Invertible Learned Primal-Dual

Jevgenija Rudzusika
KTH Royal Institute of Technology
Stockholm, Sweden
jevaks@kth.se

Buda Bajić
KTH Royal Institute of Technology
Stockholm, Sweden
budabp@kth.se

Ozan Öktem
KTH Royal Institute of Technology
Stockholm, Sweden
ozan@kth.se

Carola-Bibiane Schönlieb
University of Cambridge
Cambridge, UK
cbs31@cam.ac.uk

Christian Etmann
University of Cambridge
Cambridge, UK
cetmann@damtp.cam.ac.uk

Abstract

We propose invertible Learned Primal-Dual as a method for tomographic image reconstruction. This is a learned iterative method based on the Learned Primal-Dual neural network architecture, which incorporates ideas from invertible neural networks. The invertibility significantly reduces the GPU memory footprint of the Learned Primal-Dual architecture, thus making it applicable to 3D tomographic reconstruction as demonstrated in the experiments.

1 Introduction

Combining model-based and data driven methods has demonstrated promising performance in medical image reconstruction [4]. A key idea is to define domain adapted neural networks that integrate components from regularization theory. In learned iterative schemes this is done by unrolling a suitable optimization scheme and replacing selected terms with trainable neural networks [23].

Several unrolled reconstruction methods have been proposed for various imaging modalities: computed tomography (CT) [3], photoacoustic tomography [14], magnetic resonance (MR) [12], and lensless imaging [22]. These are based on unrolling different schemes, such as gradient descent [2], [29], primal-dual [3], alternating direction method of multiplier (ADMM) [22], [27].

Here we focus on CT reconstruction, where unrolled methods have shown an outstanding performance in the low dose 2D setting. However, training and deploying these methods for 3D CT reconstruction is still a challenge due to the significantly increased memory requirements. Recent progress in direction of decreasing the GPU memory footprint of such methods is presented in [13], where a multi-scale approach is combined with learned gradient-descent (LGD), as well as in [29] that relies on greedy training of LGD on image patches.

The aim here is to explore the idea of invertibility as an alternative path for reducing GPU memory requirements. Invertible neural networks (INNs) were introduced in NICE [7] for the purpose of modelling complex probability densities. Here invertibility was achieved by additive coupling, which was later extended to affine coupling in [8]. Since then many works rely on additive/affine coupling to construct invertible architectures [11, 15, 18, 9] for classification, density modelling and

NeurIPS 2021 Workshop on Deep Learning and Inverse Problems, virtual.

Algorithm 1 LPD [3] [a]

```
1: Choose initial primal and dual variables
   $(x_0, u_0) = \texttt{init}(y)$, where $(x_0, u_0) \in (X^N, Y^N)$
2: For $i = 1, 2, \dots, M$ do:
3:     Dual update: $u_i = u_{i-1} + \Gamma^i\left(u_{i-1}, Tx^{(2)}_{i-1}, y\right)$
4:     Primal update: $x_i = x_{i-1} + \Lambda^i\left(x_{i-1}, T^* u^{(1)}_i\right)$
5: return $x^{(1)}_M$
```

[a]Note that this LPD formulation slightly differs from the one given in Alg. 3 in [3], but it still reflects the actual implementation used in experiments in [3].

Algorithm 2 iLPD

```
1: Choose initial primal and dual variables
   $(x_0, u_0) = \texttt{init}(y)$, where $(x_0, u_0) \in (X, Y)$
2: For $i = 1, 3, \dots, 2M - 1$ do:
3:     Dual update: $\begin{cases} x_i = x_{i-1} \\ u_i = u_{i-1} + \Gamma^i(Tx_{i-1}, y) \end{cases}$
4:     Primal update: $\begin{cases} x_{i+1} = x_i + \Lambda^i(T^* u_i) \\ u_{i+1} = u_i \end{cases}$
5: return $x_{2M}$
```

denoising. As outlined in [11], one of the key benefits of INN is that depth of the network can be increased while maintaining a constant memory footprint. This allows to increase the number of unrolled iterations in a learned iterative method i-RIM [26, 25] and to apply deep learning to 3D MR image reconstruction.

Similarly to [26, 25], we combine the idea of INN with the learned iterative method in [3] that introduces the *Learned Primal-Dual (LPD)* architecture for CT reconstruction. The key difference in our work is that we do not rely on affine coupling, since a slight change in the LPD architecture is sufficient to make it invertible. We demonstrate that our architecture significantly reduces the GPU memory requirements in comparison to the original LPD in the 2D setting, while achieving comparable image reconstruction quality. Furthermore, we are able to apply our method in the 3D setting, where it achieves superior results comparing to other methods.

2 Background

2.1 Learned Primal-Dual

The LPD architecture is a domain adapted neural network especially suited for solving an inverse problem. It is typically trained in a supervised manner and it has been used for CT reconstruction. The latter can be formulated as the task of recovering a 2D/3D image $x \in X$ from noisy tomographic data/sinogram $y \in Y$, where

$$y = T(x) + \delta(x)\,. \tag{1}$$

In the above, the (linear) forward operator $T : X \to Y$ models how data is generated in the absence of noise, and $\delta(x) \in Y$ is observation noise. We phrase a machine learning based approach applied to (1) as the problem of finding a (non-linear) parametrized mapping $T^\dagger_\theta : Y \to X$ that satisfies the pseudo-inverse property: $T^\dagger_\theta(y) \approx x$. A key element in this approach is to parametrize the set of pseudo-inverse mappings with parameter $\theta \in \Theta$. Supervised training is performed by minimizing the loss functional

$$\mathrm{L}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(T^\dagger_\theta(y), x)] \tag{2}$$

for training dataset $\mathcal{D}$. Here ℓ measures how close the obtained reconstruction is to the ground truth.

The LPD architecture is inspired by the iterative scheme in the primal-dual hybrid gradient (PDHG) algorithm [5]. This architecture incorporates a forward operator into a deep neural network by unrolling a proximal primal-dual optimization scheme and replacing proximal operators with convolutional neural networks (CNNs). More precisely, the LPD architecture is given in Algorithm 1, where M is the number of unrolling iterates, N is the total number of memory channels and superscripts (1) and (2) denote the 1st and the 2nd channels of assigned variables. The functions Γ^i and Λ^i correspond to CNNs with different learned parameters but with the same architecture for each unrolled iteration i. To simplify notation, we suppress the explicit dependence of Γ^i and Λ^i on the learned parameters $\theta \in \Theta$.

2.2 Invertible Neural Networks via Coupling Layers

Additive coupling [7] is a simple method to construct invertible layers. Let $\tilde{X} = X^{C_1+C_2}$ represent a space of feature maps with $C_1 + C_2$ channels. Let $f : X^{C_1} \to X^{C_2}$ be an arbitrary function (meant to be a neural network). Then, an invertible mapping $\Phi : (x_{i-1}, u_{i-1}) \mapsto (x_i, u_i)$ and its inverse Φ^{-1} are defined by

$$\begin{aligned} x_i &= x_{i-1} + f(u_{i-1}) \\ u_i &= u_{i-1} \end{aligned} \quad \text{and} \quad \begin{aligned} u_{i-1} &= u_i \\ x_{i-1} &= x_i - f(u_i). \end{aligned} \tag{3}$$

A INN of arbitrary depth can then be constructed by a simple composition of these invertible mappings. If activations are restored by successively applying the layers' inverses (instead of storing them), the memory demand is independent of the network's depth, as described in [11]. Due to the need for re-computing activations, one additional forward pass has to be performed per training step.

3 Invertible Learned Primal-Dual

Inspired by additive coupling layers, we define the invertible **dual update mapping** G_y^i and **primal update mapping** F^i. While standard coupling layers typically map between image spaces of the same resolution, we rely on the primal and dual operators T and T^* to convert between the primal and dual spaces, extended to multiple channels (X^N and Y^N). The **dual update mapping** $G_y^i : (x_{i-1}, u_{i-1}) \mapsto (x_i, u_i)$ is given by

$$\begin{aligned} x_i &= x_{i-1} \\ u_i &= u_{i-1} + g_y^i(x_{i-1}), \end{aligned} \tag{4}$$

where the **primal-to-dual mapping** is defined by $g_y^i := g_{y,\text{post}}^i \circ T \circ g_{\text{pre}}^i : X^N \to Y^N$. Here, g_{pre}^i (*pre-operator network*) and $g_{y,\text{post}}^i$ (*post-operator network*) are CNNs, where the pre-operator network operates on the primal space and the post-operator network operates on the dual space. In this context, tomographic data y is a parameter of the post-operator network $g_{y,\text{post}}^i$, even though in practice it is incorporated via concatenation with the input variable x_{i-1}. Note that these networks allow for arbitrarily changing the number of channels if desired.

The **primal update mapping** $F^i : (x_{i-1}, u_{i-1}) \mapsto (x_i, u_i)$ is defined as

$$\begin{aligned} x_i &= x_{i-1} + f^i(u_{i-1}) \\ u_i &= u_{i-1}, \end{aligned} \tag{5}$$

where the **dual-to-primal mapping** is given by $f^i := f_{\text{post}}^i \circ T^* \circ f_{\text{pre}}^i : Y^N \to X^N$, similarly to the g_y^i.

We can now define an unrolled scheme through the composition

$$\mathcal{F}_y = (F^M \circ G_y^M) \circ \cdots \circ (F^1 \circ G_y^1) \quad \text{where} \quad \mathcal{F}_y : X^N \times Y^N \to X^N \times Y^N. \tag{6}$$

Since F^i and G_y^i are invertible, the whole unrolled scheme $\mathcal{F}_y$ is invertible, and we can use memory-efficient backpropagation schemes. Combined with projection operator $P : X^N \times Y^N \to X$ (e.g. retaining only one channel) and an initialization scheme `init`, the unrolled scheme $\mathcal{F}_y$ defines the iLPD reconstruction method:

$$T_\theta^\dagger : y \mapsto P \circ \mathcal{F}_y \circ \texttt{init}(y). \tag{7}$$

The above parametrization of $T_\theta^\dagger$ is general and results in a deep neural network that can be adapted for solving a particular inverse problem. We simplify this general approach to the form described in Algorithm 2, by making several specific choices. First, we choose the number of memory channels for dual and primal variables to be $N = 1$. Next, we substitute the pre-operator networks for both dual and primal updates with identity functions, $g_{\text{pre}}^i = f_{\text{pre}}^i = Id$, and chose the post-operator networks to be 3-layer CNNs, $g_{y,\text{post}}^i = \Gamma^i$ and $f_{y,\text{post}}^i = \Lambda^i$ (see Appendix A.1). Note that M steps in LPD are equivalently represented as $2M$ steps in iLPD. Therefore, the main difference between LPD and iLPD is that Γ^i and Λ^i do not take u_{i-1} and x_i as inputs respectively, i.e. we achieve invertibility of LPD by removing skip connections.

4 Results

4.1 2D CT reconstruction

In the first set of experiments we simulate tomographic data from CT images of human abdomen provided by Mayo Clinic for the AAPM Low Dose CT Grand Challenge [21]. The dataset contains

Table 1: Performance metrics for various reconstruction methods in low-dose CT.

	LPD	LPD-NoMem	iLPD-10	iLPD-20
PSNR	**47.05**	46.90	46.10	46.65
SSIM	**0.9997**	0.9997	0.9996	0.9996
GPU Memory (MiB)	16554	16208	**6268**	6280

3D scans of 10 patients, of which 9 are used for training and one for testing. We split the reconstructed 3D volumes with 3 mm slice thickness into axial slices. This results in 2 378 images of size of size 512×512, of which 2168 are used for training and 210 are used exclusively for testing. In addition, 1% of training images is reserved for model validation during training. Since the images are given in Hounsfield scale, we re-scale those by a factor 10^{-3}, so that pixel intensity values are approximately in the interval $[0; 2]$, which is convenient for applying deep learning models. We use the same procedure for simulating the tomographic data, as described in the original LPD paper [3]. In particular, the forward operator T is a fan-beam ray transform with 1 000 projection angles and 1 000 detector elements and the data is corrupted by Poisson noise (see Appendix A.2).

We compare the performance of the iLPD with the original LPD architecture [3] in terms of PSNR, SSIM and GPU memory footprint. The results are presented in Table 1. The original LPD architecture consists of 10 unrolled iterations ($M = 10$) and has 5 memory channels ($N = 5$). First, we can see that removing memory channels from LPD (as in LPD-NoMem with $N = 1$) results only in a slight drop in performance. Furthermore, enforcing invertibility in iLPD-10 ($N = 1, M = 10$) leads to additional drop in performance comparing to LPD-NoMem. However, this drop can be partially compensated by increasing the number of primal-dual iterations as in iLPD-20 ($M = 20$) leading to performance more close to original LPD. Furthermore, both invertible networks require much less GPU memory during training. In addition, the memory footprint stays almost constant while depth of the network (which depends on the number of unrolled iterations) is increased. To summarize, we can see that enforcing invertibility leads to reduced memory consumption which comes at the cost of slight drop in performance.

4.2 3D CT reconstruction

We evaluate the applicability of our method to 3D data, using a CT database of 42 walnuts scanned with cone beam CT (CBCT), provided by the FleX-ray lab at Centrum Wiskunde & Informatica [6]. Data acquisition was performed with $0.3°$ uniform angle increment resulting in 1 200 projections. All walnuts were scanned with source positioned at 3 different heights and the projection data were combined to create artefact-free ground-truth reconstructions. In our experiments, we select 120 uniformly spaced angles out of 1200. Additionally we down-sample both, measurements and ground-truth reconstructions, by a factor of 2. This results in reconstructions of size 251^3 and projection data of size 384×486. This was the maximum reconstruction size possible under the GPU memory constraints of this study. The maximum reconstruction size from previously published results in 3D reconstruction of this dataset performed by a learned iterative method was 168^3 [13], so with iLPD we managed to reconstruct 3.375 larger volumes than [13] thanks to the reduction of memory consumption achieved by utilizing invertibility.

Additionally, we perform reconstruction with Feldkamp-Davis-Kress (FDK) [10], learned FDK (NN-FDK) [19], post-processing with U-Net [16] and greedily learned gradient descent on patches (gLGDp) [29] (see Appendix A.1). For training of all methods we have chosen 40 walnuts, which leaves one for validation (#1) and one for testing (#2). The obtained reconstructions for the test walnut are shown in Figure 1 (Appendix A.3) while performance metrics are presented in Table 2. Both qualitative and quantitative results show that the proposed iLPD out-performs all competitors. Concerning execution times, clearly FDK is the fastest to execute, this is simply because execution time largely depends on the number of evaluations of T and T^* (FDK has one evaluation of T^*, and iLPD has 20 evaluations of T and T^*). However, for all methods execution time is within acceptable limits for practical applications.

Table 2: Performance metrics for various reconstruction methods in sparse-angle CBCT.

	FDK	NN-FDK	U-Net	gLGDp	iLPD-20
PSNR	22.85	30.14	33.10	32.94	**34.68**
SSIM	0.30	0.76	0.74	0.86	**0.87**
Execution time (sec)	1.06	4.69	3.22	5.74	20.42

5 Conclusion

We have proposed a new algorithm in the family of deep learning based iterative reconstruction schemes. The algorithm is a modified LPD method, where by changing its original architecture we achieved invertibility. The proposed iLPD method requires significantly less GPU memory for training than the original LPD and therefore it is applicable for 3D CT reconstruction. We demonstrated that iLPD algorithm gives state of the art results for both 2D and 3D CT.

References

[1] Jonas Adler, Holger Kohr, and Ozan Öktem. Operator discretization library (ODL). 2017.

[2] Jonas Adler and Ozan Öktem. Solving ill-posed inverse problems using iterative deep neural networks. *Inverse Problems*, 33(12):124007, 2017.

[3] Jonas Adler and Ozan Öktem. Learned primal-dual reconstruction. *IEEE Transactions on Medical Imaging*, 37(6):1322–1332, 2018.

[4] Simon Arridge, Peter Maass, Ozan Öktem, and Carola-Bibiane Schönlieb. Solving inverse problems using data-driven models. *Acta Numerica*, 28:1–174, 2019.

[5] Antonin Chambolle and Thomas Pock. A First-Order Primal-Dual Algorithm for Convex Problems with Applications to Imaging. *Journal of mathematical imaging and vision*, 40(1):120–145, 2011.

[6] Henri Der Sarkissian, Felix Lucka, Maureen van Eijnatten, Giulia Colacicco, Sophia Bethany Coban, and Kees Joost Batenburg. A cone-beam X-ray computed tomography data collection designed for machine learning. *Scientific data*, 6(1):1–8, 2019.

[7] Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-linear Independent Components Estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[8] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP. *arXiv preprint arXiv:1605.08803*, 2016.

[9] Christian Etmann, Rihuan Ke, and Carola-Bibiane Schönlieb. iUNets: Fully invertible U-Nets with learnable up-and downsampling. *arXiv preprint arXiv:2005.05220*, 2020.

[10] Lee A Feldkamp, Lloyd C Davis, and James W Kress. Practical cone-beam algorithm. *Josa a*, 1(6):612–619, 1984.

[11] Aidan N Gomez, Mengye Ren, Raquel Urtasun, and Roger B Grosse. The reversible residual network: Backpropagation without storing activations. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 2211–2221, 2017.

[12] Kerstin Hammernik, Teresa Klatzer, Erich Kobler, Michael P Recht, Daniel K Sodickson, Thomas Pock, and Florian Knoll. Learning a variational network for reconstruction of accelerated MRI data. *Magnetic resonance in medicine*, 79(6):3055–3071, 2018.

[13] Andreas Hauptmann, Jonas Adler, Simon R Arridge, and Ozan Oktem. Multi-scale learned iterative reconstruction. *IEEE Transactions on Computational Imaging*, 2020.

[14] Andreas Hauptmann, Felix Lucka, Marta Betcke, Nam Huynh, Jonas Adler, Ben Cox, Paul Beard, Sebastien Ourselin, and Simon Arridge. Model-based learning for accelerated, limited-view 3-D photoacoustic tomography. *IEEE Transactions on Medical Imaging*, 37(6):1382–1393, 2018.

[15] Jörn-Henrik Jacobsen, Arnold Smeulders, and Edouard Oyallon. i-RevNet: Deep Invertible Networks. In *ICLR 2018-International Conference on Learning Representations*, 2018.

[16] Kyong Hwan Jin, Michael T McCann, Emmanuel Froustey, and Michael Unser. Deep convolutional neural network for inverse problems in imaging. *IEEE Transactions on Image Processing*, 26(9):4509–4522, 2017.

[17] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint*, 1412.6980, 2014.

[18] Diederik P Kingma and Prafulla Dhariwal. Glow: generative flow with invertible 1×1 convolutions. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 10236–10245, 2018.

[19] Marinus J Lagerwerf, Daniël M Pelt, Willem Jan Palenstijn, and Kees Joost Batenburg. A computationally efficient reconstruction algorithm for circular cone-beam computed tomography using shallow neural networks. *Journal of Imaging*, 6(12):135, 2020.

[20] Sil C. van de Leemput, Jonas Teuwen, Bram van Ginneken, and Rashindra Manniesing. MemCNN: A Python/PyTorch package for creating memory-efficient invertible neural networks. *Journal of Open Source Software*, 4(39):1576, 7 2019.

[21] C McCollough. TU-FG-207A-04: Overview of the low dose CT Grand Challenge. *Medical physics*, 43(6Part35):3759–3760, 2016.

[22] Kristina Monakhova, Joshua Yurtsever, Grace Kuo, Nick Antipa, Kyrollos Yanny, and Laura Waller. Unrolled, model-based networks for lensless imaging. 2019.

[23] Vishal Monga, Yuelong Li, and Yonina C Eldar. Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Processing Magazine*, 38(2):18–44, 2021.

[24] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. 2017.

[25] Patrick Putzky, Dimitrios Karkalousos, Jonas Teuwen, Nikita Miriakov, Bart Bakker, Matthan Caan, and Max Welling. i-RIM applied to the fastMRI challenge. *arXiv preprint arXiv:1910.08952*, 2019.

[26] Patrick Putzky and Max Welling. Invert to learn to invert. *Advances in Neural Information Processing Systems*, 32:446–456, 2019.

[27] Jian Sun, Huibin Li, Zongben Xu, et al. Deep ADMM-Net for compressive sensing MRI. *Advances in Neural Information Processing Systems*, 29, 2016.

[28] Wim Van Aarle, Willem Jan Palenstijn, Jeroen Cant, Eline Janssens, Folkert Bleichrodt, Andrei Dabravolski, Jan De Beenhouwer, K Joost Batenburg, and Jan Sijbers. Fast and flexible X-ray tomography using the ASTRA toolbox. *Optics express*, 24(22):25129–25147, 2016.

[29] Dufan Wu, Kyungsang Kim, and Quanzheng Li. Computationally efficient deep neural network for computed tomography image reconstruction. *Medical physics*, 46(11):4763–4776, 2019.

A Appendix

A.1 Implementation details

The methods described above were implemented in Python using Operator Discretization Library (ODL) [1] and Pytorch [24]. All operator-related components, such as the forward operator T, were implemented in ODL with ASTRA [28] as back-end. These were then converted into Pytorch layers utilizing the *OperatorModule* functionality of ODL. The neural network layers and training were implemented using PyTorch. The invertibility is incorporated employing MemCNN [20] library.

We re-implemented LPD network in Pytorch keeping the structure and hyper-parameters the same as in the original work [3]. In particular, we used $M = 10$ unrolled iterations and $N = 5$ memory channels for both primal and dual variables. A three-layer CNN is applied in both image and projection domain at each unrolled iteration for both LPD and iLPD, i.e., Λ^i and Γ^i have two hidden convolutional layers with 32 filters and one convolutional output layer. Parametric Rectified Linear Units (PReLU) are used as activation functions after hidden layers. In 3D experiments 2-dimensional convolutional layers are replaced with 3-dimensional layers, but in both cases the kernel size is 3 along each dimension. Both LPD and iLPD were initialized with zeros in both 2D and 3D experiments.

An implementation of NN-FDK has been provided by the authors of the method [19].

The U-Net post-processor has the same architecture as in [16]. The only difference is that instead of 64 number of filters in convolutional layers, we used 40 due to memory constraints. For the same reason, training was performed on patches of size 128^3 instead of on entire volumes. Here an initial reconstruction is first performed using FDK and a neural network is trained on pairs of noisy FDK images and noiseless/low noise ground truth images, learning a mapping between them.

We implemented gLGDp method as suggested in [29]. We used only 3 unrolled iterations (since no benefit was observed from using more), patch size of 160^3 and same U-net modifications. Although we did not split the projection angles to subsets as in the original work, we did not observe additional improvements from including more than one gradient step within a learned iteration. The algorithm was initialized with FDK reconstructions.

All learned iterative methods as well as U-net post-processor were trained using Adam as the optimizer [17] using the ℓ_2-distance to the ground-truth as loss. Networks are trained for $100,000$ iterations in 2D and for $10,000$ iterations in 3D. In all cases, the batch size was chosen to be 1. The initial learning rate is set to 10^{-3} for LPD and U-net post-processor, 5×10^{-4} for iLPD and 10^{-4} for gLGDp with cosine annealing.

All experiments in 2D and in 3D are executed on single GeForce RTX 3090 GPU with available memory of 24234 MiB. Finally, we provide an implementation of proposed method[1].

A.2 Data simulation

The data used in 2D experiments (Sec.4.1) are corrupted by noise such that the number of photons absorbed by the detector is a Poisson distributed random variable

$$H_{\text{noisy}} = \text{Poisson}\left(H_0 e^{-\mu_0 T(x)}\right),$$

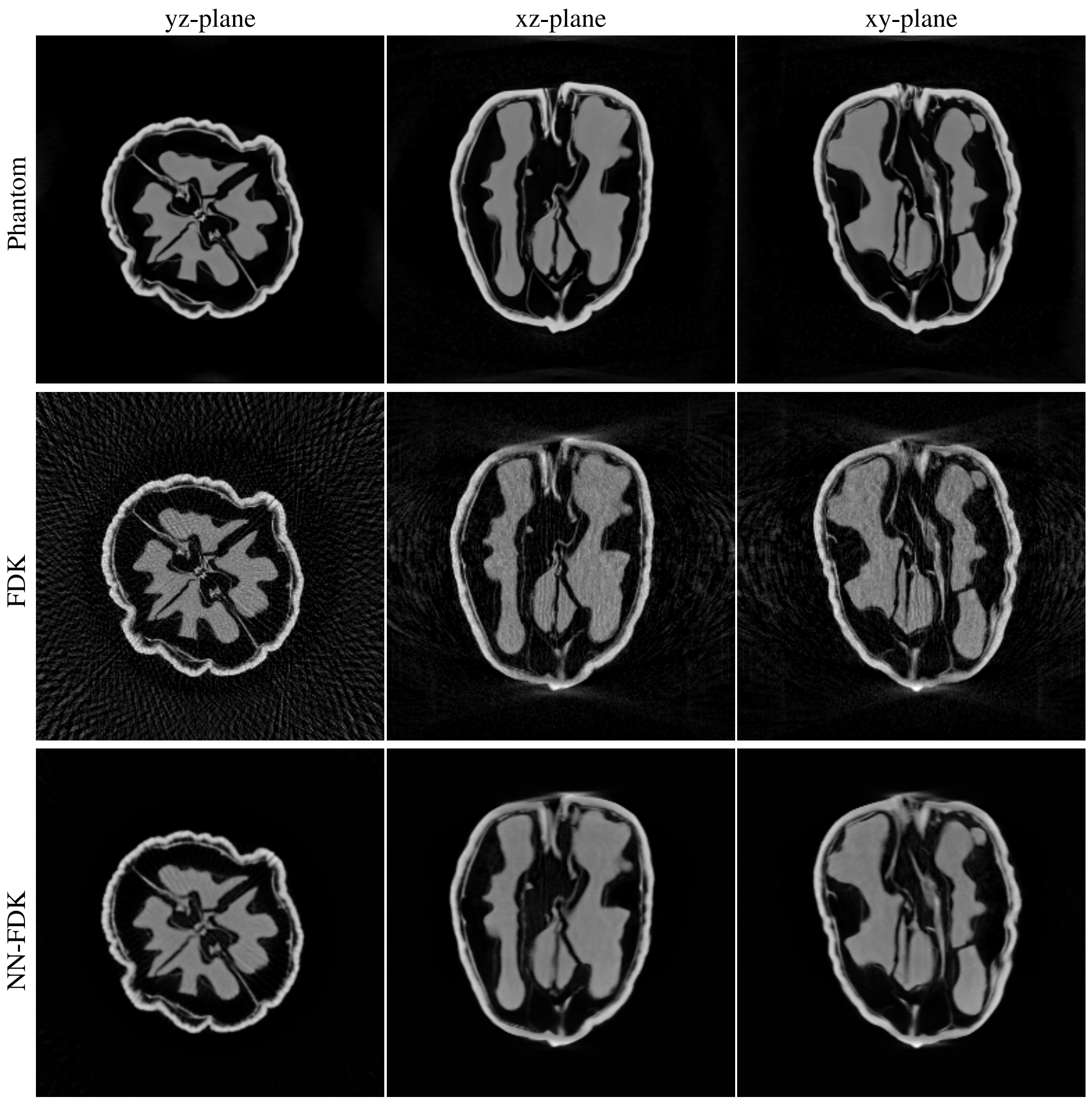
where H_0 the number of photons per pixel before attenuation and $\mu_0 = 0.0192\text{mm}^{-1}$ is X-ray attenuation of water at the mean X-ray energy of 70 keV. Then, we linearize the tomographic data

$$y_{\text{noisy}} = -\ln\left(\frac{1}{H_0} H_{\text{noisy}}\right)$$

so that $y_{\text{noisy}} \approx T(x)$. We use value $H_0 = 10^4$, which corresponds to a low dose CT scan.

[1] github.com/JevgenijaAksjonova/invertible_learned_primal_dual

A.3 Images



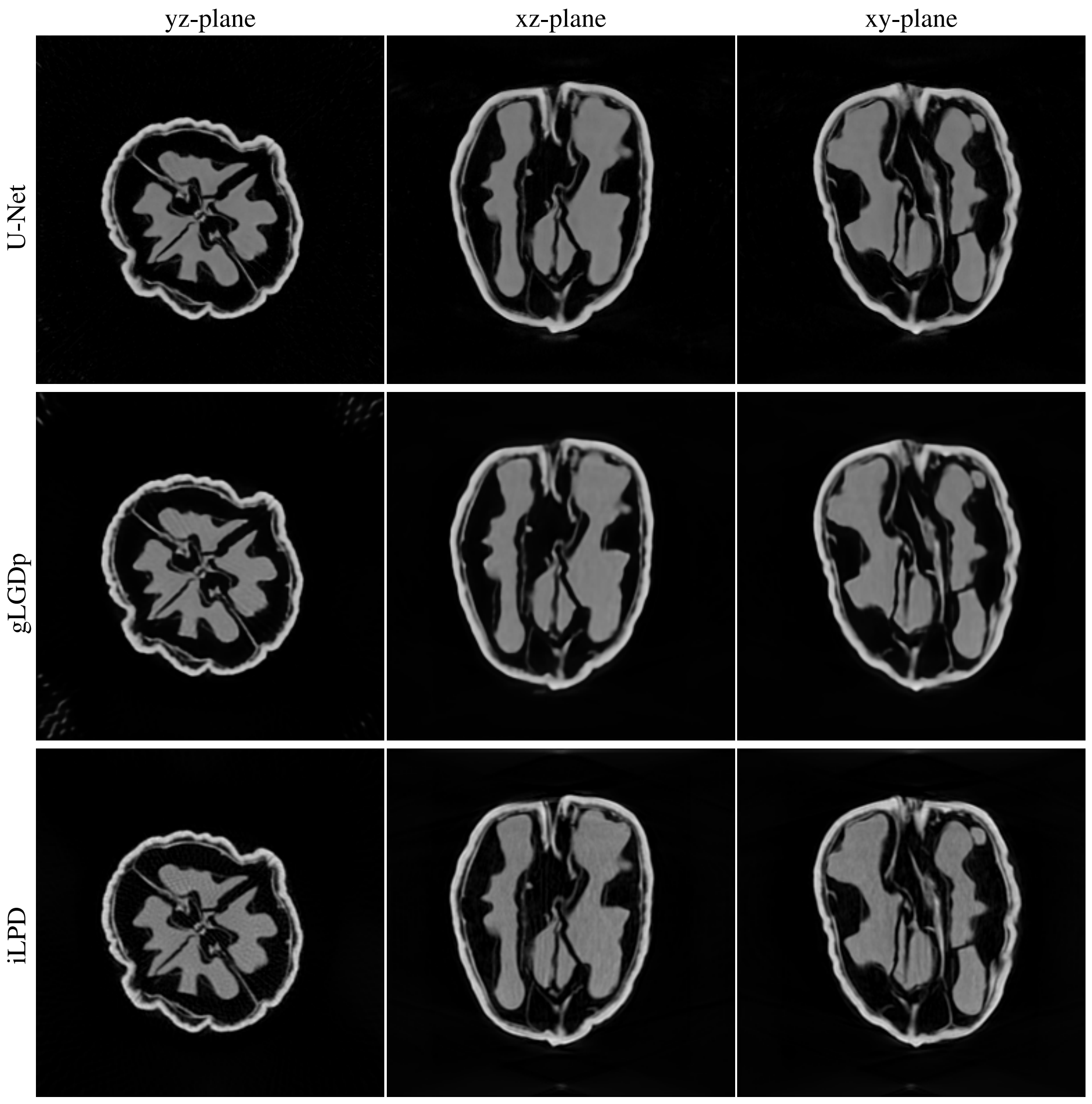


Figure 1: Reconstructions of the walnut used for testing from 120 angles and resolution 251^3. Reconstructions are compared to the phantom computed from a total of 1200 angles and three scanning positions to negate cone beam artefacts. The proposed algorithm iLPD (PSNR=**34.68**) is compared to FDK (PSNR=22.85), NN-FDK (PSNR=30.14), U-Net (PSNR=33.10) and gLGDp (PSNR=32.94). Despite the artefacts appearing in smooth regions, iLPD seems to be better at reconstructing small structural details in comparison to other methods.