. \tag{19}$$

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

$$(\mathrm{Id}\, - \lambda\mathbf{T}_\lambda + \mathbf{T}_\lambda\nabla r)\mathbf{x} = \mathbf{T}_\lambda\mathbf{H}^\top\mathbf{y}. \tag{20}$$

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 4)

$$(\lambda\mathbf{T}_\lambda - \mathbf{T}_\lambda\nabla r)^N \circ \mathbf{T}_\lambda(\mathbf{H}^\top\mathbf{y}). \tag{21}$$

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. The inner U-net has a depth of 4, the learning rate is set to 10^{-4}, and the initial value for λ is 0.01. We choose $N = 3$. One feature of PNN compared to other deep unfolding networks is that it contains skip connections.

5) ISTA-net: Our fifth competing method is ISTA-net, 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

$$\mathbf{x}_{n+1} = \mathbf{W}^\top \operatorname{soft}\left(\mathbf{W}(\mathbf{x}_n - \tau\mathbf{H}^\top(\mathbf{H}\mathbf{x}_n - \mathbf{y}), \lambda\tau\right), \tag{22}$$

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 5a). The 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 32 input and output channels; $\mathbf{A}_n$ has 1 input channel and 32 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, $\chi = 0.1$, $\mathbf{x}_0$ is the FBP reconstruction, λ and τ are initialized to 100 and 0.1 respectively.

6) 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 of their network reads

$$\mathbf{z}_{n+1} = \mathrm{CNN}(\mathbf{z}_n + \sigma\mathbf{H}\tilde{\mathbf{x}}_n; \mathbf{y}) \tag{23}$$

$$\mathbf{x}_{n+1} = \mathrm{CNN}(\mathbf{x}_n - \tau\mathbf{H}^\top\mathbf{z}_{n+1}) \tag{24}$$

$$\tilde{\mathbf{x}}_{n+1} = \mathbf{x}_{n+1} + \gamma(\mathbf{x}_{n+1} - \mathbf{x}_n). \tag{25}$$

The CNNs act both in the image and projection domains. Furthermore, buffers of previous iterates of size $N_p \in \mathbb{N}$ 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. We used 9 layers, $N_d = N_p = 3$, and 32 filters in the convolutional layers. This network is illustrated in Figure 5b.

The competing networks were also trained with the MSE loss (using (17) for unfolding networks and a regularization term for ISTA-net weighted by χ) in a standard end-to-end manner. The number of epochs was chosen such that the training and testing losses have stabilized. Note that codes are publicly available for these networks. We re-implemented them in Pytorch and kept the setting of the parameters advocated by the authors, except for PNN, for which we reduced the number of layers to 3 to obtain a stable behavior for training. The total number of parameters of each network is reported in Table I.

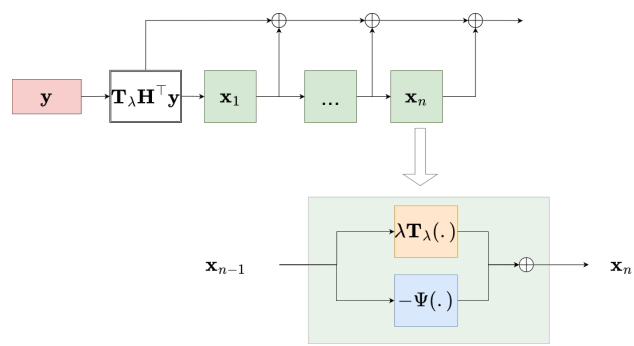
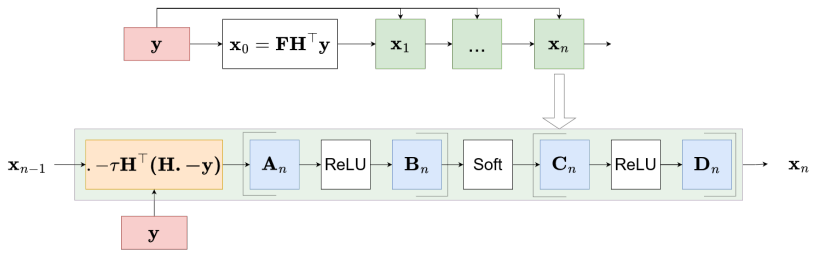


Fig. 4: Architecture of PNN [52]: The network maps a linear function of the measurements $\mathbf{T}_\lambda\mathbf{H}^\top\mathbf{y}$ to a reconstruction $\mathbf{x}_n$ by successive applications of an operator of the form $\lambda\mathbf{T}_\lambda - \Psi$, while summing the intermediate outputs of 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.

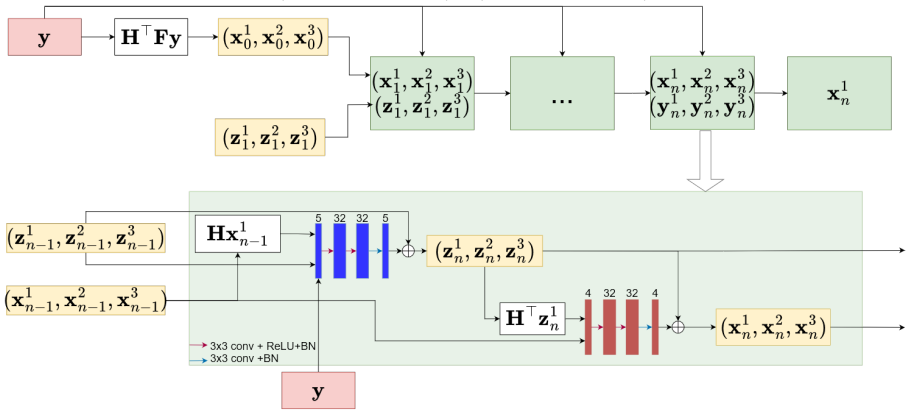
VII. Results

A. Assessing the benefits of the Cauchy fidelity term

Figure 6c shows the full reconstructed images on grid G of size 400×400 obtained using the DBFB algorithm with a quadratic fidelity term and the reweighed DBFB algorithm with a Cauchy fidelity on a test instance of the Abdomen dataset (shown in Figure 6a). Since the two data fidelity terms can be put into our optimization framework Alg. 1-Alg. 2, the comparison is straightforward. Figure 6d shows



(a) 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.



(b) 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.

Fig. 5: Two competing unfolded proximal algorithms

the corresponding ROIs as well as the FBP reconstruction. The full image contains two intense objects out of the ROI and at the border of the reconstruction grid. In the solution obtained using the quadratic data fidelity term, the reduction of sub-sampling streaks is selective; only the streaks originating from objects within G have been eliminated in the ROI. When trading the quadratic term with a Cauchy term, as we proposed, the intensity of these streaks is reduced. This artifact reduction translates into an improvement of the PSNR as shown in Figure 6b. Figure 6e shows the ROIs obtained using the same reconstruction methods and grid size when increasing the number of projections from 110 to 600. The images now look identical and close to the ground truth. This observation highlights that, for relatively 'clean' data (no modeling of beam hardening and scattering), the benefits of using a Cauchy fidelity over a quadratic fidelity emerge when data is sub-sampled.

We have shown that by using a regularized cost function with a Cauchy fidelity term and, thus, a more complex optimization framework, we can successfully reconstruct truncated data on a short grid and that the reconstruction is at least as good as the one obtained with quadratic fidelity and even better when the data are sub-sampled.

B. Comparing iterative RDBFB algorithm with U-RDBFB

Figure 7a shows the evolution of the PSNR along the iterations when inserting the ramp filter in the reweighted DBFB algorithm, re-tuning the regularization strength, and still using the same data. The PSNR stagnates around 300 iterations with the ramp filter while it stagnates around 1250 iterations without it (see Figure 6b). Thus applying the ramp filter on the reprojection error before backprojection can empirically accelerate convergence without degrading the solution (reconstructed ROI displayed in Figure 7c) in an early stopping scenario. This motivates our translation of a data iteration of DBFB to a data layer of U-RDBFB which embeds the ramp filter. It also provides empirical evidence that performance can be optimized by introducing mismatched adjoints without learning.

U-RDBFB also includes learned parameters, especially adjoints to the STV operators. It performs a total of 28 RDBFB iterations. Figure 7c compares the reconstruction ROI obtained with U-RDBFB, 500 and 28 iterations of the reweighed DBFB algorithm with the ramp filter. We see that after 28 iterations of the reweighed DBFB algorithm, there is a local offset near the intense object, and some streaks remain. On the contrary, the image obtained with U-RDBFB does not contain

	U-RDBFB	U-net	PNN	PD-net	ISTA-net
$\lvert\Theta\rvert$	2.3169×10^4	1.9278×10^6	1.9278×10^6	2.5470×10^5	1.7109×10^5

TABLE I: Number of learnable parameters (Θ)

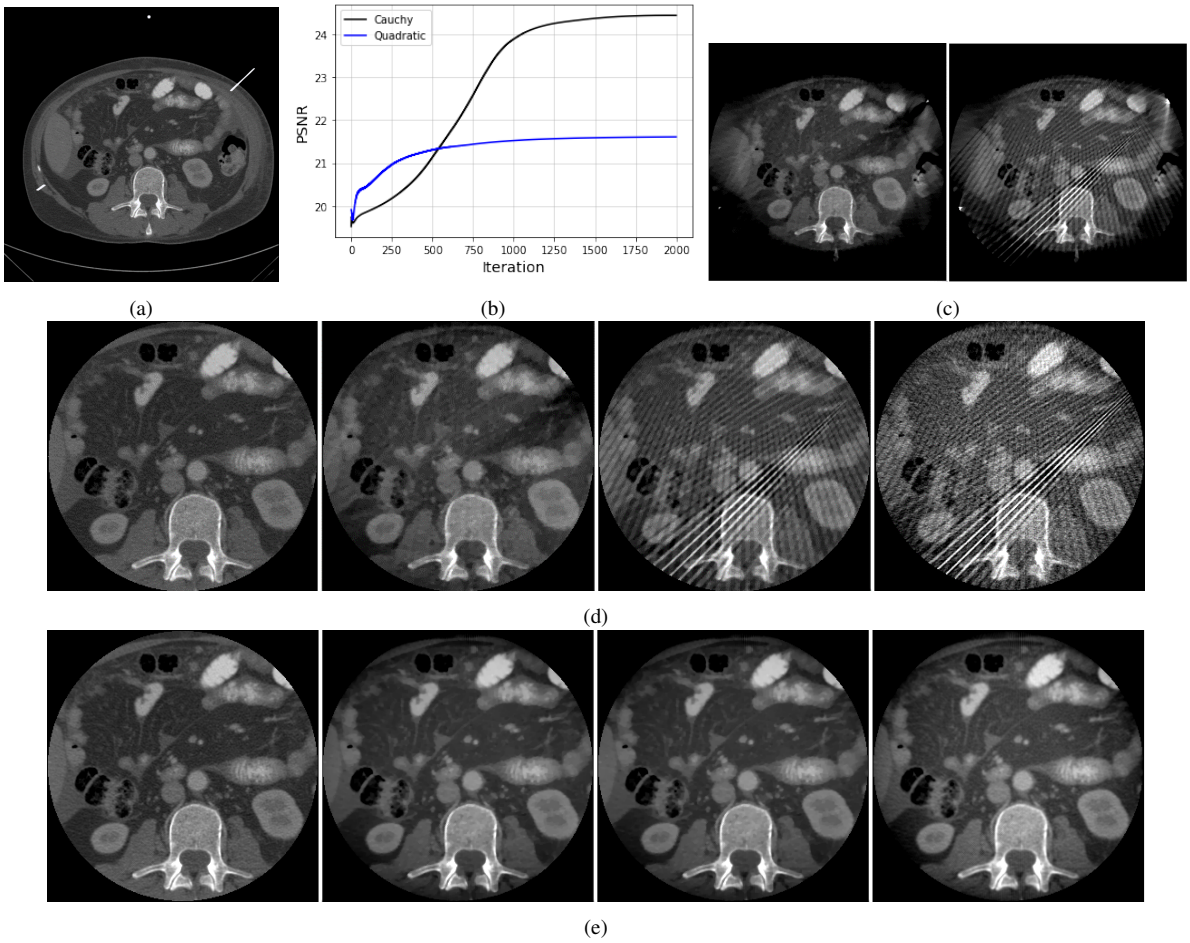


Fig. 6: (a) Ground truth $\overline{\mathbf{x}}_{\mathrm{P}}$. (b) Evolution of the PSNR along iterations using a Cauchy or quadratic data term for 110 projections. (c) Reconstructed extended ROIs using 110 projections, a Cauchy data fidelity, and a quadratic data fidelity. (d) Reconstructed ROIs using 110 projections. (e) Reconstructed ROIs using 600 projections. From left to right: Ground truth, reweighted DBFB with Cauchy fidelity, DBFB with quadratic fidelity, FBP.

these artifacts. It is similar to the ground truth and slightly smoother than the reconstruction after 500 iterations of the reweighed DBFB algorithm but with approximately 18 times fewer iterations.

C. Comparing U-RDBFB with deep learning methods on the Abdomen dataset

Table II reports the performance of U-RDBFB compared to U-net and other deep unfolding networks on the testing set of the Abdomen dataset and Figures 8a-8b display the training and testing losses as a function of the number of epochs for all these networks. U-RDBFB performs, on average, better than the other unfolding networks (PNN, PD-net, and ISTA-net) and U-net for all considered metrics. We note that the peaks in the training and testing losses associated with U-RDBFB correspond to the addition of a new data layer during incremental training.

Figure 10c illustrates the reconstructed ROIs for four examples from the test set of the Abdomen dataset. The FBP reconstruction is also displayed as it is also the input of U-net. The figure confirms that U-RDBFB reduces streaks more effectively than the other unfolding networks. At first sight, the images produced by U-net have fewer artifacts than most deep unfolding networks. However, in the second-row and fourth-row images, U-net introduces an artificial dark background. 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.

Figure 10b shows the complete reconstruction on grid G for all unfolding networks. In all cases, since the training loss acts on the ROI only, the exterior is always poorly reconstructed (with ISTA-net, it is very sparse).

D. Changing the testing set

We now evaluate the generalization ability of the methods trained on the Abdomen dataset, using examples from the Head and Geometrical datasets. In Table III and Table IV, we report the performance of the trained networks when tested on the Head and the Geometrical datasets. U-RDBFB outperforms the competing unfolding networks for both datasets. ISTA-net is second-best on the Head dataset, and the iterative algorithm ranks second-best on the Geometrical dataset.

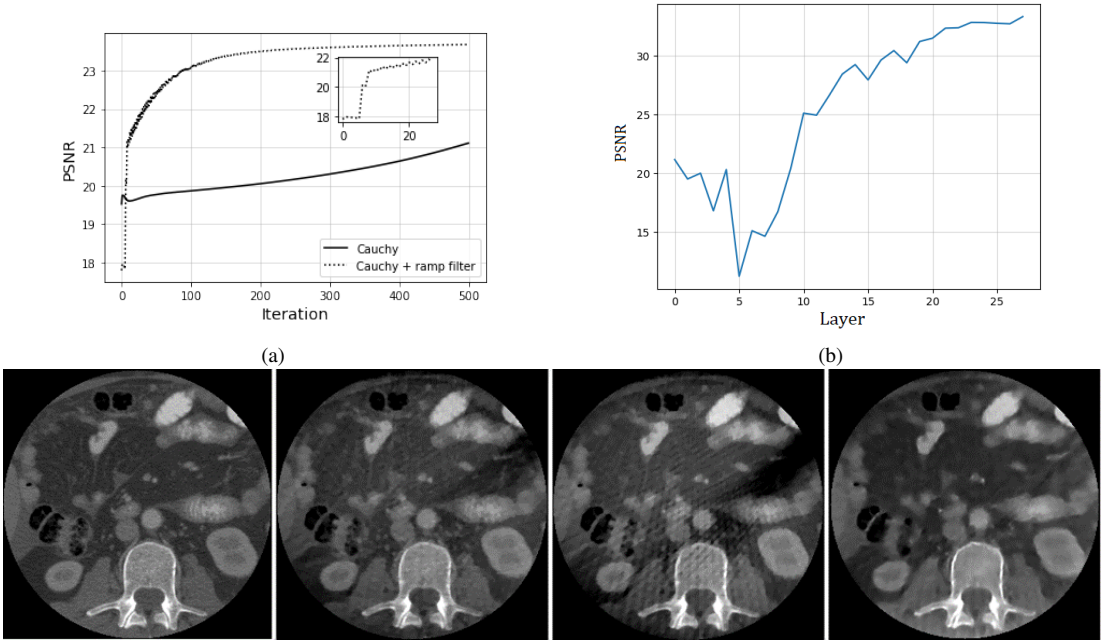


Fig. 7: (a) Evolution of the PSNR along iterations using a Cauchy fidelity term with and without the ramp filter for the example of Fig 6a. (b) Evolution of the PSNR along layers in U-RDBFB for the example of Fig 6a. (c) Reconstructed ROIs. From left to right: Ground truth, Cauchy with ramp filter (500 iterations of the modified reweighted DBFB), Cauchy with ramp filter (28 iterations of the modified reweighted DBFB), U-RDBFB.

Metrics	U-RDBFB	U-net	PPN	PD-net	ISTA-net	RDBFB
PSNR	**38.7**	38.6	**38.7**	36.5	38.2	33.9
SSIM	**0.981**	0.972	0.975	0.956	0.975	0.903
MAE ($\times 10^{-3}$)	**3.54**	4.80	4.00	5.53	4.95	8.41
PieApp	**0.389**	0.502	0.611	0.599	0.613	0.653

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

Metrics	U-RDBFB	U-net	PPN	PD-net	ISTA-net	RDBFB
PSNR	**31.7**	16.8	18.5	17.6	23.1	21.1
SSIM	**0.979**	0.902	0.838	0.765	0.957	0.908
MAE ($\times 10^{-3}$)	**2.17**	7.86	4.87	6.47	3.71	6.25
PieApp	**0.276**	0.885	0.962	0.984	0.650	0.476

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

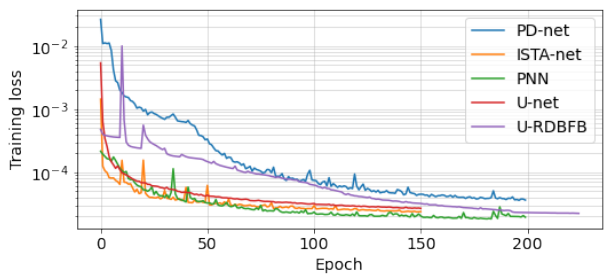
Metrics	U-RDBFB	U-net	PPN	PD-net	ISTA-net	RDBFB
PSNR	**27.3**	25.1	27.0	25.2	25.8	26.1
SSIM	**0.856**	0.653	0.733	0.736	0.816	0.848
MAE ($\times 10^{-3}$)	16.6	44.1	52.5	25.4	18.4	**16.5**
PieApp	**0.267**	1.170	1.265	1.324	0.897	0.158

TABLE IV: Quantitative assessment of the reconstructed ROIs. Mean values computed over the testing set of the Geometrical dataset.

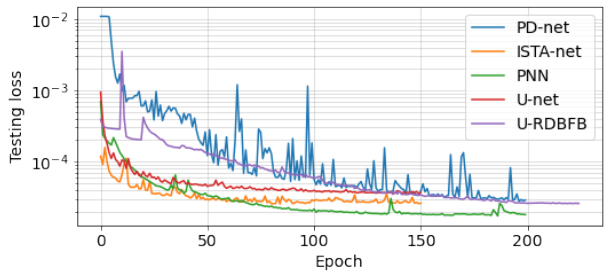
The four reconstructed images displayed in Figures 11a-11b confirm this trend. For the Head dataset, U-net performs poorly relative to U-RDBFB on all metrics except for SSIM, where the two methods are rather close. One explanation is that it introduces an offset in some images while limiting the streaks (cf second head image). However, when applied to the geometrical images, it yields unwanted background patterns that strongly degrade our metrics. Offsets and background artifacts are also visible with most unfolding networks, especially PNN and PD-net, except for U-RDBFB. We still note that the head images and the first geometrical image obtained with U-RDBFB have a slightly patchy look, often characteristic of TV regularization. This suggests that U-RDBFB retains the characteristics of the original optimization problem, avoiding the generation of unexpected content as is possible with U-net.

VIII. Discussion

The results show that U-RDBFB outperforms its iterative counterpart as measured in PSNR, SSIM, and PieApp. It not only presents results similar to its iterative counterpart RDBFB in terms of streaks reduction, but it also recovers image details in a much lower number of iterations. We note that U-RDBFB leads to smoother images: the noise is reduced, but the resolution of the image is somehow decreased. This



(a)



(b)

Fig. 8: (a) MSE on the training set as a function of the epoch number. (b) MSE on the testing set as a function of the epoch number.

could be explained by the fact that U-RDBFB is designed to minimize the ℓ_2 norm of the error with respect to the noiseless ground truth, using very few regularization layers; it tends to selectively smooth some parts of the image to remove remaining artifacts.

All metrics agree that U-RDBFB improves upon learned post-processing U-net and other unfolding networks for our Abdomen dataset. U-net was often associated with a high PSNR compared to the other reconstructions, but this was not always reflected by the PieApp metric. This may be explained by the learned post-processing being limited by the information content of the FBP input, while the unfolding networks act directly with the information content of the data, which is greater than that of the FBP.

The computation time for U-RDBFB was about 200 ms in GPU for a 400×400 reconstruction grid. This is much faster than the iterative reconstruction, which, in our case, requires around 180 s after the regularization parameters have been selected, but slower than other deep learning methods (38 ms for U-net, 68 ms for PNN, 74 ms for ISTA-net, 85 ms for PD-net).

U-RDBFB contains fewer learnable parameters than all the other networks. Thanks to our incremental strategy, training U-RDBFB was also found to be easier than other unfolding networks, such as PNN, whose stability highly depends on the initialization for parameter λ and the learning rate. Optimizing the architecture and, more precisely, the number of parameters of a neural network is key to transferring its performance to out-of-distribution examples, as shown on the Head and Geometrical datasets. Generally, deep unfolding networks are introduced mainly to ensure data consistency through $\mathbf{H}$ and embed a fast optimization scheme for fast inference. Our results suggest that including additional a priori knowledge can further boost the performance of deep-learning-based techniques. Note that the structure of U-RDBFB (K, N_k and distribution of $\mathcal{L}_D$ and $\mathcal{L}_R$) was not extensively fine-tuned. Our results illustrate that the most straightforward choices work well in our context of ROI imaging from angularly sub-sampled data.

Our results also hinted that even without learning, trading a quadratic fidelity for a Cauchy fidelity, discarding the data incompatible with the a priori support of the object, and including some preconditioning through the ramp filter are still of interest to improve reconstruction and reduce the number of iterations.

IX. Conclusion

In this paper, we introduced an iterative reweighted algorithm where each inner optimization problem is solved using dual block coordinate forward-backward iterations, and we proposed an unfolded version of it, yielding a neural network for ROI reconstruction from a few measurements. These methods include a convex surrogate to a Cauchy data fidelity and a TV-based regularization to limit sub-sampling streaks originating from inside and outside the reconstruction grid. Our experiments demonstrated the benefits of the unfolded strategy over the original iterative algorithm. By balancing the capacity of the network and the use of prior knowledge, our architecture displayed high generalization ability compared to various neural networks, including U-net and other deep unfolding networks. Future work will investigate loss functions that better preserve image resolution and further optimization of the structure of our network for application to other geometries (e.g., cone-beam) using real CT data.

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

$$\tilde{\phi}(\zeta, \overline{\zeta}) = \phi(\overline{\zeta}) + \beta \frac{(\zeta - \overline{\zeta})\overline{\zeta}}{1 + (\overline{\zeta}/\kappa)^2} + \frac{\beta}{2} \frac{(\zeta - \overline{\zeta})^2}{(1 + (\overline{\zeta}/\kappa)^2)}, \quad (26)$$

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

$$(\forall \zeta \in \mathbb{R}) \;\; \tilde{\phi}(\zeta, \overline{\zeta}) \geqslant \phi(\zeta) \quad \text{and} \quad \tilde{\phi}(\overline{\zeta}, \overline{\zeta}) = \phi(\overline{\zeta}). \quad (27)$$

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

$$(\forall \mathbf{x} \in \mathbb{R}^L) \quad \tilde{f}(\mathbf{x}, \overline{\mathbf{x}}) = \tilde{g}(\mathbf{H}\mathbf{x} - \mathbf{y}; \mathbf{H}\overline{\mathbf{x}} - \mathbf{y}), \quad (28)$$

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

$$(\forall(\mathbf{x},\overline{\mathbf{x}}) \in (\mathbb{R}^L)^2) \quad Q(\mathbf{x},\overline{\mathbf{x}}) = \tilde{f}(\mathbf{x},\overline{\mathbf{x}}) + r(\mathbf{x}). \tag{29}$$

Appendix B
Implementation of DBFB

Step **(D)** involves the calculation of the proximity operator $\mathrm{prox}_{\gamma_n^{-1}\sigma h_0(\cdot,\mathbf{B}_0\overline{\mathbf{x}})}$, which has a closed-form [29, Example 24.2], for $(\forall(\mathbf{z},\overline{\mathbf{z}}) \in (\mathbb{R}^T)^2)$,

$$\begin{aligned}
&\mathrm{prox}_{\gamma_n^{-1}\sigma h_0(\cdot,\mathbf{B}_0\overline{\mathbf{z}})}(\mathbf{z}) = \mathrm{prox}_{\gamma_n^{-1}\sigma\tilde{g}(\cdot-\mathbf{y},\overline{\mathbf{z}}-\mathbf{y})}(\mathbf{z}),\\
&= \mathbf{y} + \mathrm{prox}_{\gamma_n^{-1}\sigma\tilde{g}(\cdot,\overline{\mathbf{z}}-\mathbf{y})}(\mathbf{z}-\mathbf{y})\\
&= \left(\mathbf{y}_t + \frac{\mathbf{z}_t-\mathbf{y}_t}{1+\beta\gamma_n^{-1}\sigma\big(1+(\overline{\mathbf{z}}_t-\mathbf{y}_t)^2/\kappa^2\big)^{-1}}\right)_{t=1}^{T}.
\end{aligned} \tag{30}$$

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

$$\mathrm{prox}_{\gamma r_j}(\mathbf{z}) = \left(\max\left\{0, 1-\frac{\gamma\alpha_{j,l}}{\|\mathbf{z}_\ell\|_2}\right\}\mathbf{z}_\ell\right)_{\ell=1}^{L}, \tag{31}$$

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

Appendix C
Parameter learning

We discuss our choices for $(\mathcal{L}_{k,n}^{(\theta)})_{n=0}^{N_k-1}$.

- step size for $\mathcal{L}_{\mathrm{D}}$: $\nu_{k,n,0} = \mathcal{L}_{k,n}^{(\nu)} = \mathrm{softplus}(a_{k,n})$ where $a_{k,n}$ is a learnable real-valued parameter.
- Parameters of the Cauchy function for $\mathcal{L}_{\mathrm{D}}$:
  - $\kappa_{k,n} = \mathcal{L}_{k,n}^{(\kappa)} = W_\kappa\,\mathrm{softplus}(c_{k,n})$ 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., $\mathbf{H}^\top(\mathbf{F}\mathbf{x}_n-\mathbf{y})$. 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.
  - $\beta_{k,n} = \mathcal{L}_{k,n}^{(\beta)} = W_\beta\,\mathrm{softplus}(d_{k,n})$.
- 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 $\mathcal{L}_{\mathrm{R}}$: $\xi_{k,n} = \mathcal{L}_{k,n}^{(\xi)} = \mathrm{softplus}(e_{k,n})$ where $e_{k,n}$ is learned.
- step size for $\mathcal{L}_{\mathrm{R}}$: For every $j \in \{1,\ldots,J\}$, $\nu_{k,n,j} = \mathcal{L}_{k,n}^{(\nu_j)} = W_\nu\,\mathrm{softplus}(b_{k,n,j})$.
- Parameters of the STV regularization for $\mathcal{L}_{\mathrm{R}}$: For every $j \in \{1,\ldots,J\}$,

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

  where $A_{k,n}$ is a grouped convolution of 7 groups with size 3×3 kernels and $J=7$ channels and $B_{k,n}$ is a grouped convolution of 14 groups with size 5×5 kernels and 14 channels.

In $\mathcal{L}_D$, initial values of $a_{k,n}$, $d_{k,n}$ are set to 1. In $\mathcal{L}_R$, initial values for $b_{k,n,j}$, $e_{k,n}$ are 1. Normalization scalars W_κ, W_ν, W_β and W_α are set to 10^{-5}, 10, 10 and 0.05 respectively. Finally, in each layer $\mathcal{L}_R$, operations $(\tilde{\mathbf{L}}_j^{k,n})_{j=1}^{J}$ are learned; they are convolutions with a kernel of the same size as for $(\mathbf{L}_j^{k,n})_{j=1}^{J}$.

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*, 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.

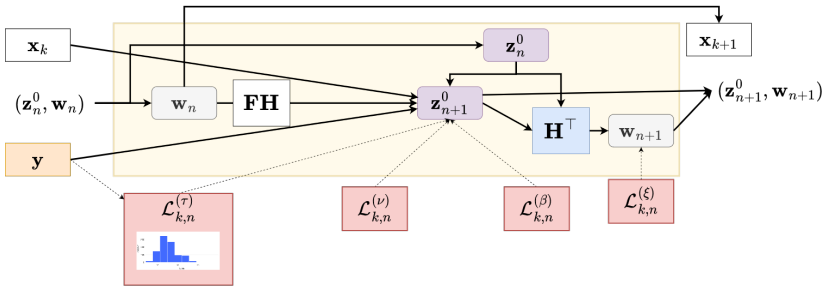
[54] J. Cheng, H. Wang, L. Ying, and D. Liang. Model learning: Primal dual networks for fast MR imaging. Preprint, 2019. Available: https://arxiv.org/abs/1908.02426.

[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.

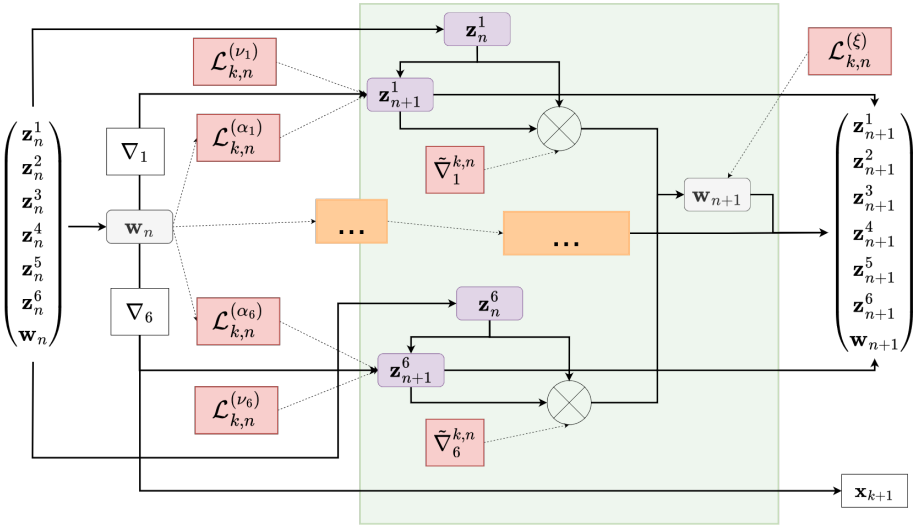
[56] D. Kazantsev, V. Pickalov, D. Nagella, E. Pasca, P. J. Withers. TomoPhantom, a software package to generate 2D–4D analytical phantoms for CT image reconstruction algorithm benchmarks. *SoftwareX*, vol. 7, pp. 150–155, 2018.

[57] W. Yu, C. Wang, M. Huang. Edge-preserving reconstruction from sparse projections of limited-angle computed tomography using ℓ_0-regularized gradient prior. *Review of Scientific Instruments*, vol. 88, pp. 043703, 2017.

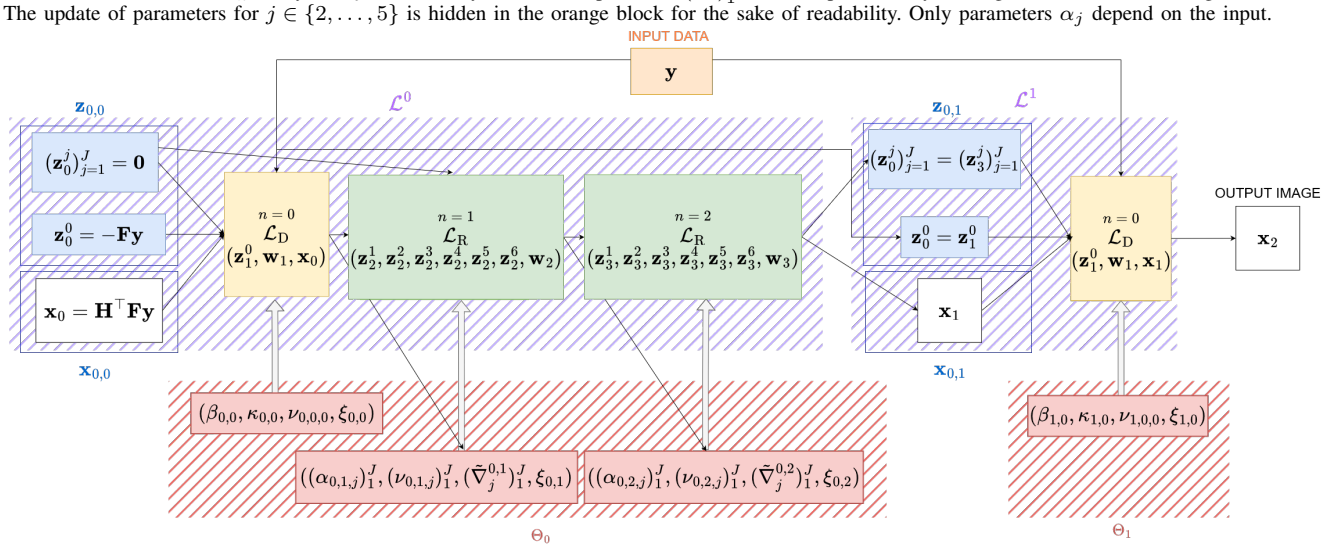
[58] T. Wang, K. H. Nakamoto, H Zhang and H. Liu, Reweighted Anisotropic Total Variation Minimization for Limited-Angle CT Reconstruction. *IEEE Transactions on Nuclear Science*, vol. 64, pp. 2742–2760, 2017



(a) Schematic view of a N_k-th layer $\mathcal{L}_D$ (12). The layer relies on $\overline{\mathbf{x}} = \mathbf{x}_k$, the k-th reweighted iterate to generate the next reweighted iterate $\mathbf{x}_{k+1}$. 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.

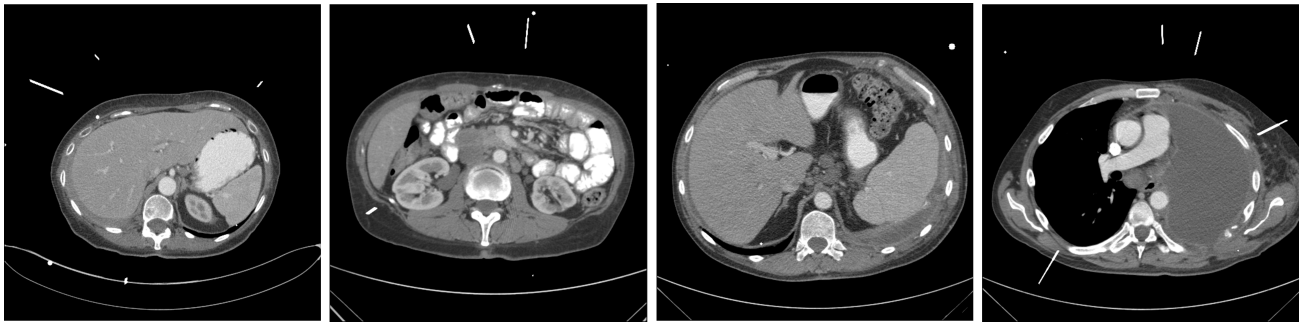


(b) Schematic view of a N_k-th layer $\mathcal{L}_R$ (13). The layer takes as inputs $\mathbf{w}_n$, $(\mathbf{z}_n^j)_1^J$ from the previous layer and generates the next reweighted iterate $\mathbf{x}_{k+1}$. The update of parameters for $j \in \{2, \dots, 5\}$ is hidden in the orange block for the sake of readability. Only parameters α_j depend on the input.

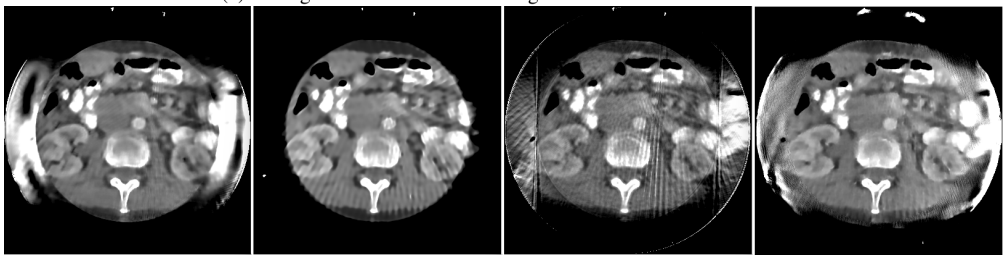


(c) U-RDBFB in the case where $\mathcal{A} = \mathcal{L}^1 \circ \mathcal{L}_2^0$ where $\mathcal{L}^0 = (\mathcal{L}_R)^2 \circ \mathcal{L}_D$ and $\mathcal{L}_2^1 = \mathcal{L}_D$ (i.e., $K = 2$, $N_0 = 3$, $N_1 = 1$). Red blocks represent the hidden structures to infer all the parameters $\theta \in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

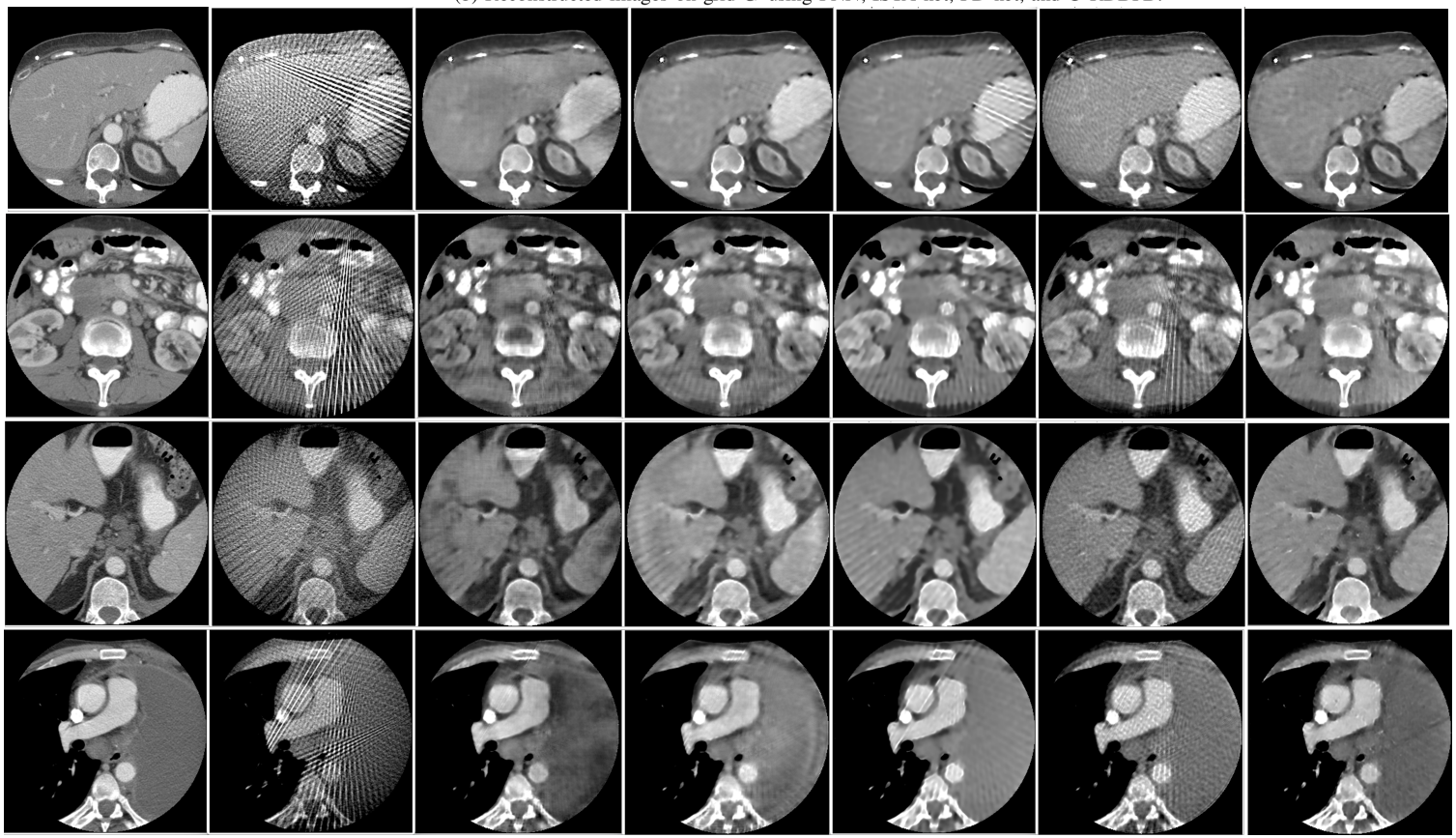
Fig. 9: Architecture of U-RDBFB



(a) Four ground truths of the testing set of the Abdomen dataset

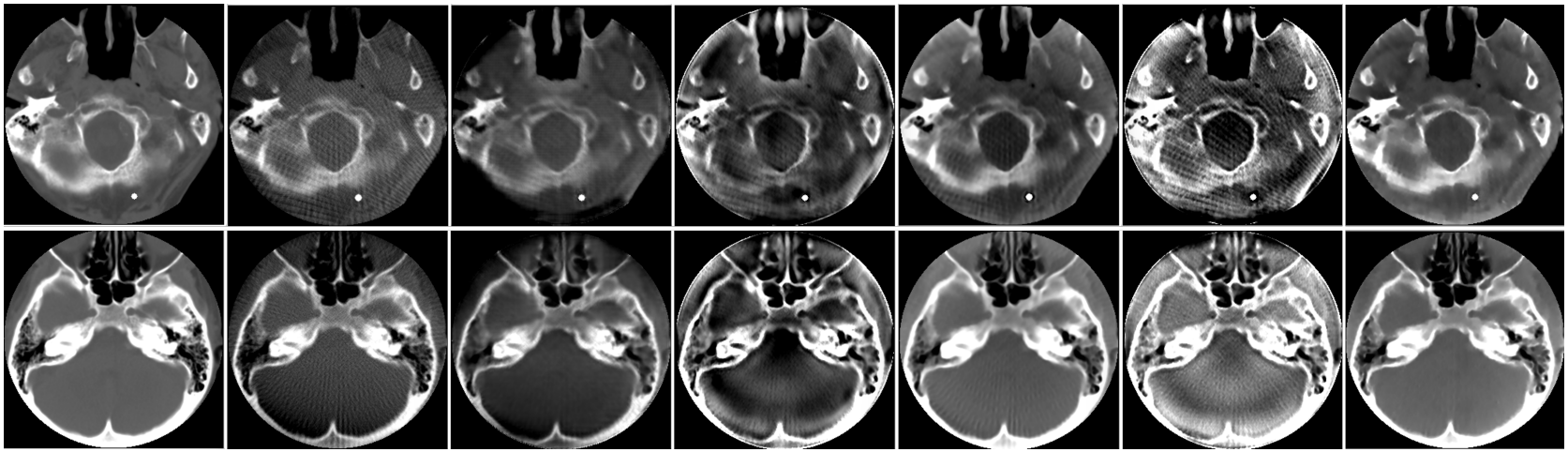


(b) Reconstructed images on grid G using PNN, ISTA-net, PD-net, and U-RDBFB.

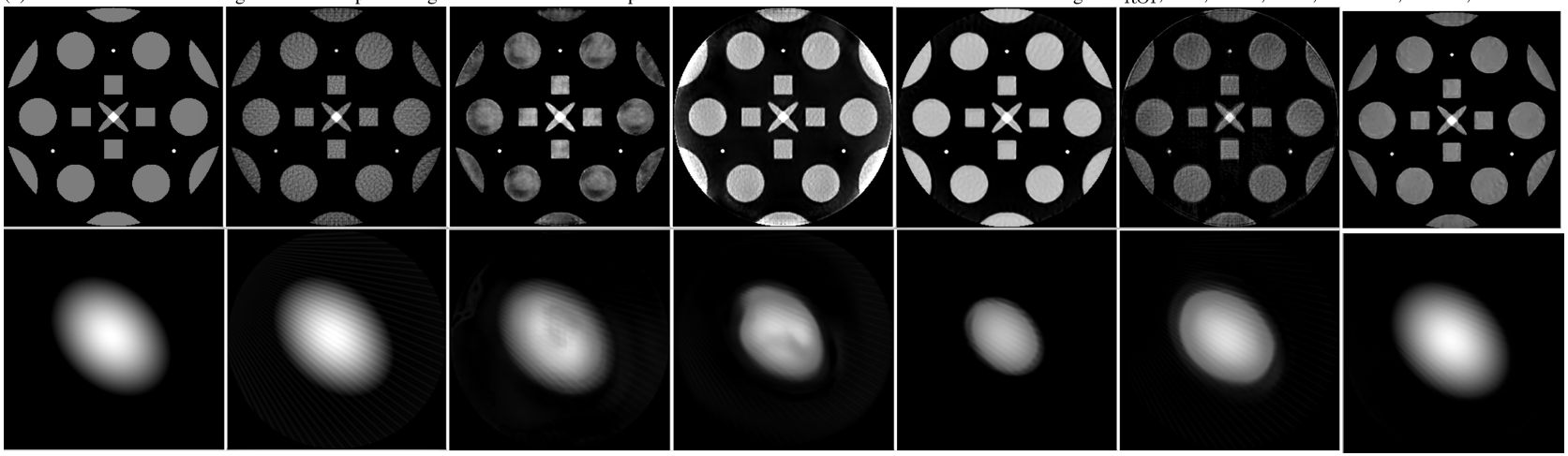


(c) Reconstructed ROIs using different deep learning methods on four examples in the test set of the Abdomen dataset. From left to right: $\overline{\mathbf{x}}_{\mathrm{ROI}}$, FBP, U-net, PNN, ISTA-net, PD-net, U-RDBFB

Fig. 10: Evaluation on the Abdomen dataset.



(a) Reconstructed ROIs using different deep learning methods on four examples in the test set of the Head dataset. From left to right: $\overline{\mathbf{x}}_{\mathrm{ROI}}$, FBP, U-net, PNN, ISTA-net, PD-net, U-RDBFB



(b) Reconstructed ROIs using different deep learning methods on four examples in the test set of the Geometrical dataset. From left to right: $\overline{\mathbf{x}}_{\mathrm{ROI}}$, FBP, U-net, PNN, ISTA-net, PD-net, U-RDBFB

Fig. 11: Evaluation on the Head and Geometrical datasets.