Learned Primal-dual Reconstruction

Jonas Adler*† Ozan Öktem*
*Department of Mathematics, KTH - Royal Institute of Technology
†Elekta AB, Box 7593, SE-103 93 Stockholm, Sweden
Email: {jonasadl, ozan}@kth.se

Abstract—We propose a Learned Primal-Dual algorithm for tomographic reconstruction. The algorithm includes the (possibly non-linear) forward operator in a deep neural network inspired by unrolled proximal primal-dual optimization methods, but where the proximal operators have been replaced with convolutional neural networks. The algorithm is trained end-to-end, working directly from raw measured data and does not depend on any initial reconstruction such as Filtered Back-Projection (FBP).

We evaluate the algorithm on low dose Computed Tomography (CT) reconstruction using both analytic and human phantoms against classical reconstruction given by FBP and total variation (TV) regularized reconstruction as well as deep learning based post-processing of a FBP reconstruction.

For the analytic data we demonstrate Peak Signal to Noise Ratio (PSNR) improvements of >10 dB when compared to both TV reconstruction and learned post-processing. For the human phantom we demonstrate a 6.6 dB improvement compared to TV and a 2.2 dB improvement as compared to learned post-processing. The proposed algorithm also improves upon the compared algorithms with respect to the structural similarity index (SSIM) and the evaluation time is ≈ 600 ms for a 512 × 512 pixel dataset.

Index Terms—Inverse problems, Tomography, Deep learning, Primal-Dual, Optimization

I. INTRODUCTION

Tomographic reconstruction such as Computed Tomography (CT) and Magnetic Resonance Imaging (MRI) is by now an established diagnostic tool in medicine. In tomographic reconstruction, the patients interior is inferred from indirect data such as x-ray projection images (CT) or k-space coefficients (MRI). These imaging problems and several others can be phrased in the general framework of inverse problems.

Inverse problems refer to problems where one seeks to reconstruct parameters characterizing the system under investigation from indirect observations. Such problems arise in several areas of science and engineering. Mathematically, an inverse problem can be formulated as reconstructing (estimating) a signal \( f_{\text{true}} \in X \) from data \( g \in Y \) where

\[
g = \mathcal{T}(f_{\text{true}}) + \delta g. \tag{1}
\]

In the above, \( X \) and \( Y \) are typically Hilbert Spaces, \( \mathcal{T} : X \rightarrow Y \) (forward operator) models how a given signal gives rise to data in absence of noise, and \( \delta g \in Y \) is a single sample of a \( Y \)-valued random variable that represents the noise component of data.

A. Variational regularization

A common approach in solving (1) is to maximize the likelihood of the signal, or equivalently minimizing the log-likelihood

\[
\min_{f \in X} \mathcal{L}(\mathcal{T}(f), g), \tag{2}
\]

where \( \mathcal{L} : Y \times Y \rightarrow \mathbb{R} \) is the data log-likelihood [1].

This minimization is for typical choices of \( \mathcal{T} \) ill-posed, that is, a solution (if it exists) is unstable with respect to the data \( g \) in the sense that small changes to data results in large changes to a reconstruction. Hence, a maximum likelihood solution typically leads to over-fitting against data.

Variational regularization avoids over-fitting by introducing a functional \( \mathcal{S} : X \rightarrow \mathbb{R} \) (regularization functional) that encodes a priori information about the true (unknown) \( f_{\text{true}} \) and penalizes unlikely solutions [2], [3]. Hence, instead of minimizing only the data log-likelihood, one now seeks to minimize a regularized objective functional by solving

\[
\min_{f \in X} [\mathcal{L}(\mathcal{T}(f), g) + \lambda \mathcal{S}(f)] \quad \text{for a fixed } \lambda \geq 0. \tag{3}
\]

In the above, \( \lambda \) (regularization parameter) governs the influence of the a priori knowledge encoded by the regularization functional against the need to fit data.

B. Optimization schemes

In imaging, the minimization in (3) is a large scale optimization problem which traditionally has addressed using gradient based methods such as gradient descent or its extensions to higher order derivatives, e.g. quasi-Newton or Newton methods. However, given that many regularizers of interest result in a non-differentiable object functional, gradient based methods are not in general directly applicable. This difficulty can be circumvented by solving a smooth approximation to the objective functional, but this introduces additional parameters and gives non-exact solutions.

Proximal methods have been developed in order to directly work with non-smooth objective functionals. Here, a proximal step replaces the gradient step. The simplest example of such an algorithm is the proximal point algorithm for minimizing an objective functional of the form \( \mathcal{G} : X \rightarrow \mathbb{R} \). It can be seen as the proximal equivalent of the gradient descent scheme and is given by

\[
f_{i+1} = \text{prox}_{\tau \mathcal{G}}(f_i) \tag{4}
\]
where $\tau \in \mathbb{R}^+$ is a step size and the proximal operator is defined by

$$
\text{prox}_{\tau \mathcal{F}}(f) = \arg \min_{f' \in X} \left[ \mathcal{G}(f') + \frac{1}{2\tau} \| f' - f \|_X^2 \right]
$$

(5)

While this algorithm could, in theory, be applied to solve (3) it is rarely used directly since (5) does not have a closed form solution. Proximal primal-dual schemes offer a work around. In these schemes, an axillary dual variable in the range of the operator is introduced and the primal ($f \in X$) and dual variables are updated in an alternating manner.

One well known primal-dual scheme is the Primal Dual Hybrid Gradient (PDHG) algorithm, also known as the Chambolle-Pock algorithm [4], with a recent extension to non-linear forward operators [5]. The algorithm (algorithm 1) is adapted for minimization problems with the following structure:

$$
\min_{f \in X} \left[ \mathcal{F}(\mathcal{K}(f)) + \mathcal{G}(f) \right]
$$

(6)

where $\mathcal{K}: X \to U$ is a (possibly non-linear) operator, $U$ is a Hilbert space and $\mathcal{F}: U \to \mathbb{R}$ and $\mathcal{G}: X \to \mathbb{R}$ are functionals on the dual/primal spaces. Note that (3) is a special case of this if we set $\mathcal{F} = \mathcal{L}(-;g)$, $\mathcal{K} = T$ and $\mathcal{G} = S$.

**Algorithm 1 Non-linear Primal-Dual Hybrid Gradient**

1. Given: $\sigma, \tau > 0$ s.t. $\sigma \tau \| \mathcal{K} \|_2^2 < 1$, $\theta \in [0,1]$ and $f_0 \in X$, $h_0 \in U$.
2. for $i = 1, \ldots$ do
3. $h_{i+1} \leftarrow \text{prox}_{\sigma \mathcal{F}}(h_i + \sigma \mathcal{K}(f_i))$
4. $f_{i+1} \leftarrow \text{prox}_{\tau \mathcal{G}}(f_i - \tau \partial \mathcal{K}(f_i))^*(h_{i+1})$
5. $f_{i+1} \leftarrow f_{i+1} + \theta (f_{i+1} - f_i)$

In the above, $\mathcal{F}^*$ is the Fenchel conjugate of $\mathcal{F}$, $h \in U$ is the dual variable and $[\partial \mathcal{K}(f_i)]^*: U \to X$ is the adjoint of the (Fréchet) derivative of $\mathcal{K}$ in point $f_i$.

**a) Example: total variation (TV) regularized CT**

In CT, the forward operator $\mathcal{T}$ is given by the ray-transform $\mathcal{P}: X \to Y$, which integrates the signal over a set of lines $\mathcal{M}$ given by the acquisition geometry. Hence, elements in $Y$ are functions on lines $\mathcal{P}(f) = \int_{\ell} f(x)dx$ for $\ell \in \mathcal{M}$.

and the adjoint of the derivative is given by back-projection.

A typical example of variational regularization in imaging is TV regularization, which applies to signals that are represented by scalar functions of bounded variation. The corresponding regularization functional is then given as the 1-norm of the gradient magnitude, i.e., $\mathcal{S}(f) := \| \nabla f \|_1$, $\nabla : X \to X^d$, $d$ is the dimension of the space.

The PDHG method can be used to solve the TV regularized CT optimization problem

$$
\min_{f \in X} \| \mathcal{P} f - g \|_2^2 + \lambda \| \nabla f \|_1
$$

by selecting

$\mathcal{K}: X \to Y \times X^d$ as $\mathcal{K}(f) := [\mathcal{P}(f), \nabla f]$,

$\mathcal{F}(h^{(1)}, h^{(2)}) := \| h^{(1)} - g \|_2^2 + \| h^{(2)} \|_1$ and $\mathcal{G}(f) := 0$.

**II. MACHINE LEARNING IN INVERSE PROBLEMS**

Machine learning is widely used for non-linear function approximation under weak assumptions and has recently emerged as the state of the art for several image processing tasks such as classification and segmentation. Applied to the inverse problem in (1), it can be phrased as the problem of finding a (non-linear) mapping $\mathcal{T}_\Theta^\dagger: Y \to X$ satisfying the following pseudo-inverse property:

$$
\mathcal{T}^\dagger_\Theta(g) \approx f_{\text{true}} \quad \text{whenever data } g \text{ is related to } f_{\text{true}} \text{ as in (1)}.
$$

A key element in machine learning approaches is to parametrize the set of such pseudo-inverse operators by a parameter $\Theta \in Z$ where $Z$ is some parameter space and the main algorithmic complication is to select an appropriate structure of $\mathcal{T}^\dagger_\Theta$ such that, given appropriate training, the pseudo-inverse property is satisfied as well as possible.

In the context of tomographic reconstruction, three main research directions have been proposed. The first is so called learned post-processing or learned denoisers. Here, the learned reconstruction operator is of the form

$$
\mathcal{T}_\Theta = \Lambda_\Theta \circ \mathcal{T}^\dagger
$$

where $\Lambda_\Theta: X \to X$ is a learned post-processing operator and $\mathcal{T}^\dagger: Y \to X$ is a pseudo-inverse, e.g. given by Filtered Back-Projection (FBP) in CT reconstruction. This type of method is relatively easy to implement, given that the pseudo-inverse can be applied off-line, before the learning is performed, which reduces the learning to inferring an $X \to X$ transformation. This has been investigated by several authors [6], [7], [8].

Another method is to learn a regularizer and use this regularizer in a classical variational reconstruction scheme according to (3). Examples of this include dictionary learning [9], but several alternative methods have been investigated, such as learning a variational auto-encoder [10] or using a cascade of wavelet transforms, giving the so called scattering transform [11].

Finally, some authors investigate learning the full reconstruction operator, going all the way from data to reconstruction. Doing this in one step is typically very computationally expensive and does not scale to the data sizes encountered in tomographic reconstruction so instead learned iterative schemes have been studied. These schemes resemble classical optimization methods used for tomographic reconstruction such as the PDHG algorithm but use machine-learning to find the best update in each iteration given the last iterate and results of applying the forward operator and its adjoint as input.

Given a structure of $\mathcal{T}_\Theta^\dagger$, the “learning” part refers to choosing a “optimal” set of parameters $\Theta$ given some training data, where the concept of optimality is typically quantified through a loss functional that measures the quality of a learned pseudo-inverse $\mathcal{T}_\Theta^\dagger$. In this article, we assume training data is given by a $(Y \times X)$--valued random variable $(g, f)$ with a known probability density $\mu$. Estimating $\Theta \in Z$ from training data can be formulated as minimizing the loss functional $L(\Theta) \to \mathcal{L}(\Theta)$:

$$
L(\Theta) := \mathbb{E}_{(g,f) \sim \mu} \left[ \| \mathcal{T}^\dagger_\Theta(g) - f \|_X^2 \right].
$$

(7)
A. Related work in learned iterative schemes

We here review results on learned iterative schemes, see [12] for a wider review on machine learning for medical imaging and [13] for usage of machine learning for solving inverse problems in general.

One of the first attempts at learning an optimizer was the Learned Iterative Soft Thresholding Algorithm (LISTA) [19], which learns a version of the ISTA optimizer. An application of this general idea to large scale inverse problems is [15], which learns an ADMM-like scheme for MRI reconstruction.

A further development along these lines, where one learns over a broader class of schemes instead of restricting attention to a specific type of scheme like ADMM is given in [16]. The authors consider finite dimensional inverse problems typically arising in image restoration. This approach was in [13] further extended to non-linear forward operators in to the infinite dimensional setting. The authors also demonstrate the feasibility of learning iterative schemes for (nonlinear, pre-log) CT. Similar approaches in the field of MRI reconstruction has also been considered [17], [18]. Here, the situation is somewhat simpler when compared to CT since the forward operator is approximated by a Fourier transform, i.e. MRI reconstruction amounts to inverting the Fourier transform.

III. CONTRIBUTION AND OVERVIEW OF PAPER

In this paper we propose a learned reconstruction operator \( T_\Theta \) which is given in the form of an unrolled primal-dual optimization scheme inspired by the PDHG method, but where the proximal operators have been replaced with parametrized operators. We demonstrate that this achieves very high performance on the CT reconstruction problem, surpassing several recent learning based methods on both analytical and human data.

We emphasize that we learn the whole reconstruction operator, mapping data to reconstruction, and not just a post-processing nor only the proximal operators in isolation.

In addition, we make all of our code and learned parameters open source so that the community can reproduce the results and apply the methods to other inverse problems [19].

IV. SOLVING INVERSE PROBLEMS USING LEARNED PRIMAL-DUAL SCHEMES

We here introduce how primal-dual algorithms can be learned from data and how this can be used to solve inverse problems.

A. Learned PDHG

The aim is to derive a learned reconstruction scheme inspired by PDHG, algorithm 1. We follow the observation in [20], [21], that proximal operators can be replaced by other operators that are not necessarily proximal operators. The aforementioned publications replace a proximal operator with a denoising operator such as Block Matching 3D (BM3D). Our idea is to replace the proximal operators by parametrized operators where the parameters are learned from training data, resulting in a learned reconstruction operator.

In order to make the learned reconstruction operator well defined and implementable on a computer we also need to select a stopping criterion. Choosing a proper stopping criterion is an active researched area, but for simplicity and usability we use a fixed number of iterates. By selecting a fixed number of iterations, the computation budget is also fixed prior to training which is an desirable property in time constrained applications.

Algorithm 2 below outlines the resulting variant of the PDHG algorithm with \( I \) iterations in which the primal proximal has been replaced by a learned proximal, \( \Gamma_{\Theta^d} \) and the dual proximal by a learned proximal \( \Lambda_{\Theta^p} \). Note that in this article we consider only a single forward model and no regularizing operator, so we have \( K = T, U = Y \), but we give the algorithm in full generality for completeness.

Algorithm 2 Learned PDHG

1: Initialize \( f_0 \in X, h_0 \in U \)
2: for \( i = 1, \ldots, I \) do
3: \( h_{i+1} \leftarrow \Gamma_{\Theta^d} (h_i + \sigma \mathcal{K}(f_i), g) \)
4: \( f_{i+1} \leftarrow \Lambda_{\Theta^p} (f_{i} - \tau \partial \mathcal{K}(f_{i})^*(h_{i+1})) \)
5: \( f_{i+1} \leftarrow f_{i+1} + \theta (f_{i+1} - f_{i}) \)
6: return \( f_I \)

In the algorithm, there are several parameters that need to be selected. These are the parameters of the dual proximal, \( \Theta^d \), the primal proximal, \( \Theta^p \), the step lengths, \( \sigma, \tau \) and the overrelaxation parameter, \( \theta \). In a learned PDHG algorithm these would all be inferred, learned, from training data.

We implemented this algorithm and show its performance in the results section. While the performance was comparable to traditional methods, it did not improve upon the state of the art in deep learning based image reconstruction.

B. Learned Primal-Dual

To gain significant improvements, guided by recent advances in machine learning, the following modifications to the learned PDHG algorithm was done.

- Following [16], [13], extend the primal space to allow the algorithm some “memory” between the iterations.
  \[
  f = [f^{(1)}, f^{(2)}, \ldots, f^{(N_{\text{mem})}}] \in X^{N_{\text{mem}}}
  \]
  Similarly extend the dual space \( U \) to \( U^{N_{\text{dual}}} \).

- Instead of explicitly enforcing updates of the form \( h_i + \sigma \mathcal{K}(f_i) \), allow the network to learn how to combine the previous update with the result of the operator evaluation.

- Instead of hard-coding the over-relaxation \( f_{i+1} \leftarrow f_i + \theta (f_{i+1} - f_i) \), let the network to freely learn in what point the forward operator should be evaluated.

- Instead of using the same learned proximal operators in each iteration allow them to differ. This increases the size of the parameter space but it also significantly improves reconstruction quality.

The above modifications result in a new algorithm, here termed Learned Primal-Dual algorithm, that is outlined in algorithm 3.
Algorithm 3 Learned Primal-Dual

1. Initialize $f_0 \in X^{N_{\text{primal}}}$, $h_0 \in \mathcal{U}^{N_{\text{dual}}}$
2. for $i = 1, \ldots, I$
3. \quad $h_i \leftarrow \Gamma_{\Theta_i}(h_{i-1}, \mathcal{K}(f_{i-1}^{(1)}), g)$
4. \quad $f_i \leftarrow \Lambda_{\Theta_i}(f_{i-1}, [\partial \mathcal{K}(f_{i-1})]^*(h_1))$
5. return $f_I^{(1)}$

1) Choice of starting point: In theory, the Learned Primal-Dual scheme can be used with any choice of starting points $f_0$ and $h_0$. The most simple starting point, both from a conceptual and computational perspective, is zero initialization

\[
\begin{align*}
    f_0 &= [0, 0, \ldots, 0] \\
    h_0 &= [0, 0, \ldots, 0]
\end{align*}
\]

where 0 is the zero element in the primal or dual space.

In cases where a good starting guess is available, it would make sense to use it. One such option is to assume that there exists a pseudo-inverse $\mathcal{T}^\dagger$, e.g. FBP for CT. For the dual variable, the data $g$ enters into each iterate so there is no need for a good initial guess. This gives the starting point

\[
\begin{align*}
    f_0 &= [\mathcal{T}^\dagger(g), \mathcal{T}^\dagger(g), \ldots, \mathcal{T}^\dagger(g)] \\
    h_0 &= [0, 0, \ldots, 0]
\end{align*}
\]

In our tests, we found that providing the Learned Primal-Dual algorithm with such an initial guess marginally decreased training time, but did not give better final results. Given that using the pseudo-inverse $\mathcal{T}^\dagger$ adds more complexity by making the learned reconstruction operator depend on an earlier reconstruction, we report values only from zero-initialization.

2) Connection to variational regularization: We note that by selecting $N_{\text{primal}} = 2$ and $N_{\text{dual}} = 1$ the Learned Primal-Dual algorithm naturally reduces to the classical PDHG algorithm by making the following choices:

\[
\begin{align*}
    \Gamma_{\Theta_i}(h, \mathcal{K}(f^{(2)}), g) &= \text{prox}_{\sigma \mathcal{F}_g}(h + \sigma \mathcal{K}(f^{(2)})) \\
    \Lambda_{\Theta_i}(f^{(1)}, \mathcal{K}(f^{(1)}))^*(h) &= \text{prox}_{\tau \mathcal{G}_g}(f^{(1)} - \tau [\mathcal{K}(f^{(1)})]^*(h)) \\
    &\quad \left(1 + \theta \right) \text{prox}_{\tau \mathcal{G}_g}(f^{(1)} - \theta [\mathcal{K}(f^{(1)})]^*(h)) - \theta f^{(1)}
\end{align*}
\]

Even if the learned proximal operators do not have explicit access to the proximals, the universal approximation property of neural networks [22] guarantees that given sufficient training data these equalities can be approximated arbitrarily well.

A wide range of other optimization schemes can also be seen as special cases of the Learned Primal-Dual algorithm. For example, the gradient descent algorithm with step-length $\alpha$ for solving (6) is given by

\[
f_{i+1} = f_i - \alpha \left( [\partial \mathcal{K}(f_i)]^* (\nabla \mathcal{F}_g (\mathcal{K}(f_i))) + \nabla \mathcal{G}_g(f_i) \right)
\]

and can be obtained by selecting

\[
\begin{align*}
    \Gamma_{\Theta_i}(h, \mathcal{K}(f^{(2)}), g) &= [\nabla \mathcal{F}_g](\mathcal{K}(f^{(2)})) \\
    \Lambda_{\Theta_i}(f^{(1)}, \mathcal{K}(f^{(1)}))^*(h) &= \left[ f^{(1)} - \alpha [\nabla \mathcal{G}_g(f^{(1)}) + [\partial \mathcal{K}(f^{(1)})]^*(h)] \right] \\
    &\quad - \alpha [\nabla \mathcal{G}_g(f^{(1)}) + [\partial \mathcal{K}(f^{(1)})]^*(h)]
\end{align*}
\]

More advanced gradient based methods such as Limited memory BFGS are likewise sub-cases obtained by appropriate choices of learned proximal operators.

In summary, the Learned Primal-Dual algorithm contains a wide range of optimization schemes as special cases. If the parameters are appropriately selected, then the proposed algorithm should always perform at least as well as current variational regularization schemes given the same stopping criteria.

C. Parametrization of the learned proximal operators

Given the above scheme, a parametrization of the learned proximal operators is needed in order to proceed. We used learned proximal operators of the form

\[
\mathcal{W}_{w_j,b_j} : \mathcal{X}^n \rightarrow \mathcal{X}^m
\]

where $\mathcal{I}d$ is the identity operator that makes the network a residual network. There are two main reasons for choosing such a structure. First, proximal operators (as the name implies) are typically close to the identity and second, there is rich evidence in the machine learning literature [23] that networks of this type are easier to train. Heuristically this is because each update does not need to learn the whole update, but only a small offset from the identity.

Additionally, we used affine operators $\mathcal{W}_{w_j,b_j}$ parametrized by weights $w_j$ and biases $b_j$. Following recent advances in deep learning, the affine operators are defined in terms of so called convolution operators\(^1\) (here given on the primal space, but equivalently on the dual space). These are given as affine combinations of regular convolution operators, more specifically:

\[
\mathcal{W}_{w_j,b_j} : \mathcal{X}^n \rightarrow \mathcal{X}^m
\]

where the $k$:th component is given by

\[
\left[ \mathcal{W}_{w_j,b_j}([f^{(1)}, \ldots, f^{(n)}]) \right]^{(k)} = b_j^{(k)} + \sum_{l=1}^n w_j^{(k,l)} \cdot f^{(l)}
\]

where $b_j \in \mathbb{R}^m$ and $w_j \in \mathcal{X}^{n \times m}$.

The non-linearities were chosen to be Parametric Rectified Linear Units (PReLU) functions

\[
\mathcal{A}_{c_j}(x) = \begin{cases} 
    x & \text{if } x \geq 0 \\
    -c_j x & \text{else}
\end{cases}
\]

This type of non-linearity has proven successful in other applications such as classification [24].

\(^1\)Technically cross-correlation operators in the implementation, but these only differ by a flip.
V. Implementation and Evaluation

We evaluate the algorithm on two low dose CT problems. One simplified using analytical phantoms based on ellipses and one with a more realistic forward model and human phantoms. We briefly describe these test cases and how we implemented the Learned Primal-Dual algorithm. We also describe the methods we compare against.

A. Test cases

a) Ellipse phantoms: This problem is identical to [13] and we restate it briefly. Training data is randomly generated ellipses on a \(128 \times 128\) pixel domain. The forward operator is the ray transform and hence \(T = P\).

The projection geometry was a sparse 30 view parallel beam geometry with 182 detector pixels. 5% additive Gaussian noise was added to the projections. Since the forward operator is linear, the adjoint of the derivative is simply the adjoint, which for the ray transform is the back-projection

\[
[\partial T(f)]^* = P^*.
\]

b) Human phantoms: In order to evaluate the algorithm on a clinically realistic use-case we consider reconstruction of simulated data from human abdomen CT scans as provided by Mayo Clinic for the AAPM Low Dose CT Grand Challenge [25]. The data includes full dose CT scans from 10 patients, of which we used 9 for training and 1 for evaluation. We used the 3mm slice thickness reconstructions, giving a total of 2168 training images. Each image was \(512 \times 512\) pixels.

We used a two-dimensional fan-beam geometry with 1000 angles, 1000 pixels, source to axis distance 500mm and axis to detector distance 500mm. In this setting, we consider the more physically correct non-linear forward model given by Beer-Lamberts law

\[
T(f)(\ell) = e^{-\mu P(f)(\ell)}
\]

where the unit of \(f\) is \(g/cm^3\) and \(\mu\) is the mass attenuation coefficient, in this work selected to 0.2 cm\(^{-1}\) which is approximately the value in water at x-ray energies. We used Poisson noise corresponding to \(10^4\) incident photons per pixels, which would correspond to a low dose CT scan. We find the action of the adjoint of the derivative by straightforward computation

\[
[\partial T(f)]^* (g) = -\mu P^* (e^{-\mu P(f)(\cdot)} g(\cdot)) \quad \text{for } g \in Y.
\]

The forward model can also be linearised by applying \(-\log(\cdot)/\mu\) to both the data and forward operator, which then simply becomes the ray-transform as for the ellipse data. We implemented both the pre-log (nonlinear) and post-log (linear) forward models and compare their results.

For validation of the ellipse data case, we simply use the (modified) Shepp-Logan phantom and for the human phantom data we use one held out set of patient data. See fig. 1 for examples.

B. Implementation

The methods described above were implemented in Python using Operator Discretization Library (ODL) [26] and TensorFlow [27]. All operator-related components, such as the forward operator \(T\), were implemented in ODL, and these were then converted into TensorFlow layers using the as_tensorflow_layer functionality of ODL. The neural network layers and training were implemented using TensorFlow. The implementation utilizes abstract ODL structures for representing functional analytic notions and is therefore generic and easily adaptable to other inverse problems. In particular, the code can be easily adapted to other imaging modalities.

We used the ODL operator RayTransform in order to evaluate the ray transform and its adjoint using the GPU accelerated ’astra_gpu’ backend [28].

1) Incorporating the forward operator in neural networks: In order to minimize the loss function (7), Stochastic Gradient Descent (SGD) type methods are typically used and these
require (an estimate of) the gradient of the loss function
\[ \nabla L(\Theta) = E_{(g,f)\sim\mu} \left[ 2[\partial\Theta T^\dagger_{\Theta}(g)]^*(T^\dagger_{\Theta}(g) - f) \right], \]
where \([\partial\Theta T^\dagger_{\Theta}(g)]^*\) is the adjoint of the derivative (with respect to \(\Theta\)) of the learned reconstruction operator applied in \(g^2\). This introduces a challenge since it will depend on each component of the neural network, including the learned proximal operators but also the forward operator \(T\) and the backward operator \([\partial T(f)]^*\), propagated through all \(I\) iterations.

To solve this, we used the built-in automatic differentiation functionality of TensorFlow which uses the chain rule\(^3\). This in turn requires the adjoints of the derivatives of each individual component which for the proximals were computed by TensorFlow and for the operators by ODL.

2) Deep neural network and training details: As mentioned in section IV-C, we used a residual Convolutional Neural Network (CNN) with PReLU nonlinearities. We let the number of data that persists between the iterates be \(N_{\text{primal}} = N_{\text{dual}} = 5\).

The convolutions were all \(3 \times 3\) pixel size, and the number of channels was, for each learned proximal, \(5 \rightarrow 32 \rightarrow 32 \rightarrow 5\). This gives the total number of parameters for the convolutions per learned proximal as \(3^2(5 \cdot 32 + 32^2 + 32 \cdot 5) = 12096\). In addition to this, there was \(32 + 32 + 5 = 69\) bias parameters and as many parameters for the PReLU nonlinearities, for a total of 12234 parameters per learned proximal.

We let the number of unrolled iterations be \(I = 10\), that is the operator \(T\) and the adjoint of its derivative \([\partial T(f_i^{(1)})]^*\) are both evaluated 10 times by the network. Since each iterate involves two 3-layer networks, one for each proximal, the total depth of the network is 60 convolutional layers and the total number of parameters 244680. In the context of deep learning, this is a deep network but with a small number of parameters.

We used the Xavier initialization scheme [29] for the convolution parameters, and initialized all biases to zero.

We trained the network by solving the optimization problem (7) using training data as explained above using the ADAM optimizer in TensorFlow [30]. We used \(10^9\) batches on each problem and used a learning rate schedule according to cosine annealing [31], i.e. the learning rate in step \(t\) was
\[ \eta_t = \frac{\eta_0}{2} \left( 1 + \cos \left( \pi \frac{t}{t_{\text{max}}} \right) \right) \]
where the initial learning rate \(\eta_0\) was set to \(10^{-3}\). We also let the parameter \(\beta_2\) of the ADAM optimizer to 0.99 and let all other parameters use the default choices. We performed global gradient norm clipping [32], limiting the gradient norms to 1 in order to improve training stability and used a batch size of 5 for the ellipse data and 1 for the human phantoms.

We did not use any regularization of the learned parameters, nor did we utilize any tricks such as dropout or batch normalization. Neither did we perform any data augmentation.

The training was done using a single GTX 1080 Ti GPU and took about 11 hours for the ellipse data and 40 hours for the human phantoms.

C. Comparison

We compare the algorithm to several widely used algorithms, including standard FBP and (isotropic) TV regularized reconstruction. We also compare against several learned schemes. These are briefly summarize here, see the references for full descriptions.

The FBP reconstruction done with a Hann filter and used the method \(\text{fbp\_op}\) in ODL. The TV reconstruction was performed using 1000 iterations of the classical PDHG algorithm, implemented in ODL as \(\text{chambolle\_pock\_solver}\). The filter bandwith in the FBP reconstruction and the regularization parameter in the TV reconstruction were selected in order to maximize the Peak Signal to Noise Ratio (PSNR).

The partially \textit{Learned Gradient} method in [13] is similar to the algorithm proposed in this article, but differs in that instead of learning proximal operators it learns a gradient operator and the forward operator enters into the neural network through the gradient of the data likelihood. Publicly available code and parameters [33] were used.

The next comparison is against a deep learning based approach for post-processing based on a so called \(U\)-\textit{Net} [34] following [7]\(^4\). Here an initial reconstruction is first performed using FBP and a neural network is trained on pairs of noisy FBP images and noiseless/low noise ground truth images, learning a mapping between them.

Additionally, our comparison includes learned PDHG, algorithm 2, as well as the following two simplified versions of the Learned Primal-Dual algorithm. The first is a \textit{Learned Primal} algorithm, which does not learn any parameters for the dual proximal, instead it returns the residual
\[ \Gamma_{\Theta_i}(h_{i-1}, T(f_{i-1}^{(2)}), g) = T(f_{i-1}^{(2)}) - g \]
The second, \textit{Learned Residual} algorithm, further simplifies the problem by discarding the forward operator completely, and can be seen as selecting
\[ \Lambda_{\Theta_i}(f_{i-1}, [\partial T(f_{i-1}^{(1)})]^*(h_{i-1}^{(1)})) = \Lambda_{\Theta_i}(f_{i-1}) \]
Since this method does not have access to the data \(g\), we select the initial guess according to a FBP, see (9). This makes the algorithm a learned denoiser.

For the human phantoms we compare both nonlinear and linearized versions of the forward operator, but given that training times are noticeably longer, we only compare to the previously established methods of FBP, TV and U-Net denoising.

All learned algorithms were trained using the same training scheme as outlined in section V-B2.

We measure the run-time, PSNR and the structural similarity index (SSIM) [35].

All methods that we compare against are available in the accompanying source code.

\(^2\)This is often called gradients in the machine learning literature.

\(^3\)Denoted back-propagation in the machine learning literature

\(^4\)We re-implemented the algorithm according to the specifications of the paper but found that using the training procedure as stated in the paper gave sub-optimal results. We hence report values from using the same training scheme as for our other algorithms in order to give a more fair comparison.
VI. RESULTS

The quantitative results for the ellipse data is given in Table I, where we can see that the proposed Learned Primal-Dual scheme out-performs the classical schemes (FBP and TV) significantly w.r.t. the reconstruction error as measured by both PSNR and SSIM. We also note that the Learned Primal-Dual scheme gives a significant improvement over the previous deep learning based methods such as the learned gradient scheme and U-Net based post-processing, giving an improvement exceeding 6 dB. The learned primal dual algorithm also outperforms the Learned PDHG and the Learned Residual algorithms by wide margins.

The only method that achieves results close to the Learned Primal-Dual method is the Learned Primal method, but the Learned Primal-Dual algorithm gives a noticeable improvement of 1.3 dB.

The results are visualized in Fig. 2. We note that small structures, such as the small inserts, are much more clearly visible in the Learned Primal and Learned Primal-Dual reconstructions than in the other reconstructions. We also note that both the Learned PDHG and Learned Primal reconstruction seem to have a halo artefact close to the outer bone which is absent in the Learned Primal-Dual reconstruction.

With respect to run-time the learned methods that involve calls to the forward operator (Learned Gradient, PDHG, Primal, Primal-Dual) are slower than the methods that do not (Learned U-Net, Residual) by a factor \( \approx 6 \). When compared to TV regularized reconstruction all learned methods are at least 2 orders of magnitude faster.

Quantitative results for the human phantoms data are presented in Table II. We note that the FBP reconstruction has a much more competitive image quality than it had for the ellipse data, both quantitatively and visually. It is likely for this reason that the U-Net performs better than it did on the ellipses, outperforming TV by 4.4 dB. However, if we look at the SSIM we note that this improvement does not translate as well to the structural similarity, where the method is comparable to TV regularization.

Both quantitatively and visually, the nonlinear and linear versions of the Learned Primal-Dual algorithm give very similar results. We will focus on the linearized version which gave slightly better results.

The Learned Primal-Dual algorithm gives a 10.5 dB improvement over the FBP reconstruction, a 6.6 dB improvement over TV and 2.2 dB over the learned U-net. This is less than for the ellipse data, but still represents a significant improvement. On the other hand, while the learned U-Net did
not improve the SSIM as compared to TV regularization, the Learned Primal-Dual scheme gives a large improvement.

This improvement is also present in the images when inspected visually in fig. 3. In particular, we see that some artefacts visible in the FBP reconstruction are still discernible in the U-Net and TV reconstructions. Examples include streaks, especially around the edges of the phantom and structures spuriously created from noise, such as a line in the muscle above the right bone. These are mostly missing in the Learned Primal-Dual reconstruction. However, we do note that the images do look slightly over-smoothed. Both of these observations become especially apparent if we look at the zoomed in images, where we note that the Learned Primal-Dual algorithm is able to reconstruct finer detail than the other algorithms, but gives a very smooth texture.

With respect to the run time, the Learned Primal-Dual is more competitive with the FBP and U-Net algorithms for full size data than for the ellipse data. This is because the size of the data is much larger, which increases the runtime of the FBP reconstruction, which is also needed to compute the initial guess for the U-Net. As for the ellipse data, both learned methods outperform TV regularized reconstruction by two orders of magnitude with respect to runtime.

VII. DISCUSSION

The results show that the Learned Primal-Dual algorithm outperforms classical reconstruction algorithm by large margins as measured in both PSNR and SSIM and also improves upon learned post-processing methods for both simplified ellipse data and for human phantoms. In addition, especially for the $512 \times 512$ human phantoms, the reconstruction time is comparable with even filtered back-projection and learned post-processing.

One interesting, and to the best of our knowledge, unique feature of the Learned Primal-Dual algorithm in the field of deep learning based CT reconstruction, is that it gives reconstructions working directly from data, without any initial reconstruction as input.

Since the algorithm is iterative, we can visualize the iterates to gain insight into how it works. In fig. 4 we show some iterates with the nonlinear forward operator. We note that the reconstruction stays very bad until the 8:th iterate when most
structures seem to come in place, but the image is still noisy. Between the 8th and 10th iterate, we see that the algorithm seems to perform an edge-enhancing step. It thus seems like the learned iterative scheme works in two steps, first finding the large scale structures and then fine-tuning the details.

Similarly to the edge-enhancement that seems to be performed in the primal space, we note that in the dual space the sinogram that is back-projected seems to be band-pass filtered to exclude both very low and very high frequencies.

We note that in the very noisy and under-sampled data used for the ellipse phantoms, the learned algorithms that make use of the forward operator, such as the Learned Gradient, Primal and Primal-Dual algorithms outperform even state of the art post-processing methods by large margins and that in this regimen, TV regularization performs relatively well when compared to post-processing methods. This improvement in reconstruction quality when incorporating the forward operator, while still significant, is not as large for the human phantom in which the data was less noisy.

To explain this, we conjecture that in the limit of highly noisy data where the initial reconstruction as given by e.g. FBP becomes very bad, learned schemes that incorporate the forward model and work directly from data, such as the Learned Primal-Dual algorithm, has a significant advantage over post-processing methods and that this advantage increases with decreasing data quality.

Further along these lines, we note that for the human data the post-processing gives a large improvement in PSNR when compared to TV regularization, but that this does not translate to the SSIM. On the other hand, the Learned Primal-Dual algorithm gives an improvement in both PSNR and SSIM. This can be by explained by the learned post-processing being limited by the information content of the FBP while the Learned Primal-Dual scheme works directly with data and is thus limited by the information content of the data, which is greater or equal to that of the FBP. In theory, the Learned Primal-Dual scheme can thus find structures that are not present in the FBP, something post-processing methods cannot.

In these experiments we found that while the algorithm seems to handle non-linear forward models well, we did not observe any significant performance improvement by doing so. This may indicate that performing reconstructions on post-log data is preferable.

The structure of the neural network used in this work was not fine-tuned and we suspect that better results could be obtained by a better choice of network. We also observed that the choice of optimizer and learning rate decay had a large impact on the results, and we suspect that further research into how to correctly train learned reconstruction operators will prove fruitful.

Finally, we observe that the reconstructions, while outperforming all of the compared methods with respect to PSNR and SSIM, suffers from a perceived over-smoothing when inspected visually. We suspect that the particular choice of objective function used in this article, the squared norm (7), is a main cause of this and invite future researchers to implement learned reconstruction operators that use more advanced loss functions such as perceptual losses [36].

Fig. 4: Iterates 2, 4, 6, 8 and 10 in the Learned Primal-Dual algorithm when reconstructing the human phantoms using a nonlinear forward model. Left: Reconstruction ($f^{(1)}_i$). Middle: Point of evaluation for the forward operator ($f^{(2)}_i$). Right: Point of evaluation for the adjoint of the derivative ($h^{(1)}_i$). Windows selected to cover most of the range of the values.

VIII. CONCLUSIONS

We have proposed a new algorithm in the family of deep learning based iterative reconstruction schemes inspired by the PDHG algorithm, where we replace the proximal operators by learned operators. In contrast to several recently proposed algorithms, the new algorithm works directly from tomographic data and does not depend on any initial reconstruction.

We demonstrated that the algorithm gives state of the art results on a ray transform inversion problem for both analytical and human phantoms. For analytical phantoms, it improves upon both classical algorithms such as FBP and TV, and post-processing based algorithms by at least 6 dB while also improving the SSIM. The improvements for the human phantom were more modest, but the algorithm still improves
upon a TV regularized reconstruction by 6.6 dB and gives an improvement of 2.2 dB when compared to a learned post-processing.

We hope that this algorithm will inspire further research in Learned Primal-Dual schemes and that the method will be applied to other imaging modalities.

IX. ACKNOWLEDGMENTS

The work was supported by the Swedish Foundation of Strategic Research grant AM13-0049 and Industrial PhD grant ID14-0055. The work was also supported by Elekta.

REFERENCES

[1] M. Bertero, H. Lantéri, and L. Zanni, “Iterative image reconstruction: a point of view,” in Proceddings of the Interdisciplinary Workshop on Mathematical Methods in Biomedical Imaging and Intensity-Modulated Radiation (IMRT), Pisa, Italy, Y. Censor, M. Jiang, and A. K. Louis, Eds., 2008, pp. 37–63.

[2] H. W. Engl, M. Hanke, and A. Neubauer, Regularization of inverse problems, ser. Mathematics and its Applications. Kluwer Academic Publishers, 2000, no. 375.

[3] O. Scherzer, M. Grasmair, H. Grossauer, M. Haltmeier, and F. Lenzen, Variational Methods in Imaging, ser. Applied Mathematical Sciences. New York: Springer-Verlag, 2009, vol. 167.

[4] A. Chambolle and T. Pock, “A first-order primal-dual algorithm for convex problems with applications to imaging,” HAL-archives, Tech. Rep. 04090826, 2010. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00490826

[5] T. Valkonen, “A primal-dual hybrid gradient method for nonlinear operators with applications to MRI,” Inverse Problems, vol. 30, no. 5, p. 055012, 2014.

[6] H. Chen, Y. Zhang, M. K. Kalra, F. Lin, Y. Chen, P. Liao, J. Zhou, and G. Wang, “Low-Dose CT with a Residual Encoder-Decoder Convolutional Neural Network (RED-CNN),” ArXiv e-prints, Feb. 2017. [Online]. Available: https://arxiv.org/abs/1702.00288

[7] K. H. Jin, M. T. McCann, E. Froustey, and M. Unser, “Deep convolutional neural network for inverse problems in imaging,” ArXiv, ArXiv:cs.CV 1611.03679, 2016. [Online]. Available: https://arxiv.org/abs/1611.03679

[8] E. Kang, J. Min, and J. C. Ye, “Wavenet: a deep convolutional neural network using directional wavelets for low-dose x-ray CT reconstruction,” CoRR, vol. abs/1610.09736, 2016. [Online]. Available: http://arxiv.org/abs/1610.09736

[9] Q. Xu, H. Yu, X. Mou, L. Zhang, J. Hsieh, and G. Wang, “Low-dose x-ray ct reconstruction via dictionary learning,” IEEE Transactions on Medical Imaging, vol. 31, no. 9, pp. 1682–1697, Sept 2012.

[10] T. Meinhardt, M. Möller, C. Hazirbas, and D. Cremers, “Learning Proximal Operators: Using Denoising Networks for Regularizing Inverse Imaging Problems,” ArXiv e-prints, Apr. 2017. [Online]. Available: https://arxiv.org/abs/1704.03488

[11] I. Dokmanic, J. Bruna, S. Mallat, and M. de Hoop, “Inverse problems with invarient multiscalar statistics,” CoRR, vol. abs/1609.05502, 2016. [Online]. Available: http://arxiv.org/abs/1609.05502

[12] G. Wang, “A Perspective on Deep Imaging,” ArXiv e-prints, Sep. 2016. [Online]. Available: https://arxiv.org/abs/1609.04375

[13] J. Adler and O. Oktem, “Solving ill-posed inverse problems using iterative deep neural networks,” ArXiv e-prints, Apr. 2017. [Online]. Available: https://arxiv.org/abs/1704.04058

[14] K. Gregor and Y. LeCun, “Learning fast approximations of sparse coding,” in In Proceedings of the 27:th International Conference on Machine Learning, 2010.

[15] Y. Yang, J. Sun, H. Li, and Z. Xu, “Deep ADMM-Net for compressive sensing MRI,” in Advances in Neural Information Processing Systems, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, 2016, vol. 29, pp. 10–18. [Online]. Available: http://papers.nips.cc/paper/6536-deep-admm-net-for-compressive-sensing-mri.pdf

[16] P. Putzky and M. Welling, “Recurrent inference machines for solving inverse problems,” 2017. [Online]. Available: https://openreview.net/pdf?id=HkSOlP9lg