Near-Exact Recovery for Sparse-View CT via Data-Driven Methods

Martin Genzel
Mathematical Institute
Utrecht University
m.genzel@uu.nl

Ingo Gühring
Institut für Mathematik
Technische Universität Berlin
guehring@math.tu-berlin.de

Jan Macdonald
Institut für Mathematik
Technische Universität Berlin
macdonald@math.tu-berlin.de

Maximilian März
Institut für Mathematik
Technische Universität Berlin
maerz@math.tu-berlin.de

Abstract

This work presents an empirical study on the design and training of iterative neural networks for image reconstruction from tomographic measurements with unknown geometry. It is based on insights gained during our participation in the recent AAPM DL-Sparse-View CT challenge and a further analysis of our winning submission (team name: `robust-and-stable`) subsequent to the competition period. The goal of the challenge was to identify the state of the art in sparse-view CT with data-driven techniques, thereby addressing a fundamental research question: Can neural-network-based solvers produce near-perfect reconstructions for noise-free data? We answer this in the affirmative by demonstrating that an iterative end-to-end scheme enables the computation of near-perfect solutions on the test set. Remarkably, the fanbeam geometry of the used forward model is completely inferred through a data-driven geometric calibration step.

1 Introduction

In recent years, deep learning methods have been successfully applied to problems of the natural sciences [11, 17, 6]. A prominent example of such *scientific machine learning* is the development of efficient solutions strategies for inverse problems [2, 13], such as those encountered in medical imaging. Despite unprecedented empirical performance in various practical scenarios, a sound theoretical understanding of data-driven reconstruction methods seems to be out of reach to date.

For this reason, more and more critical voices are heard, pointing out a lack of evidence for the reliability of deep-learning-based solution strategies. For instance, Sidky et al. [18] have recently demonstrated that *post-processing* of filtered backprojection images with the prominent UNet-architecture may not yield satisfactory recovery precision in sparse-view computed tomography (CT). This gave rise to the recent AAPM Grand Challenge "Deep Learning for Inverse Problems: Sparse-View Computed Tomography Image Reconstruction". The goal of the challenge was *"to identify the state-of-the-art in solving the CT inverse problem with data-driven techniques"* [19] and thereby to evaluate whether deep-learning-based schemes can achieve (near-)exact precision, similarly to the model-based benchmark of total variation (TV) minimization.

This article presents an analysis of our winning submission to the AAPM challenge (team name: `robust-and-stable`). Besides a description of the underlying methodology, our main objective is to distill several key insights regarding the design and training of our iterative network that are of broader interest. In a nutshell, our approach is rooted in the following (debatable) observation:

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

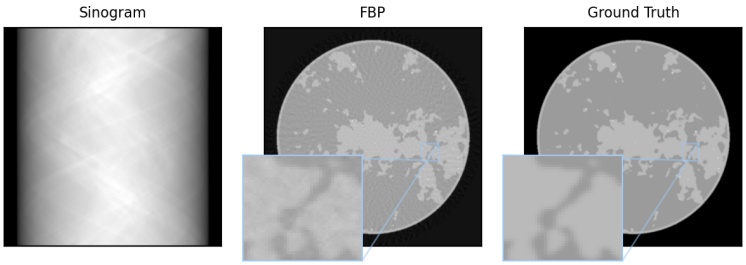

Figure 1: **Example data.** One example of a triple of 128-view sinogram, filtered backprojection, and ground truth phantom image taken from the AAPM challenge training dataset.

High reconstruction accuracy is only possible if the forward model is explicitly incorporated into the solution map, e.g., by an iterative promotion of data-consistency.

We propose a conceptually simple, yet powerful deep learning pipeline which turns a post-processing UNet [16] into an iterative reconstruction scheme. While many of its technical ingredients have been previously reported in the literature, the overall strategy is novel. Our design differs from more common unrolled networks in several aspects, most notably: (a) we make use of a *pre-trained* UNet as the computational backbone, and (b) data-consistency is inspired by an ℓ^2-gradient step, but employs the *filtered* backprojection (FBP) instead of the regular adjoint.

A specific difficulty of the AAPM challenge setup was an *unknown (fanbeam) forward model*. Therefore, a crucial first step of our strategy consists in the data-driven estimation of the underlying fanbeam geometry. We accomplish this by fitting a generic, parameterized fanbeam operator to the provided sinogram-image pairs.

Regarding the actual inversion method, our main findings can be summarized as follows:

(i) End-to-end neural networks can achieve near-perfect accuracy on the prescribed CT reconstruction task.
(ii) Our iterative scheme invokes the (learned) forward operator only 5 times (FBP 6 times), in stark contrast to classical model-based solvers with hundreds or thousands of iterations.
(iii) A careful training procedure is more important than specific details of the network architecture.

The proposed approach could be useful for other inverse problems as well, given that it outperforms state-of-the-art approaches, such as the learned primal-dual algorithm [1], by an order of magnitude.

AAPM Challenge and Dataset. The challenge data consists of synthetic 2D grayscale images comparable to real-world mid-plane breast CT scans, see [19] for details. A fanbeam geometry with 128 projections over 360 degrees was used to create sinograms and FBPs, see Fig. 1 for an example. The exact fanbeam geometry was unknown to the challenge participants, see Step 1 in Section 2. A training set of 4000 triples of 512×512 phantom images, 128-view sinograms, and FBPs, was provided. A test set of 100 tuples of sinograms and FBPs without (publicly available) ground truth phantoms was used for the final challenge evaluation. Initially, about 50 international teams have participated out of which 25 submitted their method for the final evaluation.

2 Methodology

In this section, we give an overview of our approach and motivate its design choices. A public code repository can be found under `https://github.com/jmaces/aapm-ct-challenge`.

Step 1: Data-Driven Geometry Identification. The first step of our reconstruction pipeline learns the unknown forward operator from the provided training data. The continuous version of *tomographic fanbeam measurements* is based on computing line integrals $p(s, \varphi) = \int_{L(s,\varphi)} x_0(x, y)\, \mathrm{d}(x, y)$, where x_0 is the unknown image and $L(s, \varphi)$ denotes a line in fanbeam coordinates, i.e., φ is the *fan rotation angle* and s encodes the *sensor position*; see [4] for more details. In an idealized[1] situation, the fanbeam model is specified by few geometric parameters (see Fig. 6). In the AAPM challenge, the resulting forward operator is *severely ill-posed*, since only the measurements of a

[1] We have found that this basic model was enough to accurately describe the AAPM challenge setup.

few fan rotation angles are acquired. Furthermore, the exact geometric setup is not disclosed to the challenge participants — it is only known that fanbeam measurements are taken. We have addressed this lack of information by a data-driven estimation strategy that fits the geometry parameters to the given training data pairs of discrete images $\boldsymbol{x}_0 \in \mathbb{R}^{512\cdot 512=N}$ and fanbeam measurements $\boldsymbol{y}_0 \in \mathbb{R}^{128\cdot 1024=m}$. In fact, the parameters in Fig. 6 are redundant and it suffices to estimate a smaller subset of parameters, see Appendix B for details.

To that end, we have implemented a discrete fanbeam transform (and corresponding filtered backprojection) from scratch in `PyTorch`. A distinctive aspect of our implementation is the use of a vectorized numerical integration that enables the efficient computation of derivatives with respect to the geometric parameters by means of *automatic differentiation*. This enables a data-driven parameter identification. We estimate the free parameters $\boldsymbol{\theta}_{\texttt{fan}}$ of the implemented forward operator $\boldsymbol{F}[\boldsymbol{\theta}_{\texttt{fan}}] \in \mathbb{R}^{m\times N}$ from the collection of $M = 4000$ sinogram-image pairs $\{(\boldsymbol{y}_0^i, \boldsymbol{x}_0^i)\}_{i=1}^M$ by solving

$$\min_{\boldsymbol{\theta}_{\texttt{fan}}} \ \frac{1}{M}\sum_{i=1}^M \left\|\boldsymbol{F}[\boldsymbol{\theta}_{\texttt{fan}}](\boldsymbol{x}_0^i) - \boldsymbol{y}_0^i\right\|_2^2 \tag{1}$$

with a variant of gradient descent. Subsequently, we determine additional free parameters $\boldsymbol{\theta}_{\texttt{fbp}}$ of the implemented filtered backprojection $\texttt{FBP}[\boldsymbol{\theta}_{\texttt{fan}}, \boldsymbol{\theta}_{\texttt{fbp}}] \in \mathbb{R}^{N\times m}$ by solving

$$\min_{\boldsymbol{\theta}_{\texttt{fbp}}} \ \frac{1}{M}\sum_{i=1}^M \left\|\boldsymbol{x}_0^i - \texttt{FBP}[\boldsymbol{\theta}_{\texttt{fan}}, \boldsymbol{\theta}_{\texttt{fbp}}](\boldsymbol{y}_0^i)\right\|_2^2 , \tag{2}$$

while keeping the already identified parameters $\boldsymbol{\theta}_{\texttt{fan}}$ fixed. Finally, we still recognized a systematic error in our forward model, presumably caused by subtle differences in the numerical integration in comparison to the true forward model of the AAPM challenge. In compensation, we compute the (pixelwise) mean error over the training set, as an additive correction of the model bias. From now on, we will use the short-hand notation $\boldsymbol{F}$ and $\texttt{FBP}$ for the estimated operators, respectively.

Step 2: Pre-Training a UNet as Computational Backbone. The centerpiece of our reconstruction scheme is a UNet-architecture $\boldsymbol{U}[\boldsymbol{\theta}]\colon \mathbb{R}^N \to \mathbb{R}^N$ [16]. It is first employed as a residual network to post-process sparse-view filtered backprojection images, resulting in the reconstruction mapping

$$\text{UNet}[\boldsymbol{\theta}]\colon \mathbb{R}^m \to \mathbb{R}^N,\ \boldsymbol{y} \mapsto [\boldsymbol{U}[\boldsymbol{\theta}] \circ \texttt{FBP}]\,(\boldsymbol{y}). \tag{3}$$

The learnable parameters $\boldsymbol{\theta}$ are trained by (approximately) solving

$$\min_{\boldsymbol{\theta}} \ \frac{1}{M}\sum_{i=1}^M \left\|\boldsymbol{x}_0^i - \text{UNet}[\boldsymbol{\theta}](\boldsymbol{y}_0^i)\right\|_2^2 + \mu\cdot\|\boldsymbol{\theta}\|_2^2 , \tag{4}$$

where we choose $\mu = 10^{-3}$. This is tackled by 400 epochs of mini-batch stochastic gradient descent and the Adam optimizer [9] with initial learning rate 0.0002 and batch size 4.

Step 3: Constructing an Iterative Scheme. Our main reconstruction method, called ItNet, incorporates the (approximate) forward model $\boldsymbol{F}$ from Step 1 via the following iterative procedure:

$$\text{ItNet}_K[\boldsymbol{\theta}]\colon \mathbb{R}^m \to \mathbb{R}^N,\ \boldsymbol{y} \mapsto \left[\bigcirc_{k=1}^K \left(\mathcal{DC}_{\lambda_k,\boldsymbol{y}} \circ \boldsymbol{U}[\tilde{\boldsymbol{\theta}}_k]\right) \circ \texttt{FBP}\right](\boldsymbol{y}), \tag{5}$$

with learnable parameters $\boldsymbol{\theta} = \{\tilde{\boldsymbol{\theta}}_k, \lambda_k\}_{k=1}^K$, and the k-th *data-consistency* layer

$$\mathcal{DC}_{\lambda_k,\boldsymbol{y}}\colon \mathbb{R}^N \to \mathbb{R}^N,\ \boldsymbol{x} \mapsto \boldsymbol{x} - \lambda_k\cdot\texttt{FBP}(\boldsymbol{F}\boldsymbol{x} - \boldsymbol{y}).$$

ItNet is trained analogously to (4) with $\mu = 10^{-4}$. The UNet-parameters $\tilde{\boldsymbol{\theta}}_k$ are initialized by the weights obtained in Step 2. In summary, the central aspects of the architecture in (5) and important design choices are:

(i) The computational centerpiece is the **UNet**-architecture. This contrasts earlier generations of unrolled iterative schemes, which rely on basic convolutional blocks instead, e.g., see [1, 23]. Other recent state-of-the-art architectures also rely on more advanced sub-networks, e.g., see [10, 12, 7, 15, 20].

(ii) The **data-consistency** layer is inspired by a gradient step $\boldsymbol{x} \mapsto \boldsymbol{x} - \lambda_k\cdot\boldsymbol{F}^\mathsf{T}(\boldsymbol{F}\boldsymbol{x} - \boldsymbol{y})$ on the loss $\boldsymbol{x} \mapsto \frac{\lambda_k}{2}\|\boldsymbol{F}\boldsymbol{x} - \boldsymbol{y}\|_2^2$. However, we replace the unfiltered backprojection $\boldsymbol{F}^\mathsf{T}$ by its filtered counterpart $\texttt{FBP}$ for two reasons: (a) counteracting the fact that the unfiltered backprojection is smoothing, and (b) producing images with pixel values at the right intensity scale.

Table 1: **Comparison evaluation.** Average RMSE on a subset of 125 validation images from the AAPM data. See Appendix C for details on ItNet_4 and ItNet-post trained for the challenge.

Baselines		Our Network Variants				Comparison Networks	
Chall. FBP	Our FBP	UNet	ItNet_4	ItNet-post	ItNet-post (ens.)	Tiramisu	LPD
5.72e-3	3.40e-3	3.50e-4	1.64e-5	1.05e-5	**6.42e-6**	2.24e-4	1.24e-4

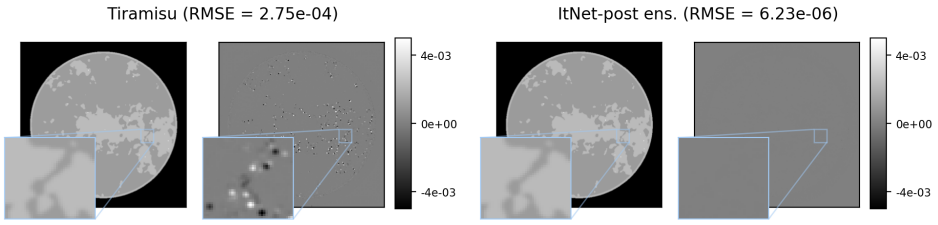

Figure 2: **Reconstruction results.** We display reconstructions and errors for a validation image. We compare a post-processing Tiramisu with ItNet-post. The ground truth image is omitted, since it is visually indistinguishable from ItNet-post. See Fig. 7 for the corresponding FBP reconstructions.

(iii) It is crucial to **initialize** the UNet-parameters $\tilde{\boldsymbol{\theta}}_k$ by the weights from Step 2. This increases both the speed of convergence and the final accuracy (see Fig. 8). In other words, a careful initialization of the UNet-block leads to a better local minima. To the best of our knowledge, such an effect has not been reported in the literature yet.

In our experiments, we witnessed only minor effects by computing more than $K = 5$ iterations in (5). We discuss the choice of K in more detail in Appendix A, in particular, see Fig. 4.

3 Results and Analysis

In terms of quantitative similarity measures, we restrict ourselves to reporting the average root-mean-squared-error (RMSE), which was the main evaluation metric for the AAPM challenge.

Winning the Challenge. With an ensembling of ten ItNet_5 (more precisely a variant thereof referred to as ItNet-post, see Appendix C) we were able to achieve near-exact recovery and win the AAPM challenge with a margin of about an order of magnitude compared to the runners-up teams. Notably, four out of the five best performing teams estimated the forward fanbeam operator and made use of the sinogram data. Two out of them obtain a TV minimization solution that is further processed by a learned network. This requires a much higher number of iterations compared to our ItNet-post.

Comparison Evaluations. We compare variants of ItNet with different baselines and other state-of-the-art methods.[2] More precisely, we consider a post-processing of the FBP by the (in comparison to the UNet-architecture) more advanced *Tiramisu-architecture* [3, 5, 8] as well as the iterative *learned primal-dual* (LPD) scheme [1] (modified by replacing the unfiltered backprojection with the FBP). The average results are reported in Table 1. To give a visual impression as well, reconstructions of an image from the validation set can be found in Fig. 2 and 7.

Additional experiments regarding the aspects of data-consistency and number K of iterations can be found in Appendix A.

[2]We report the RMSE on a subset of 125 images from the training set used for validation. Hence, values differ slightly from the actual results on the official test set. In the final challenge evaluation, ItNet-post has achieved an RMSE of 6.37e-6, see https://www.aapm.org/GrandChallenge/DL-sparse-view-CT/winners.asp.

4 Discussion

We have demonstrated that, using only few evaluations of the forward model, data-driven approaches can achieve (near-)exact recovery on sparse-view CT tasks. While our approach provides first evidence of feasibility, several aspects are beyond the scope of this work. The reconstruction error of ItNet-post reported in Table 1 is not zero. However, it is close to numerical precision, and from a practical viewpoint, the inverse problem of the AAPM challenge can be considered 'solved' by our method. The academic setup of the challenge has provided an ideal experimental area to test our research hypothesis that (near-)exact recovery is possible. Although this is an important reliability check, a foundational understanding of learned reconstruction methods is still in its infancy. Similar case studies for different inverse problems and more realistic data are natural steps for future research.

Acknowledgments and Disclosure of Funding

We would like to thank the organizers of the AAPM DL-Sparse-View CT challenge for hosting the event and providing the data and Emil Sidky in particular for interesting and valuable discussions after the end of the challenge period. M. G. is supported by the Priority Programme DFG-SPP 1798 of the German Research Foundation (DFG). I. G. is supported from the Research Training Group DAEDALUS (GRK 2433) funded by the German Research Foundation (DFG).

References

[1] J. Adler and O. Öktem. Learned primal-dual reconstruction. *IEEE Trans. Med. Imag.*, 37(6):1322–1332, 2018.

[2] S. Arridge, P. Maass, O. Öktem, and C.-B. Schönlieb. Solving inverse problems using data-driven models. *Acta Numer.*, 28:1–174, 2019.

[3] T. A. Bubba, G. Kutyniok, M. Lassas, M. März, W. Samek, S. Siltanen, and V. Srinivasan. Learning the invisible: A hybrid deep learning-shearlet framework for limited angle computed tomography. *Inverse Probl.*, 35(6):064002, 2019.

[4] J. A. Fessler. Analytical tomographic image reconstruction methods (chapter 3 of book draft). https://web.eecs.umich.edu/~fessler/book/c-tomo.pdf, 2017.

[5] M. Genzel, J. Macdonald, and M. März. Solving Inverse Problems With Deep Neural Networks – Robustness Included? Preprint arXiv:2011.04268, 2020.

[6] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[7] K. Hammernik, J. Schlemper, C. Qin, J. Duan, R. M. Summers, and D. Rueckert. Σ-net: systematic evaluation of iterative deep neural networks for fast parallel MR Image reconstruction. Preprint arXiv:1912.09278, 2019.

[8] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio. The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11–19, 2017.

[9] D. P. Kingma and J. Ba. Adam: a method for stochastic optimization. Preprint arXiv:1412.6980, 2014.

[10] F. Knoll, T. Murrell, A. Sriram, N. Yakubova, J. Zbontar, M. Rabbat, A. Defazio, M. J. Muckley, D. K. Sodickson, C. L. Zitnick, and M. P. Recht. Advancing machine learning for MR image reconstruction with an open competition: Overview of the 2019 fastMRI challenge. *Magn. Reson. Med.*, 84(6):3054–3070, 2020.

[11] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[12] M. J. Muckley, B. Riemenschneider, A. Radmanesh, S. Kim, G. Jeong, J. Ko, Y. Jun, H. Shin, D. Hwang, M. Mostapha, S. Arberet, D. Nickel, Z. Ramzi, P. Ciuciu, J.-L. Starck, J. Teuwen, D. Karkalousos, C. Zhang, A. Sriram, Z. Huang, N. Yakubova, Y. Lui, and F. Knoll. State-of-the-art machine learning MRI reconstruction in 2020: Results of the second fastMRI challenge. Preprint arXiv:2012.06318, 2020.

[13] G. Ongie, A. Jalal, R. G. Baraniuk, C. A. Metzler, A. G. Dimakis, and R. Willett. Deep learning techniques for inverse problems in imaging. *IEEE J. Sel. Areas Inf. Theory*, 1(1):39–56, 2020.

[14] P. Putzky and M. Welling. Recurrent inference machines for solving inverse problems. Preprint arXiv:1706.04008, 2017.

[15] Z. Ramzi, P. Ciuciu, and J.-L. Starck. XPDNet for MRI reconstruction: an application to the fastMRI 2020 brain challenge. Preprint arXiv:2010.07290, 2020.

[16] O. Ronneberger, P. Fischer, and T. Brox. U-Net: convolutional networks for biomedical image segmentation. In N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, editors, *Medical Image Computing and Computer Assisted Intervention – MICCAI 2015*, pages 234–241. Springer Cham, 2015.

[17] J. Schmidhuber. Deep learning in neural networks: An overview. *Neural Netw.*, 61:85–117, 2015.

[18] E. Sidky, I. Lorente, J. G. Brankov, and X. Pan. Do CNNs solve the CT inverse problem? *IEEE Trans. Biomed. Eng.*, 68(6):1799–1810, 2021.

[19] E. Sidky, X. Pan, J. Brankov, I. Lorente, S. Armato, K. Drukker, L. Hadjiyski, N. Petrick, K. Farahani, R. Munbodh, K. Cha, J. Kalpathy-Cramer, B. Bearce, and AAPM Working Group on Grand challenges. Deep learning for inverse problems: Sparse-view computed tomography image reconstruction (dl-sparse-view ct). https://www.aapm.org/GrandChallenge/DL-sparse-view-CT/, 2021.

[20] A. Sriram, J. Zbontar, T. Murrell, A. Defazio, C. L. Zitnick, N. Yakubova, F. Knoll, and P. Johnson. End-to-end variational networks for accelerated MRI reconstruction. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 64–73, 2020.

[21] Y. Wu and K. He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[22] T. Würfl, F. C. Ghesu, V. Christlein, and A. Maier. Deep learning computed tomography. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016*, pages 432–440, 2016.

[23] Y. Yang, J. Sun, H. Li, and Z. Xu. Deep ADMM-Net for compressive sensing MRI. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 10–18. Curran Associates, Inc., 2016.

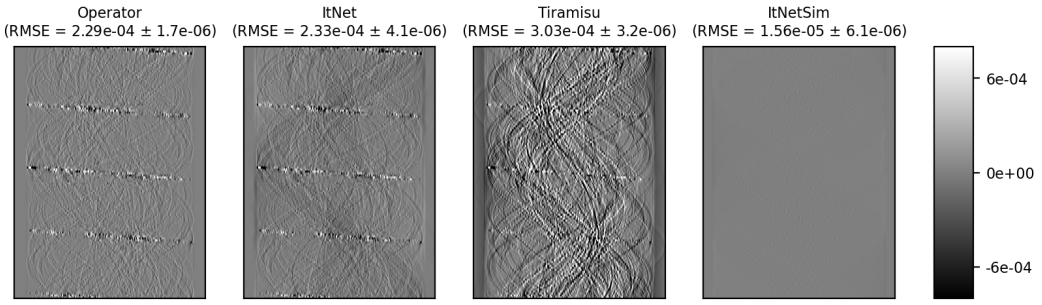

Figure 3: **Data consistency.** We analyze the accuracy of our forward model by displaying the error $\boldsymbol{y}_0 - \boldsymbol{F}\boldsymbol{x}_0$ for sinogram-image pairs $(\boldsymbol{y}_0, \boldsymbol{x}_0)$ from the validation set (left). The visualization of $\boldsymbol{y}_0 - \boldsymbol{F} \cdot \text{ItNet}(\boldsymbol{y}_0)$ is visually nearly indistinguishable (second from left), which shows that ItNet inherits the inaccuracies from the forward model. Indeed, ItNetSim allows for a much reduced consistency error in the case of an exact forward model (right). Computing $\boldsymbol{y}_0 - \boldsymbol{F} \cdot \text{Tiramisu}(\boldsymbol{y}_0)$ reveals a lack of data-consistency of the Tiramisu post-processing (second from right). All images are shown in the same dynamical range.

A Additional Results and Analysis

We provide further experiments and results complementing Section 3.

Data-Consistency. We analyze the aspect of data-consistency, i.e., the term $\boldsymbol{y}_0 - \boldsymbol{F} \cdot \text{ItNet}(\boldsymbol{y}_0)$, in Fig. 3. We observe that the consistency error is dominated by the error from estimating the forward model in Step 1 (see Section 2) indicating that the performance of ItNet can be improved if the exact forward model is available. To verify this hypothesis, we train an ItNet on sinogram data simulated from the ground truth phantom images using our estimated forward model. As expected, the resulting ItNetSim shows an improved reconstruction accuracy (about factor 2) and a much reduced data-consistency error (see Fig. 3 right image). It is also interesting to note that $\boldsymbol{y}_0 - \boldsymbol{F} \cdot \text{Tiramisu}(\boldsymbol{y}_0)$ is about a factor 20 larger than the consistency error of ItNet, revealing that the post-processing by Tiramisu suffers from a lack of data-consistency.

The Deeper the Better? Incorporating the forward operator is an essential ingredient for highly accurate reconstruction schemes. A central question is how many forward/adjoint operator evaluations are required. We address this by training ItNet_K for different numbers of iterations.[3] Fig. 4 shows that, in contrast to classical model-based methods like TV-minimization, only a few forward operator evaluations suffice to achieve near-exact recovery using ItNet. Here, we observe a trade-off between increasing the model capacity and the difficulty of optimizing the resulting network. One possibility to overcome some of the difficulties is sharing the UNet parameters between iterations, i.e., setting $\tilde{\boldsymbol{\theta}}_1 = \cdots = \tilde{\boldsymbol{\theta}}_K$.

In Fig. 4, we observe that (a) not sharing the UNet weights consistently outperforms weight sharing by a small margin independent of the number of iterations; (b) there is a sweetspot at about $K = 5$ after which the improvement in accuracy resulting from increasing K is negligable and only the training time increases. Fig. 5 clearly demonstrates that weight sharing also changes the reconstruction dynamic within the ItNet. Earlier iteration steps of the ItNets trained with weight sharing are more effective while the non-weight-shared counterparts draw most of the performance from the later steps. We conjecture that an improved training strategy of networks without weight sharing might unlock the potential of the earlier UNet-blocks. This could lead to an even wider performance gap between the final reconstruction accuracy of ItNets with and without weight sharing. A systematic study of this aspect is left to future research.

[3] Due the immense computational effort required to conduct such an experiment, this was done on subsampled 256×256 phantom images and simulated 64-view sinograms.

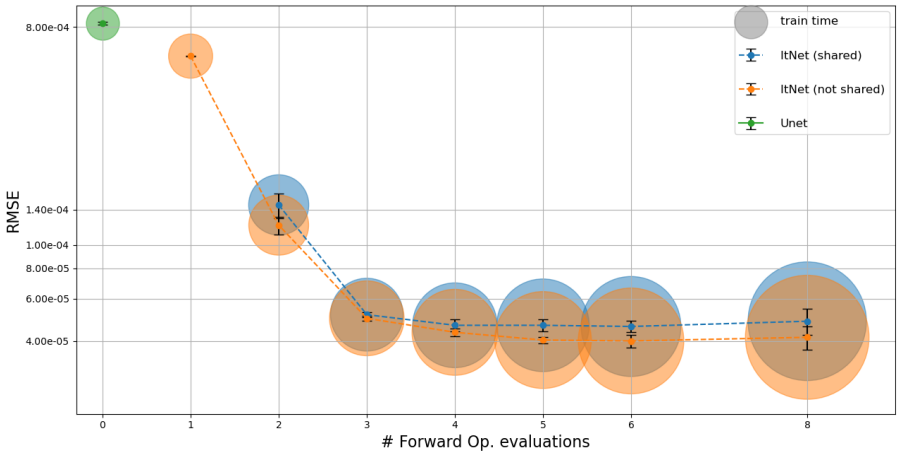

Figure 4: **The deeper the better?** Accuracy of ItNet_K for different K with (blue) and without (orange) UNet weight sharing. The radii of the circles are proportional to the training time. The mean RMSE ($\pm$ standard deviation) on a hold-out evaluation data set is reported over five different training/validation splits. The original challenge data was subsampled to the half resolution 256×256 for this experiment.

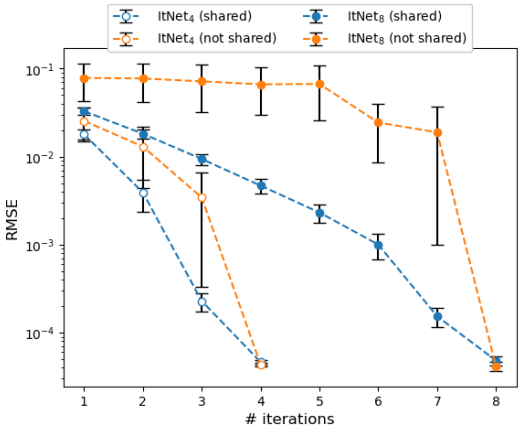

Figure 5: **A look inside.** Accuracy of ItNet_4 and ItNet_8 with (blue) and without (orange) UNet weight sharing when using only the first k iteration steps and discarding the rest. The mean RMSE ($\pm$ standard deviation) on a hold-out evaluation data set is reported over five different training/validation splits. The original challenge data was subsampled to the half resolution 256×256 for this experiment.

B Details on Step 1: Data-Driven Geometry Identification

The fanbeam model is specified by the following geometric parameters (see Fig. 6):

- d_{source} – the distance of the X-ray source to the origin,
- d_{detector} – the distance of the detector array to the origin,
- n_{detector} – the number of detector elements,
- s_{detector} – the spacing of the detector elements along the array,
- n_{angle} – the number of fan rotation angles,
- $\boldsymbol{\varphi} \in [0, 2\pi]^{n_{\text{angle}}}$ – a discrete list of rotation angles.

In order to fit the above set of parameters to the given training data, we first observe that the parametrization is redundant, and without of loss of generality, we may assume that $s_{\text{detector}} = 1$ (by rescaling d_{detector} appropriately). Further, if the field-of-view angle γ is known, then the relation

$$d_{\text{detector}} = \frac{n_{\text{detector}} \cdot s_{\text{detector}}}{2 \tan \gamma} - d_{\text{source}} \tag{6}$$

can be used to eliminate another parameter. Thus, the fanbeam geometry is effectively determined by the reduced parameter set $(d_{\text{source}}, n_{\text{detector}}, n_{\text{angle}}, \boldsymbol{\varphi})$, where $n_{\text{angle}} = 128$ and $n_{\text{detector}} = 1024$

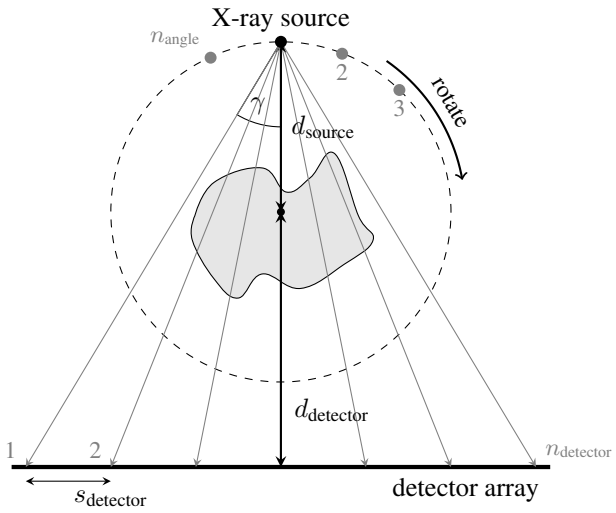

Figure 6: **Fanbeam geometry.** Illustration of the parameters determining the geometry of the fanbeam CT model.

can be directly derived from the provided training data. We determine the field of view angle as $\gamma = \arcsin(256/d_{\text{source}})$, so that the maximum inscribed circle in the discrete image is exactly contained within each fan of lines, which is a common choice for fanbeam CT. Hence, (6) leads to

$$d_{\text{detector}} = 2 \cdot s_{\text{detector}} \cdot \sqrt{d_{\text{source}}^2 - 256^2} - d_{\text{source}} \, .$$

The main difficulty of Step 1 in Section 2 lies in the estimation of the remaining parameters $(d_{\text{source}}, \boldsymbol{\varphi})$. To that end, we have implemented a discrete fanbeam transform from scratch in `PyTorch` (together with corresponding filtered backprojection). More precisely, we use a ray-driven numerical integration for the forward model and a pixel-driven and sinogram-reweighting-based filtered backprojection (with a Hamming filter) [4, Sec. 3.9.2]. In addition to the parameters $(d_{\text{source}}, \boldsymbol{\varphi})$, we introduce learnable scaling factors s_{fwd} and s_{fbp} for the forward and inverse transform, respectively. They account for ambiguities in chosing discretization units of distance compared to actual physical units.

As indicated in Section 2, we estimate the free parameters $\boldsymbol{\theta}_{\texttt{fan}} = (s_{\text{fwd}}, d_{\text{source}}, \boldsymbol{\varphi}) \in \mathbb{R}^{130}$ of the implemented forward operator $\boldsymbol{F}[\boldsymbol{\theta}_{\texttt{fan}}] \in \mathbb{R}^{m \times N}$ in a deep-learning-like fashion: The ability to compute derivatives $\frac{\mathrm{d}\boldsymbol{F}}{\mathrm{d}\boldsymbol{\theta}_{\texttt{fan}}}$ allows us to make use of the $M = 4000$ sinogram-image pairs $\{(\boldsymbol{y}_0^i, \boldsymbol{x}_0^i)\}_{i=1}^M$ by solving, cf. (1),

$$\min_{\boldsymbol{\theta}_{\texttt{fan}}} \frac{1}{M} \sum_{i=1}^M \left\| \boldsymbol{F}[\boldsymbol{\theta}_{\texttt{fan}}](\boldsymbol{x}_0^i) - \boldsymbol{y}_0^i \right\|_2^2 \tag{7}$$

with a variant of gradient descent (see remark (i) below). Finally, we determine $\boldsymbol{\theta}_{\texttt{fbp}} = s_{\text{fbp}}$ by solving, cf. (2),

$$\min_{s_{\text{fbp}}} \frac{1}{M} \sum_{i=1}^M \left\| \boldsymbol{x}_0^i - \texttt{FBP}[\boldsymbol{\theta}_{\texttt{fan}}, s_{\text{fbp}}](\boldsymbol{y}_0^i) \right\|_2^2 , \tag{8}$$

while keeping the already identified parameters fixed. Some final remarks regarding the identification of the forward model:

(i) The formulation (7) is non-convex and therefore it is not clear whether gradient descent enables an accurate estimation of the underlying fanbeam geometry. Indeed, standard gradient descent was found to be sensitive to the initialization of $\boldsymbol{\theta}_{\texttt{fan}}$ and got stuck in bad local minima. To overcome this, we solve (7) by a *block coordinate descent* instead, alternatingly optimizing over s_{fwd}, d_{source}, and $\boldsymbol{\varphi}$ with individual learning rates. This strategy was found to effectively account for large deviations of gradient magnitudes of the different parameters. Indeed, we observed a fast convergence and a reliable identification of $\boldsymbol{\theta}_{\texttt{fan}}$, independently of the initialization.

(ii) In principle, the strategy of (7) requires only few training samples to be successful. However, when verifying the robustness of the outlined strategy against measurement noise, we observed that it is beneficial to employ more training data.

An example reconstruction comparing the estimated `FBP` with the `FBP` provided by the challenge is shown in Fig. 7, which complements Fig. 2.

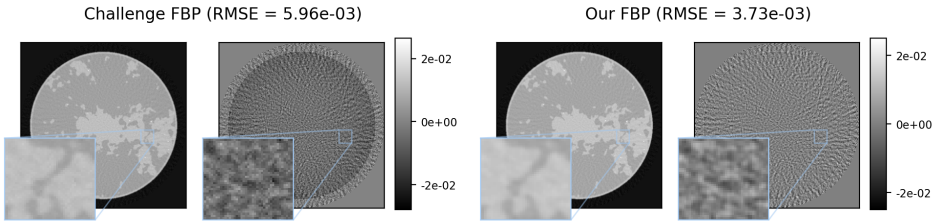

Figure 7: **FBP reconstruction results.** We display reconstructions and errors for a validation image. We compare the challenge FBP with our own FBP obtained from Step 1. See Fig. 2 for the corresponding network reconstructions.

C Details on the Exact Challenge Setup

To ensure reproducability, we give an exact account of how we trained our winning method ItNet-post for the AAPM challenge. Since the systematic investigation of Section 3 and Appendix A of several aspects of ItNet was done after the competition period, it became clear that not all of the following details have a significant impact on the performance.

We start by training an ItNet_4 (with weight sharing) for 500 epochs of mini-batch stochastic gradient descent and Adam optimizer with an initial learning rate of $8 \cdot 10^{-5}$ and a batch size of 2 (restarting Adam after 250 epochs). Then we improve the accuracy by the following **post-training** strategy: First, the ItNet is extended by one more iteration to obtain an ItNet_5 and all $\tilde{\boldsymbol{\theta}}_k$ are initialized by the optimized weights of the previously trained ItNet_4. Then, ItNet-post is fine-tuned by keeping the weights $\tilde{\boldsymbol{\theta}}_1 = \tilde{\boldsymbol{\theta}}_2 = \tilde{\boldsymbol{\theta}}_3$ of the first three UNets fixed and training only the last two iterations (without weight sharing). Hoping for an additional speed-up of the training, we use the initialization $(\lambda_1, \lambda_2, \lambda_3, \lambda_4) = [1.1, 1.3, 1.4, 0.08]$ for training ItNet_4, which was found by pre-training. Then, $\lambda_1, \lambda_2, \lambda_3$ of ItNet-post are kept from the final values of ItNet_4, while initializing $\lambda_4 = 1.0$ and $\lambda_5 = 0.1$ for the post-training.

To improve the overall performance of our networks, we have additionally applied the following "tricks" for fine tuning, which are ordered by their importance:

(i) Due to statistical fluctuations, the networks typically exhibit slightly different reconstruction errors, despite using the same training pipeline. For the computation of our final reconstructions, we therefore *ensemble* ten networks, each trained on a different split of the training set.
(ii) Due to the training with small batch sizes, we replace batch normalization of the UNet-architecture by *group normalization* [21].
(iii) We equip the UNet-architecture with a few *memory channels*, i.e., one actually has that $\boldsymbol{U}[\boldsymbol{\theta}] \colon \mathbb{R}^N \times (\mathbb{R}^N)^{c_{\text{mem}}} \to \mathbb{R}^N \times (\mathbb{R}^N)^{c_{\text{mem}}}$ (cf. [14, 1]). While the original image-enhancement channel is not altered, the output of the additional channels is propagated through ItNet, playing the role of a hidden state (in the spirit of recurrent neural networks). For our experiments, we have selected $c_{\text{mem}} = 5$.
(iv) It was beneficial to occasionally restart the training of the networks, e.g., see Fig. 8.

The following modifications did not lead to a gain in performance and were omitted:

(i) Improving the FBP in Step 1 by making some of it components learnable (e.g., the filter), cf. [22]. Although this is advantageous for the reconstruction quality of the FBP itself, it leads to worse results for UNet and ItNet. This suggests that a combination of model- and data-based methods benefits most from precise and unaltered physical models.
(ii) Adding additional convolutional-blocks in the measurement domain of ItNet.
(iii) Modifying the standard ℓ^2-loss by incorporating the RMSE or the ℓ^1-norm.
(iv) Utilizing different optimizers such as RAdam, AdamW, SGD, or MADGRAD.

In Fig. 8, we visualize the RMSE loss curves of all phases of our complete training pipeline including optimizer restarts, i.e., UNet $\to$ ItNet_4 + restart $\to$ ItNet-post + $2 \times$ restart.

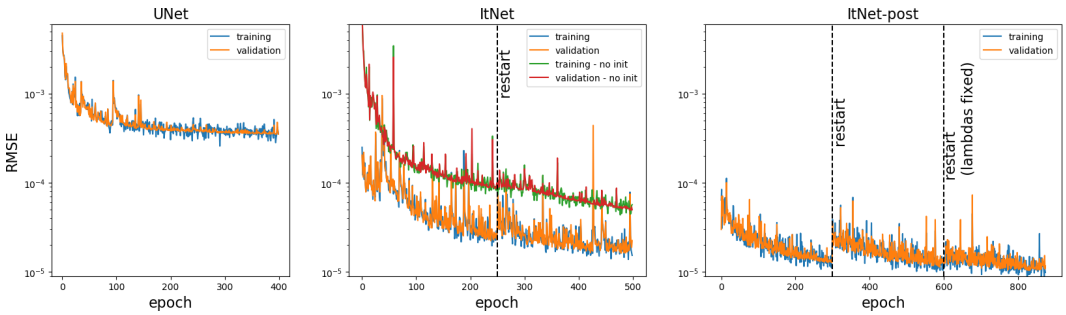

Figure 8: **Loss curves and network training.** The first two plots demonstrate that ItNet_4 improves the RMSE by approximately an order of magnitude in comparison to a post-processing by UNet. Furthermore, the gain of our UNet-initialization strategy can be seen in the second graph. The last two plots illustrate the advantages of restarting and of the post-training strategy, respectively. Note that we display the RMSE on the training and validation sets instead of the actual ℓ^2-losses, which behave similarly.