Is Deep Image Prior in Need of a Good Education?

Riccardo Barbano†,1, Johannes Leuschner†,2, Maximilian Schmidt2, Alexander Denker2, Andreas Hauptmann†,3, Peter Maaß2, Bangti Jin1

1University College London, 2University of Bremen, 3University of Oulu

{riccardo.barbano.19, b.jin}@ucl.ac.uk
{jleuschn, maximilian.schmidt, adenker, pmaass}@uni-bremen.de Andreas.Hauptmann@oulu.fi

Abstract

Deep image prior [55] was recently introduced as an effective prior for image reconstruction. It represents the image to be recovered as the output of a deep convolutional neural network, and learns the network’s parameters such that the output fits the corrupted observation. Despite its impressive reconstructive properties, the approach is slow when compared to learned or traditional reconstruction techniques. Our work develops a two-stage learning paradigm to address the computational challenge: (i) we perform a supervised pretraining of the network on a synthetic dataset; (ii) we fine-tune the network’s parameters to adapt to the target reconstruction. We showcase that pretraining considerably speeds up the subsequent reconstruction from real-measured micro computed tomography data of biological specimens. The code and additional experimental materials are available at educated-dip.github.io/docs.educated_deep_image_prior/.

1. Introduction

Inverse problems in imaging center around the recovery of an unknown image \( x \in \mathbb{R}^n \) of interest based on the given corrupted measurement \( y_\delta = Ax + \eta \), where \( y_\delta \in \mathbb{R}^m \) is the noisy measurement data, \( A \) the linear forward operator, and \( \eta \) an i.i.d. noise (e.g. Gaussian noise \( \eta \sim \mathcal{N}(0, \sigma^2 I) \)). Due to the inherent ill-posedness of the reconstruction task, suitable regularization is crucial and is key for a successful recovery of \( x \) [17, 31, 51].

Over the last years, deep learning methods have been successfully applied to solve all types of inverse problems in imaging, with supervised training being the dominating paradigm (see [3, 44] for overviews). That means, a deep neural network is trained to restore the original image from the corrupted data using a set of paired training data. Depending on the architecture of the deep learning model, a large number of such high-quality paired training data may be needed [5]. Except simulated data, these are usually not obtainable in many applications, or (too) expensive to collect. Further, there is also the challenge arising from distributional shifts in the training data (e.g. change of the noise level, a different forward operator at test time, or a different image class). Ideally, the trained model should be robust to these changes, and transfer its reconstructive properties from one domain to another using as little additional data as possible [6, 20, 40]. Unfortunately, this is often not the case.

A solution to these challenges was proposed by Ulyanov et al. [55, 56]: deep image prior (DIP) represents a new approach to regularize image reconstruction. Rather than taking the supervised route, DIP learns to both reconstruct and regularize without reference data. This is based on the assumption that a natural, or tomographic image, can be well approximated by a convolutional neural network (CNN). To achieve this, the network’s parameters are trained to generate an image that fits the measured observation \( y_\delta \). As such, the most appealing feature of DIP is that it does not need paired training data, but only a single set of test data. The approach is very attractive for imaging tasks with scarce training data, it has received enormous attention in the image reconstruction community, and it has delivered state-of-the-art performance on several imaging tasks, including CT and MRI reconstruction [5, 12], while closely matching its supervised counterparts.

While DIP has been shown to be effective, it is not completely free from drawbacks. A notable one is that DIP requires “fresh training” each time it is deployed, which leads to high computational overhead at test time when compared to supervised learned image reconstruction techniques [22, 42, 44]; the latter ones only require one feedforward pass through the network at test time. This inefficiency is considerably exacerbated by the fact that DIP is often slow (and unstable) to converge, requiring a lengthy optimization process [5, 37]. For example, reconstructing a single image of resolution \((501 \text{ px})^2\) requires approximately 30-50k iterations to reach the early-stopping point.
which translates to 3-5 hours of computing-time\(^1\). This has undeniably hindered its applicability to solve inverse problems in imaging, especially in situations where fast reconstruction is critical. These observations naturally motivate us to explore the following:

**Can DIP benefit from pretraining, and if so, how can we construct an informative synthetic dataset for pretraining so as to ameliorate the subsequent (unsupervised) reconstruction, e.g. of real \(\mu\)CT data?**

Pretraining, indeed, is one well-established paradigm to address data scarcity in supervised learning [4, 28]. Models are often pretrained using large-scale datasets (e.g. ImageNet [13]), and fine-tuned on target tasks that have less training data [16]. However, the idea of pretraining has not received the attention it deserves for DIP, and presents a new challenge. The challenge is to learn via supervised pretraining feature representations that are transferable and generalizable to subsequent fine-tuning. Our work systematically explores a supervised pretraining strategy for accelerating DIP-based tomographic image reconstruction.

**Contributions** This paper introduces a novel two-stage learning paradigm into the framework of DIP for tomographic reconstruction, which comprises a supervised stage of pretraining on a suitable synthetic dataset, setting the stage for subsequent unsupervised fine-tuning for target reconstruction tasks. Our focus boils down to two fundamental questions:

**Q1:** Is supervised pretraining with synthetic data effective in accelerating the training of DIP on subsequent reconstructive tasks?

**Q2:** How does pretraining impact the target reconstructive task?

Our contribution addresses these two questions, and can be summarized as follows:

**C1:** We develop a simple strategy to significantly accelerate the convergence of DIP on real-measured \(\mu\)CT datasets, supporting the case for DIP being recasted within the “supervised pretraining + unsupervised fine-tuning” paradigm. We showcase that a carefully designed pretraining stage with synthetic data considerably speeds up the subsequent reconstructive task: the unsupervised refinement stage can be as fast as total variation regularized iterative reconstruction.

**C2:** We conduct a thorough experimental study to shed valuable insight into the transfer between the supervised pretraining and the fine-tuning stages. We provide a novel linear analysis to study the impact of pretraining, and observe that it promotes sparsity in the parameters’ bases and hierarchical specialization of the model.

## 2. Deep Image Prior

The idea of DIP [55] is to find the minimizer of the fidelity \(\|Ax - y_\delta\|^2\), under the assumption that the unknown \(x\) is the output of a CNN, \(x = \varphi_\theta(z)\), where \(z\) is a fixed random vector, and \(\theta \in \mathbb{R}^p\) denotes the network’s parameters to be learned. The network architecture is based on U-Net [50]. DIP solves

\[
\theta^* \in \arg \min_\theta \|Az_\delta - y_\delta\|^2,
\]

and presents \(\varphi_{\theta^*}(z)\) as the recovered image. Note that the training of the network parameters \(\theta\) coincides with the recovery process, and it is repeated for each individual measurement data separately. This procedure is unsupervised, and is guided by the principle of matching the network output \(\varphi_{\theta}(z)\) to the measured data \(y_\delta\). DIP uses early-stopping to deliver a satisfactory reconstruction, as the update of \(\theta\) requires to be stopped early to avoid the deleterious effect of overfitting to noise.

## 3. Related Works

**Deep Image Prior** Since the proposal of the original DIP [55], there have been several important developments in further empowering DIP. Heckel et al. [24] propose Deep Decoder, which eases the need for early-stopping by using much fewer parameters so that the network is underparameterized, and [25] prove that it recovers smooth signals from few measurement data. Dittmer et al. [15] study DIP through the lens of regularization theory [17, 31, 51], and Cheng et al. [9] discuss its connection with Gaussian processes as the number of architecture channels grows to infinity, and propose the use of Bayesian learning (via stochastic gradient Langevin dynamics). There are several efforts to combine DIP with explicit regularization to improve the reconstruction quality. The works [5, 38] propose the use of total variation penalty for stabilizing the learning process, and the work [41] combines DIP with regularization by denoising. Ulyanov et al. [55] introduce regularization by injecting noise into the iteration, and Jo et al. [34] propose an approach to penalize the complexity of the reconstruction measured by the degrees of freedom, based on Stein’s unbiased risk estimator. The present work complements these existing studies by addressing comprehensively the computational challenge associated with DIP.

**Advances in Pretraining** Supervised pretraining on ImageNet has been common practice in computer vision, where the models are pretrained to solve image classification, and transferred to downstream tasks (e.g. object detection

---

\(^1\)The time-range is an estimate on NVIDIA GeForce RTX 2080Ti - 1080Ti.
[48, 49] and semantic segmentation [39]). Recently, He et al. [23] observe that pretraining on ImageNet does not necessarily improve the accuracy of the downstream task. Similar observations about the disputed benefit of pretraining on ImageNet are made within the context of medical image classification [46]: pretraining offers little benefit in boosting accuracy, but leads to faster training convergence. However, Hendrycks et al. show that pretraining may not improve the accuracy of the downstream task, but leads to model robustness and uncertainty calibration [26]. The aforementioned works focus on supervised learning, whereas we focus on an unsupervised learning framework. More specifically, we study pretraining with a synthetic dataset as a means for accelerating DIP reconstruction on measured μCT data, and we provide a detailed analysis of the mechanism of the pretraining strategy for acceleration.

Only recently, Cui et al. [11] use noisy PET images and their corresponding anatomical prior images (from CT or MR CT/MR images) as a dataset in the (population) pretraining phase, whilst, in our work, we focus on whether we can construct meaningful synthetic representations (i.e. pretraining with simulated noisy CT measurement data of phantom images, consisting of ellipses).

4. Proposed Method

The TV-regularized DIP approach obtains $x^*$ by

$$
\theta_s^* \in \arg\min_{\theta} \left\{ l_s(\theta) := \| A \varphi_\theta(z) - y_s \|_2^2 + \gamma \text{TV}(\varphi_\theta(z)) \right\},
$$

$$
x^* = \varphi_\theta^*(z),
$$

(1)

where $\varphi_\theta$ is a CNN, $z \in \mathbb{R}^n$ is a fixed input image, which is commonly chosen to be pixel-wise i.i.d. samples of random noise, and $\gamma \geq 0$ balances the data consistency with the regularization term $\text{TV}(\varphi_\theta(z))$, which denotes the total variation seminorm on the network output $\varphi_\theta(z)$. Several studies [5, 38] have found that the incorporation of the total variation penalty is beneficial to DIP. The loss $l_s$ in (1) is optimized with a variant of stochastic gradient descent (i.e. Adam [36]), starting with a default random initialization of the network parameters $\theta$. The learning is performed as (single-batch) test time adaptation via (1). The optimization can be stopped early, which is crucial if $\gamma = 0$ in order to avoid overfitting to the noise in the data $y_s$ [55].

In this work, we recast DIP reconstruction into the “supervised pretraining + unsupervised fine-tuning” paradigm as a two-stage process, which is termed as educated DIP (EDIP). In the first stage, we pretrain the network $\varphi_\theta(A^s y_s)$, where $A^s$ is an approximate reconstruction operator (e.g. filtered back-projection (FBP) for CT [45]). The training is carried out on a synthetic dataset $D = \{(x^n, y^n_s)\}_{n=1}^N$, composed of $N$ pairs drawn from the joint distribution of ground truth $x^n$ and corresponding simulated measurement $y^n_s$. This procedure is tailored to the target reconstruction task in (1); and learns the optimal parameters $\theta_s^*$ via supervised training by minimizing the following loss:

$$
\theta_s^* \in \arg\min_{\theta} \left\{ l_s(\theta) := \frac{1}{N} \sum_{(x^n, y^n_s) \in D} \| \varphi_\theta(A^s y^n_s) - x^n \|_2^2 \right\}. \quad (2)
$$

Note that the network $\varphi_\theta$ receives the initial guess $A^s y^n_s$ as its input (instead of the random noise image in the original DIP [55]), serving as a post-processing reconstructor [33]. In this way, we enforce “benign” inductive biases via supervised learning. This educates DIP with “human-guided” knowledge contained in the simulated dataset $D$, which is then exploited, but still needs to be amended, in solving the reconstruction task in (1).

In the second stage, for a given new query measurement $y_s$, we use the optimal parameters $\theta_s^*$ from the pretraining stage to initialize the network $\varphi_\theta(A^s y_s)$ in (1) so as to get DIP up to speed in handling high-dimensional target reconstruction tasks from real-measured data. In other words, we regard the DIP optimization as a self-adaptation stage, where the network’s parameters $\theta$ are fine-tuned in an unsupervised manner, yet their drift is conditioned on $\theta_s^*$ by initialization. Note that the robustness of the proposed method does not rely solely on how well the pretraining stage anticipates distributional shifts, when it comes to reconstructing from measurement data. The model makes a good use of pretraining — the supervised pretraining sets the stage — but ultimately adapts to distributional shifts at test time, and reserves its right to amend the supervision received during the pretraining. This is crucial as illustrated by the experimental evaluation.

There are several possible variants of the basic framework. U-Net consists of two parts, a decoder component with parameters $\theta_{\text{dec}}$, and an encoder component with parameters $\theta_{\text{enc}}$ (cf. Figure 9 in Appendix B.2 for a schematic illustration). As a direct variant of EDIP, we consider the case where only the parameters $\theta_{\text{dec}}$ of the decoder are fine-tuned, whilst the encoder parameters, which are regarded as a shared (between stages) feature extractor, are kept to the initial educated guess $\theta_s^*$. At test time, we solve (1) only with respect to $\theta_{\text{dec}}$ and rely on the pretraining to construct a suitable “universal” encoding. The learned reconstructor $\varphi_{\theta_s^*}$, reconstructs from the measurement data with $\theta = (\theta_{\text{enc}}^*, \theta_{\text{dec}}^*)$. This variant with the fixed encoder (FE) is termed as EDIP-FE.

5. Datasets

We use micro CT (μCT) data for evaluation.

5.1. Synthetic Training Dataset

We pretrain on a synthetic training dataset (i.e. ellipses images). Images composed of ellipses with random po-
sition, shape, rotation and intensity values are commonly used to train and evaluate learned reconstruction methods. We use datasets of 32,000 training and 32,000 validation images generated on-the-fly using ODL [1]. The image resolution and the distribution of the ellipses can be easily adapted to match different target data (e.g. with different reconstruction diameters).

5.2. μCT Measurement Data

We evaluate our approach on two real μCT measurement datasets. In each case, the forward operator $A$ used for both the pretraining and the DIP stage is a ray transform matching the μCT geometry of the measurements. See Appendix A for further details on the scan geometries.

Tomographic X-ray Data of a Lotus Root CT measurements of a Lotus root slice filled with different materials are available from [8] as a part of a series of open X-ray tomographic datasets. It contains fan-beam measurements corresponding to a 2D volume slice, consisting of 120 projections at angles equally distributed over $[0, 360^\circ]$ with 429 detector pixel values each. A sparse matrix modeling the forward operator for an image resolution of $(128 \text{ px})^2$ is provided with the dataset, which is used in the experiments.

We consider a 6-fold angular sub-sampling in the experimental evaluation: 20 angles, equally distributed over $[0, 360^\circ]$ in steps of $18^\circ$. We denote this setting as Sparse 2.0. As a gold-standard reference, we use a TV-regularized reconstruction from all 120 projection angles, obtained using the Adam optimizer (see Table 3 in Appendix B.3 for hyperparameters).

X-ray Walnut Dataset A collection of cone-beam CT measurement data from 42 Walnuts [14] was released by researchers from the Centrum Wiskunde & Informatica, the Netherlands. For each walnut, a set of three 3D cone-beam measurements is included, each obtained with a different source position. Projections are acquired at 1200 angles equally distributed over $[0, 360^\circ]$ in steps of $3^\circ$. Approximate reconstructions $A^\dagger y_A$ are computed via the Feldkamp-Davis-Kress (FDK) algorithm [18] (see Appendix A.2 for details). FDK is an FBP-based algorithm that includes a weighting step for cone-beam measurements, and is also referred to as “FBP” below. As a ground truth, we use the corresponding 2D slice from the 3D ground truth reconstruction provided with the dataset [14], which was obtained with accelerated gradient descent incorporating the measurements from all 1200 projection angles and all three source positions.

6. Experiments

6.1. Evaluation Metrics

We assess the quality of the reconstructed image via peak signal-to-noise ratio (PSNR) [27]. We include the structural similarity index measure (SSIM) [58, 59] for the reconstructed images. To assess the speed of convergence, we employ two simple metrics: steady PSNR and rise time. The steady PSNR is the median PSNR over the last 5k iterations of the fine-tuning stage. The rise time is the iteration number at which we reach the baseline PSNR (i.e. DIP’s steady PSNR) up to a threshold, which is set to 0.1 dB.

In addition, we always consider the iteration-wise median PSNR over 5 runs of the same experiment (with varying seeds) for the above metrics. Note that the variability between runs does not only arise from random initialization of the network parameters or noise input, but also from numerical effects in parallel computations on GPU, which still occur for the otherwise deterministic EDIP method with FBP input.

6.2. Validation Procedure

We design a workable heuristic for identifying the parameters’ configuration to be used at test time, by comparing three different pretraining runs (varying the random seed). For each run, we collect checkpoints along the optimization trajectory after every 20 epochs, and select the parameters’ configuration with the best performance on a validation set. For validation, we design a reconstructive task based on the Shepp-Logan phantom [52], a standard test image created to assess reconstruction algorithms. The phantom is by construction within the ellipses data manifold and shares the same noise distribution of ellipses measurements. The checkpoint leading to the shortest rise time is selected, among those with a steady PSNR that is at most 0.25 dB lower than the maximum reached steady PSNR.

We verify whether the checkpoint identified on the validation data lead also to the best rise time on the test data; we observe minor variability within validation and test data, yet we select one of the best two runs (see Figure 19, and Fig-

---

\(^2\)https://github.com/cicwi/WalnutReconstructionCodes
6.3. Hyperparameter Selection

The learning rate and the regularization parameter $\gamma$ are fine-tuned for the standard DIP (i.e. DIP fed with a fixed noise image as input) for test and validation. For a fair comparison, when learning EDIP, we do not conduct an additional hyperparameter search, although it can potentially be beneficial. We keep the hyperparameters that are optimal for standard DIP. See Table 1 in Appendix B.3 for the selected hyperparameter values.

6.4. The Lotus Root

Figure 1 shows the convergence properties of EDIP (pretrained on ellipses) and DIP for Lotus Sparse 20. We include in our analysis the cases where the FBP (i.e. $A^\dagger Y_S$) is fed as the input (instead of noise) when solving (1) for the standard DIP, as well as inputting noise (instead of the FBP) for EDIP. EDIP significantly outperforms DIP in terms of the convergence speed for either a fixed noise image or the FBP. It only takes 195 (and 723 if fed with a noise image) iterations to reach $-0.1 \text{ dB}$ of the baseline PSNR, against 4.1k iterations needed for the standard DIP. The optimization process is considerably more stable: the variation of the loss of EDIP is substantially smaller (throughout all the optimization) than that with DIP, implying a possibly more favorable loss landscape for EDIP. We also observe that fixing the encoder to the pretrained encoder parameters $\theta^*_{s,\text{enc}}$ (i.e. EDIP-FE) is as fast as EDIP. Figure 2 shows the reconstructed Lotus (along with its gold-standard reference and FBP) for Lotus Sparse 20. Moreover, EDIP considerably overshoots the baseline PSNR, suggesting that pre-training, if coupled with early-stopping (approximately a few hundred iterations after the rise time), would lead to better reconstructions. A substantial overshoot of the baseline PSNR is also observed on the Shepp-Logan (cf. Figure 6, and Figure 18 in Appendix D), possibly due to the in-distribution nature of the phantom with respect to the ellipses dataset. Pretraining can boost the performance of DIP, which differs from the findings for object detection [23] and classification [46].

6.5. The Walnut

Figure 3 shows the convergence properties of EDIP (pretrained on ellipses) and DIP for the Walnut Sparse 120, and Figure 4 shows the reconstructed Walnut. It is observed that finer structures (e.g. the wrinkled shell) are better reconstructed with EDIP (see insets); DIP (at least qualitatively) suffers from over-smoothing artifacts. The speed-up on EDIP (fed with the FBP) is, indeed, consistent with the observation for the Lotus root, albeit to a less dramatic degree. EDIP takes approximately 30 min at rise time (approx. 4.4k iterations), against DIP, which takes 2 h and 30 min at rise time (approx. 20.4k iterations) with NVIDIA GeForce RTX 2080Ti. A TV regularized reconstruction of the Walnut takes approximately 6 min, and requires 1.7k gradient steps to converge to 31.67 dB. EDIP takes only 3 min (i.e. approximately 421 iterations) to match 31.67 dB. In 6 min, EDIP reaches 32.80 dB, that corresponds to a gain of 1.1 dB. Finally, we observe that DIP-FE / EDIP-FE report similar performances to DIP / EDIP. Besides raising interesting questions about...
the role of the encoder, this result opens up to even faster computing time (with fewer parameters to update).

In Need to Amend  We investigate the effect of pretraining on performing reconstructions on the test data and how the adaptation stage further improves the quality of the reconstructed image. To assess to what extent the knowledge enforced through the synthetic dataset needs to be amended during the fine-tuning stage, we investigate what gets transferred. First, we observe that feature reuse plays a very important role. By initializing the network’s parameters to a pretrained configuration, we observe that the deployment of the pretrained model on an out-of-distribution task (e.g. on the Walnut) shows high input-robustness (i.e. good reconstructive properties) [26]. However, the feature reuse mechanism leads to hallucinatory behaviors (i.e. being biased too much towards the synthetic dataset used in the pretraining stage). Amending the knowledge acquired via pretraining protects from instabilities (e.g. hallucinations) due to distributional shifts (e.g. in the image class data); all are recognized drawbacks of supervised learned reconstructors [2]. Figure 2 and Figure 4 show this property: the initial reconstruction by a direct deployment of the pretrained network shows overly-smooth images, with the Lotus being seriously affected by ellipses-like artifacts — the network restores the FBP of the Lotus Sparse 20 by patching it up with ellipses. The artifacts are then removed during the fine-tuning stage. Similarly, the reconstruction of the Walnut regains more realistic texture through the fine-tuning stage (cf. Figure 4).

7. Diagnostic Study

In this section, we shed insight about what is being transferred from the pretraining to the target DIP reconstruction.

Investigating Feature Reuse  In a similar spirit to [43], we feed a noise image to EDIP (trained on pairs of FBP and ground truth image). By doing so, we make any vi-

---

Figure 3. **PSNR convergence of EDIP pretrained on the ellipses dataset compared to standard DIP on Walnut Sparse 120 measurement data.** All traces are the mean PSNR of 5 repetitions of the same experimental run (varying the random seed); the standard deviation is also reported. See Appendix C.2 for complementary tabular results.

Figure 4. **Walnut reconstruction of EDIP (pretrained on ellipses dataset), compared to standard DIP.** From the 5 runs (varying the seed), the one with the (closest to) median PSNR was selected for each method, except for EDIP (FBP) iter. 4500, which is taken from an additional run.

Figure 5. **Iterates collected throughout the Lotus reconstruction, comparing EDIP (pretrained on ellipses dataset) and standard DIP on Sparse 20 measurement data.** A video showing the reconstruction process is included in [7] and available at https://educateddip.github.io/docs.educated_deep_image_prior/. See Appendix C.2 for an analogous analysis of the Walnut data.
ual features learned in the pretraining stage completely useless. This is to shed insights into the factors that come into play when adapting the model to downstream test data. We observe faster convergence with respect to the standard DIP for the Lotus dataset (see Figure 1). EDIP (fed with FBP) still results in faster convergence, showing consistency with the intuition that decreasing feature reuse leads to diminishing benefits. Figure 5 shows that EDIP remolds the noise image differently compared to the standard DIP. The inductive biases learned at the pretraining stage prioritize a reshape of the noise image as ellipse-like structures: the model makes an educated reconstruction relying on the data-generating process and the space of solutions learned in the pretraining stage. This suggests that the model is learning features that are invariant of the input; other factors (e.g., lower distributional statistics) from the pretrained parameters are also helping the adaptation. It is then re-adapted (and improves over the initial reconstructive properties) by enforcing data-consistency via formulation (1).

On the Walnut, the benefit of pretraining seems (at least quantitatively) not to be as dramatic as for the Lotus, if a noise image input is used. Figure 14 in Appendix C.2 shows that EDIP still gives us some qualitative advantages over DIP. We hypothesize that the problem geometry of the Lotus Sparse 20 has induced more generalizable feature representations. In fact, the difficulty of the inverse problem has prevented the model from overfitting on “dataset-specific knowledge". Meanwhile, the Walnut reconstruction can be considered more of a post-processing task.

There is, indeed, room for overfitting to the image class, and such dataset-specific knowledge is usually detrimental to the transfer.

Getting It Right: Checkpoint Selection in Pretraining
Motivated by the speed-up obtained by pretraining DIP, we show that starting the fine-tuning from different pretraining runs (i.e. different starting points in the parameter space), and especially from the earlier checkpoints of the same run, has a significant effect on the adaptation speed (see Figure 6). We observe that it is essential to carefully select the start of the fine-tuning from later checkpoints. Figure 6 reports the above experimental findings: (i) pretraining runs (by just varying the seed) lead to different adaptation profiles (ii) pretraining for more epochs (20 vs. 100) leads to a faster adaptation (particularly in the case of in-distribution test data). A validation procedure that identifies the starting parameters’ configuration is needed to maximize transferable performance (e.g., convergence speed-up). In addition, we believe checkpoint selection plays an important role, and should receive the attention it deserves.

3See Figure 10 in Appendix C.1 and Figure 15 in Appendix C.2 for a direct comparison between the Lotus Sparse 20 and Walnut Sparse 120 settings.

**Spectral Evaluation of Pretraining**

We propose a spectral analysis to investigate the impact of a “good education”. We construct an approximate spectral decomposition of the Jacobian matrix of the network output (including the forward map $A$) with respect to the network parameters $\theta$, that is, $A\partial_\theta \phi/\partial \theta \in \mathbb{R}^{m \times p}$. We then consider the subspace spanned by the leading right singular vectors (corresponding to the largest singular values) as a faithful representation of the network’s parameter space, which determines the dynamics of the training process. This is equivalent to a linearization of the model in the over-parameterized regime (e.g. neural tangent kernel [32]). Due to the high-dimensionality of the output space, likewise, the parameter space, a direct computation of the full Jacobian matrix is computationally intractable. We resort to an approximation of the first $\ell$ singular vectors via a variant of the randomized singular value decomposition [21, 54]. See Appendix E.1 for a detailed description of the randomized algorithm. In the analysis, we limit ourselves to the first 1000 singular vectors (i.e. $\ell = 1000$).

We perform the analysis on EDIP and DIP, both receiving the FBP as the input, and respectively approximate the singular vectors of the Jacobian matrix, evaluated at three checkpoints (i.e. three different parameter settings) during the adaptation stage ($\theta^{\text{init}}, \theta^{[100]}, \theta^{\text{conv}}$). Figure 7 summarizes our empirical findings on the right singular vectors component-wise plots as well as the Hoyer measure of sparsity [29, 30]. This measure assumes a value of 0 whenever the vector is dense (i.e. all components are equal and non-

![Figure 6. Validation runs on the Shepp-Logan phantom for selecting the initial EDIP (FBP) model parameters pretrained on the ellipses dataset for measurement data in the Lotus Sparse 20 geometry. The model from training run 2 after 100 epochs is selected because it has the shortest rise time (with a sufficiently high steady PSNR). The complete figure is reported in Appendix D.](image-url)
Development of singular vectors of the forward map linearization w.r.t. the parameters $\theta$ for EDIP (FBP), pretrained on images of ellipses, compared to DIP (noise) on Lotus Sparse 20 measurement data. Histograms (a) and (b) show mean histograms for the singular vectors $v_1, \ldots, v_{20}$ and $v_{976}, \ldots, v_{995}$, respectively; denoting in brackets the Hoyer measure of sparsity [29, 30], which assumes a value of 0 whenever the vector is dense (i.e. all components are equal and non-zero), and 1 whenever the vector is sparse (i.e. only one component is non-zero). Refer to Appendix E.1 for the plot of the respective singular values.

For DIP, we observe that the singular vectors are equally distributed throughout the parameter space (at $\theta^{\text{init}}$) and across all frequencies (for different singular values). During the adaptation stage, we observe a “relevance shift” towards the decoder’s parameters (at $\theta^{100}$ and at $\theta^{\text{conv}}$, respectively). This is not too surprising since the heavy-lifting of representing the target image is actually done by the decoder. In addition, it is also consistent with our experimental findings: EDIP-FE (where the encoder is fixed) shows similar reconstruction properties to EDIP (see Figure 1 and Figure 3). We assess the criticality (i.e. the influence of the drift of the output with respect to the parameters) of different architectural parts and observe that the decoder layers (i.e. $\theta^{\text{dec}}$) become more critical. For EDIP, we also observe that pretraining enforces a hierarchical structure (i.e. a relevance shift towards the decoder’s parameters), and again sparsity is clearly observed after pretraining. Pretraining strongly promotes sparsity in the parameter space. Furthermore, the sparsity enforced via pretraining is not lost in the adaptation stage, which is actually further promoted. Note that pretraining induces a shift in the singular values spectrum (cf. Figure 22 in the Appendix E.1), and the overall behavior does not vary much during the adaptation of the pretrained model. In contrast, for DIP, the shift is quite dramatic in terms of the magnitude. In particular, the number of singular values larger than a given (plausible) threshold changes consistently. A similar observation was also made in [53] in the context of object detectors (but based on the normalized eigenspectrum of the Hessian). This observation may explain the different dynamics of the optimization scheme for the pretrained model and the model trained from scratch.

8. Discussions and Conclusions

Our work advances deep learning-based tomographic reconstruction in the context of unsupervised learning, and the proposed method leads to a substantial improvement of DIP-based reconstructive frameworks. We develop a two-phase learning paradigm for accelerating DIP in image reconstruction. The proposed method consists of a supervised pretraining phase on a simulated dataset to educate DIP and then a fine-tuning phase which adapts the network parameters to a single test image. The extensive experimental evaluation clearly shows that pretraining on simulated data can significantly speed up, and stabilize DIP reconstruction for real-measured sparse-view CT. The empirical study also indicates that the pretraining phase can facilitate learning the feature representation, and that adapting only the decoder’s parameters during the fine-tuning phase is also sufficient to ensure good reconstruction accuracy. The spectral analysis of the linearized model indicates a strong correlation of the sparsity pattern with the pretraining, and a drastically different shift of the singular values spectrum for the standard DIP and its educated version.
Acknowledgements

R.B. was supported by the i4health PhD studentship (UK EPSRC EP/S021930/1), and by The Alan Turing Institute under the EPSRC grant EP/N510129/1. J.L., M.S., and A.D. were funded by the German Research Foundation (DFG; GRK 2224/1). J.L. and M.S. additionally acknowledge support from the DELETO project funded by the Federal Ministry of Education and Research (BMBF, project number 05M20LBB). A.D. further acknowledges support from the Academy of Finland projects 338408, 336796, 334817. P.M. was supported by the Sino-German Center for Research Promotion (CDZ) via the Mobility Programme 2021: Inverse Problems – Theories, Methods and Implementations (IP–TMI). The research of B.J. is supported by UK EPSRC grant EP/N510129/1. J.L., M.S., and A.D. were funded by the German Research Foundation (DFG; GRK 2224/1). J.L., M.S., and A.D. were supported by the Klaus Tschira Stiftung via the project MALDIS (project number 05M20LBB). A.D. further acknowledges support from the German Federal Ministry of Education and Research (BMBF, project number 00.010.2019). A.H. acknowledges funding from the Academy of Finland projects 338408, 336796, 334817. P.M. was supported by the Sino-German Center for Research Promotion (CDZ) via the Mobility Programme 2021: Inverse Problems – Theories, Methods and Implementations (IP–TMI). The research of B.J. is supported by UK EPSRC grant EP/T000864/1.

References

[1] Jonas Adler, Holger Kohr, Axel Ringh, Julian Moosmann, Sebastian Banert, Matthias J. Ehrhardt, Gregory R. Lee, Kati Niinimäki, Barbara Gris, Olivier Verdier, Johan Karlsson, Gustav Zickert, Willem Jan Palenstijn, Ozan Öktem, Chong Chen, Hector Andrade Lourca, and Michael Lohmann. Operator Discretization Library (ODL), 2018. Zenodo.

[2] Vegard Antun, Francesco Renna, Clarice Poon, Ben Adcock, and Anders C Hansen. On instabilities of deep learning in image reconstruction and the potential costs of AI. Proc. National Academy Sci., 117(48):30088–30095, 2020.

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

[4] Shekoofeh Azizi, Basil Mustafa, Fiona Ryan, Zachary Beaver, Jan Freyberg, Jonathan Deaton, Aaron Loh, Alan Karthikesalingam, Simon Korblinth, Ting Chen, et al. Big self-supervised models advance medical image classification. Preprint, arXiv:2101.05224, 2021.

[5] Daniel Otero Baguer, Johannes Leuschner, and Maximilian Schmidt. Computed tomography reconstruction using deep image prior and learned reconstruction methods. Inverse Problems, 36(9):094004, 2020.

[6] Riccardo Barbano, Zeljko Kereta, Andreas Hauptmann, Simon R. Arridge, and Bangti Jin. Unsupervised knowledge transfer for learned image reconstruction. Preprint, arXiv:2107.02572, 2021.

[7] Riccardo Barbano and Johannes Leuschner. Supplementary Material for "Is Deep Image Prior in Need of a Good Education?". 2021. Zenodo.

[8] Tatiana A. Bubba, Andreas Hauptmann, Simo Huotari, Juho Rimpelaäinen, and Samuli Siltanen. Tomographic X-ray data of a lotus root filled with attenuating objects. Preprint, arXiv:1609.07299, 2016.

[9] Zezhou Cheng, Matheus Gadelha, Subhransu Maji, and Daniel Sheldon. A Bayesian perspective on the deep image prior. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 5443–5451, 2019.

[10] Kenneth Clark, Bruce Vendt, Kirk Smith, John Freymann, Justin Kirby, Paul Koppel, Stephen Moore, Stanley Phillips, David Maffitt, Michael Pringle, Lawrence Tarbox, and Fred Prior. The Cancer Imaging Archive (TCIA): Maintaining and operating a public information repository. J. Digital Imag., 26(6):1045–1057, 2013.

[11] Jianan Cui, Kuang Gong, Ning Guo, Chenxi Wu, Kyungsang Kim, Huafeng Liu, and Quanzheng Li. Populational and individual information based PET image denoising using conditional unsupervised learning. Phys. Med. & Biol., 66(15):155001, 2021.

[12] Mohammad Zalbagi Darestani and Reinhard Heckel. Accelerated MRI with un-trained neural networks. IEEE Trans. Computational Imaging, 7:724–733, 2021.

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In IEEE Conference on Computer Vision and Pattern Recognition, pages 248–255. Ieee, 2009.

[14] Henri Der Sarkissian, Felix Lucka, Maureen van Eijatten, Giulia Colacisco, Sophia Bethany Coban, and K. Joost Batenburg. Cone-Beam X-Ray CT Data Collection Designed for Machine Learning: Samples 1-8, 2019. Zenodo.

[15] Sören Dittmer, Tobias Kluth, Peter Maass, and Daniel Otero Baguer. Regularization by architecture: A deep prior approach for inverse problems. J. Math. Imag. Vision, 62(3):456–470, 2020.

[16] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In International Conference on Machine Learning, pages 647–655, 2014.

[17] Heinz W. Engl, Martin Hanke, and Andreas Neubauer. Regularization of Inverse Problems. Kluwer, Dordrecht, 1996.

[18] L. A. Feldkamp, L. C. Davis, and J. W. Kress. Practical cone-beam algorithm. J. Opt. Soc. Am. A, 1(6):612–619, 1984.

[19] Elizabeth R. Gerstner, Zheng Zhang, James R. Fink, Mark Muzi, Lucy Hanna, Erin Greco, Melissa Prah, Kathleen M. Schmainda, Akiva Mintz, Lale Kostakoglu, Edward A. Eikman, Benjamin M. Ellingson, Eva-Maria Ratai, A. Gregory Sorensen, Daniel P. Barboriak, and David A. Mankoff. Acrin 221: Assessment of tumor hypoxia in newly diagnosed glioblastoma using 18F-FMISO PET and MRI. Clinical Cancer Research, 22:5079–5086, 2016.

[20] Davis Gilton, Gregory Ongie, and Rebecca Willett. Model adaptation for inverse problems in imaging. IEEE Trans. Comput. Imag., 7:661–674, 2021.

[21] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. SIAM Rev., 53(2):217–288, 2011.

[22] Andreas Hauptmann, Felix Lucka, Marta Betcke, Nam Huynh, Jonas Adler, Ben Cox, Paul Beard, Sebastien Ourselin, and Simon Arridge. Model-based learning for ac-
celerated, limited-view 3-d photoacoustic tomography. *IEEE Trans. Med. Imag.*, 37(6):1382–1393, 2018. 1

[23] Kaiming He, Ross Girshick, and Piotr Dollar. Rethinking imagenet pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4918–4927, 2019. 3, 5

[24] Reinhard Heckel and Paul Hand. Deep decoder: Concise image representations from untrained non-convolutional networks. In *International Conference on Learning Representations 2019*, page arXiv:1810.03982, 2019. 2

[25] Reinhard Heckel and Mahdi Soltanolkotabi. Compressive sensing with untrained neural networks: Gradient descent finds the smoothnest approximation. In *Proceedings of the 37th International Conference on Machine Learning, PMLR 119*, pages 4149–4158, 2020. 2

[26] Dan Hendrycks, Kinmin Lee, and Mantas Mazeika. Using inverse problems in imaging. *IEEE Trans. Image Process.*, 26(9):4509–4522, 2017. 3

[27] Alain Hore and Djemel Ziou. Image quality metrics: PSNR vs. SSIM. In *20th International Conference on Pattern Recognition*, pages 2366–2369, 2010. 4

[28] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. *Preprint, arXiv:1801.06146*, 2018. 2

[29] Patrik O. Hoyer. Non-negative matrix factorization with sparseness constraints. *J. Mach. Learn. Res.*, 5:1457–1469, 2004. 7, 8

[30] Niall Hurley and Scott Rickard. Comparing measures of sparsity. *IEEE Trans. Inform. Theory*, 55(10):4723–4741, 2009. 7, 8

[31] Kazufumi Ito and Bangti Jin. *Nonsmooth Analysis and Differential Inclusions*. World Scientific, Hackensack, NJ, 2009. 7, 8

[32] Arthur Jacob, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, pages 8850–8859, 2018. 7

[33] Kyong Hwan Jin, Michael T. McCann, Emmanuel Froustey, and Michael Unser. Deep convolutional neural network for inverse problems in imaging. *IEEE Trans. Image Process.*, 26(9):4509–4522, 2017. 3

[34] Yeonsik Jo, Se Young Chun, and Jonghyun Choi. Re-thinking deep image prior for denoising. *ICCV 2021*, arXiv:2108.12841, 2021. 2

[35] Paul Kinahan, Mark Muzi, Brian Bialecki, and Laura Coombs. Data from ACRIN-FMISO-brain, 2018. *The Cancer Imaging Archive*. 20

[36] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 3

[37] Johannes Leuschner, Maximilian Schmidt, Pouliami Somanya, Vladyslav Andriashev, Sophia Bethany Coban, Alexander Denker, Dominik Bauer, Amir Hadjifaradj, Kees Joost Batenburg, Peter Maass, and Maureen van Eijnatten. Quantitative comparison of deep learning-based image reconstruction methods for low-dose and sparse-angle CT applications. *Journal of Imaging*, 7(3):44, 2021. 1

[38] Jiaming Liu, Yu Sun, Xiaojian Xu, and Ulugbek S Kamilov. Image restoration using total variation regularized deep image prior. In *ICASSP 2019*, 2019. 2, 3

[39] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3341–3440, 2015. 3

[40] Sebastian Lunz, Andreas Hauptmann, Tanja Tarvainen, Carola-Bibiane Schönlieb, and Simon Arridge. On learned operator correction in inverse problems. *SIAM J. Imag. Sci.*, 14(1):92–127, 2021. 1

[41] Gary Mataev, Peyman Milanfar, and Michael Elad. DeepRED: Deep image prior powered by RED. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, Oct 2019. 2

[42] Vishal Monga, Yuelong Li, and Yonina C. Eldar. Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Proc. Mag.*, 38(2):18–44, 2021. 1

[43] Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is being transferred in transfer learning? In Hugo Larochelle, Marc’Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, 2020*. 6

[44] Gregory Ongie, Ajil Jalal, Richard G Baraniuk, Christopher A Metzler, Alexandros G Dimakis, and Rebecca Willett. Deep learning techniques for inverse problems in imaging. *IEEE J. Sel. Areas Inform. Theory*, 1(1):39–56, 2020. 1

[45] Xiaochuan Pan, Emil Y. Sidky, and Michael Vannier. Why do commercial CT scanner still employ traditional, filtered back-projection for image reconstruction? *Inverse Problems*, 25(12):123009, 36, 2009. 3

[46] Maithra Raghu, Chiyuan Zhang, Jon Kleinberg, and Samy Bengio. Transfusion: Understanding transfer learning for medical imaging. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 3347–3357, 2019. 3, 5

[47] Eva-Maria Ratai, Zheng Zhang, James Fink, Mark Muzi, Lucy Hanna, Erin Greco, Todd Richards, Daniel Kim, Ovidiu C. Andronesi, Akiva Mintz, Lale Kostakoglou, Melissa Prah, Benjamin Ellingson, Kathleen Schmainda, Gregory Sorensen, Daniel Barbioriak, David Mankoff, Elizabeth R. Gerstner, and A. C. R. I. N. 6684 trial group. Acrin 6684: Multicenter, phase II assessment of tumor hypoxia in newly diagnosed glioblastoma using magnetic resonance spectroscopy. *PloS One*, 13(29902200):e0198548–e0198548, 2018. 20

[48] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 779–788, 2016. 3
[49] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *Advances in Neural Inf. Process. Syst.*, pages 91–99, 2015. 3

[50] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015. 2

[51] Otmar Scherzer, Markus Grasmair, Harald Grossauer, Markus Haltmeier, and Frank Lenzen. *Variational Methods in Imaging*. Springer, New York, NY, 2009. 1, 2

[52] Lawrence A Shepp and Benjamin F Logan. The fourier reconstruction of a head section. *IEEE Transactions on nuclear science*, 21(3):21–43, 1974. 4

[53] Yosuke Shinya, Edgar Simo-Serra, and Taiji Suzuki. Understanding the effects of pre-training for object detectors via eigenspectrum. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 8

[54] Joel A Tropp. Randomized algorithms for matrix computations. Caltech CMS Lecture Notes 2019-01, Pasadena, July 2019. 7, 19

[55] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9446–9454, 2018. 1, 2, 3

[56] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. *Int. J. Comput. Vis.*, 128(7):1867–1888, 2020. 1

[57] Wim van Aarle, Willem Jan Palenstijn, Jan De Beenhouwer, Thomas Altantzis, Sara Bals, K. Joost Batenburg, and Jan Sijbers. The ASTRA Toolbox: A platform for advanced algorithm development in electron tomography. *Ultramicroscopy*, 157:35–47, 2015. 4

[58] Zhou Wang and Alan C Bovik. A universal image quality index. *IEEE Signal Proc. Lett.*, 9(3):81–84, 2002. 4

[59] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Proc.*, 13(4):600–612, 2004. 4
Appendix A. CT Measurement data

A.1. 2D Cone-Beam Geometry

In all our experiments a 2D cone-beam geometry (cf. Figure 8) is used for modeling the forward operator. The scanner rotates around the object, taking projections from different source angles $\lambda$. Within each projection, each detector pixel (e.g. parameterized by $\gamma$) measures the intensity for a specific line, attenuated by the object.

![2D cone-beam geometry sketch](image)

On the Lotus root, the sparse matrix provided with the dataset is employed. On the Walnut data, a sparse matrix resembling the 2D cone-beam projection is constructed from the ASTRA geometry provided with the dataset, by selecting a single volume slice and a suitable subset of the 3D cone-beam projection lines. Note that for the Walnut data the rotation axis is slightly tilted, and the resulting geometry differs from the standard 2D fan-beam setting. In addition, the voxels/pixels are weighted differently in the 3D vs. 2D projections, because in the integration of the beams for each detector “pixel”, the contributing area/interval is spreading in two vs. one dimension(s) with increasing distance from the source. The beam density decreases antiproportionally to the squared distance vs. antiproportionally to the distance.

A.2. X-ray Walnut Details

From the 42 Walnuts provided in the dataset, we consider measurements of Walnut 1 taken with source position (or orbit) 2. The slice with offset $+3$ px from the middle slice (i.e. zero-based index 253) is selected for the 2D reconstruction task. A subset of projection values is determined from the provided ASTRA geometry by computing the 3D forward projection of a mask, containing ones for the selected 2D slice and zeros for all other voxels. We then choose one single detector row per column and angle with maximum intensity (i.e. argmax along the detector row axis of the projection of the volume slice mask). A sparse matrix representing the forward projection operator is constructed by calling the ASTRA forward projection routine for each unit vector. This is done to obtain an exact adjoint of the Jacobian, which is needed for the gradient computation in (1). While the more efficient back-projection routine of ASTRA could be employed, it is not directly applicable given our pseudo-2D geometry: some of the excluded detector rows close to the selected ones contribute to the selected 2D slice in the back-projection. As a simple workaround, the measurement values from the selected detector rows can be copied to the neighboring rows (i.e. edge-mode padding). We use the padding approach to compute the approximate FDK reconstructions. For computing the gradient of the data discrepancy term in (1), we observe that using the padding followed by the back-projection via ASTRA, would lead to degraded results, so we use the sparse matrix multiplication instead, which yields accurate gradients.

The implementation as well as the saved sparse matrix are available in the Github repository and supplementary material [7], respectively.

Appendix B. Methodology

B.1. The Loss

Our DIP implementation uses the loss function

$$l'_t(\theta) := \frac{1}{m} \| A\varphi_\theta(z) - y_\delta \|^2_2 + \gamma' TV(\varphi_\theta(z)),$$

where $m$ is the number of detector pixels (length of $y_\delta$) and $\nabla_h$ and $\nabla_v$ are the discrete difference operators in the horizontal and vertical directions, respectively. This loss is equivalent to $l_t(\theta)$ in (1), with the total variation penalty chosen to be anisotropic.

For notational brevity, we denote the type of the network input $z$ used for a method in round brackets: for example, “DIP (noise)” refers to the standard DIP with noise input, while “EDIP (FBP)” stands for the educated DIP with FBP input.

B.2. Network Architecture

Figure 9 shows the network architecture used. We adopted the architecture proposed by [5], with the only difference being that we replace batch-normalization layers with group-normalization layers.

B.3. Hyperparameter Search

For each setting, suitable hyperparameters for standard DIP (noise) are selected by grid search. While the learning rate $1e^{-4}$ is (near) optimal in all cases, the TV-
regularization parameter $\gamma'$ varies both with the CT geometry and between validation data (i.e. Shepp-Logan phantom, simulated measurement data) and test data (i.e. Lotus or Walnut, real measurement data).

The hyperparameters used for DIP and EDIP are listed in Table 1. For only two cases, we observe the hyperparameters that are optimal for DIP (noise) to be severely sub-optimal for EDIP. For instance, no speed-up is observed for EDIP (noise), applied to the Walnut Sparse 120, after pretraining on the ellipses dataset, if the default learning rate $1e^{-4}$ is used; while a higher learning rate leads to an unstable optimization. A “warm-up” learning rate scheduling with an initial learning rate of $5e^{-4}$, which is linearly decreased to $1e^{-4}$ over the first 5k iterations reveals a substantial speed-up. We use the same learning rate scheduling with DIP (noise), but fail to observe any improvement. Similarly, we observe that validating on the Shepp-Logan phantom for the Lotus Limited 45 setting requires the regularization parameter $\gamma'$ to be increased to $4e^{-6}$ (instead of $1e^{-6}$) for EDIP (FBP) to converge.

![Figure 9. U-Net diagram.](image-url)

Each light-blue box corresponds to a multi-channel feature map. The number of channels is set to 128 at every scale. The arrows denote the different operations.

**Table 1. Hyperparameters for (E)DIP on validation and test data.** The parameters are fine-tuned on DIP (noise), except for the override values specified in the rows starting with “↪”. The learning rate for Walnut Sparse 120 EDIP[-FE] (noise) pretrained on ellipses was linearly decreased over the first 5k iterations, then kept constant at $1e^{-4}$.

| Validation | Learn. rate | $\gamma'$ | Iters. |
|------------|-------------|-----------|--------|
| Lotus Sparse 20 | $1e^{-4}$ | $4e^{-5}$ | 37500 |
| Lotus Limited 45 | $1e^{-4}$ | $1e^{-6}$ | 15000 |
| $\hookrightarrow$ EDIP (FBP) | $1e^{-4}$ | $4e^{-6}$ | 10000 |
| Walnut Sparse 120 | $1e^{-4}$ | $2e^{-7}$ | 50000 |

| Test | Learn. rate | $\gamma'$ | Iters. |
|------|-------------|-----------|--------|
| Lotus Sparse 20 | $1e^{-4}$ | $1e^{-4}$ | 10000 |
| Lotus Limited 45 | $1e^{-4}$ | $6.5e^{-5}$ | 10000 |
| Walnut Sparse 120 | $1e^{-4}$ | $2e^{-7}$ | 30000 |
| $\hookrightarrow$ EDIP[-FE] (noise) pretrained on ellipses | $5e^{-4}$ to $1e^{-4}$ | $2e^{-7}$ | 30000 |

**Table 2. Hyperparameters for TV baselines on test data.**

| Test | Learn. rate | $\gamma'$ | Iters. |
|------|-------------|-----------|--------|
| Lotus Sparse 20 TV | $5e^{-4}$ | $1e^{-4}$ | 5000 |
| Lotus Limited 45 TV | $5e^{-4}$ | $4e^{-5}$ | 5000 |
| Walnut Sparse 120 TV | $5e^{-4}$ | $4e^{-7}$ | 10000 |

| Reference | Learn. rate | $\gamma'$ | Iters. |
|----------|-------------|-----------|--------|
| Lotus (full 120) TV | $1e^{-3}$ | $5e^{-5}$ | 1000 |

**Table 3. Hyperparameters for Lotus gold-standard reference reconstruction.**
Appendix C. Extended Experimental Results

Here we report additional details about the experiments.

C.1. The Lotus (Continued)

Here we also include a limited-view setting, named Lotus Limited 45: 45 angles, range [0, 135°] in steps of 3°. Figure 10 shows exemplary reconstructions on the test-fold of the synthetic datasets used for the pretraining of EDIP, for both Sparse 20 and Limited 45. As one would expect, the FBP suffers severe streak artifacts, but the trained U-Net can recover the shapes reasonably well.

![Ellipses-Lotus Sparse 20](image)

**Figure 10.** Exemplary reconstructions from the synthetic training datasets for Lotus Sparse 20 and Limited 45.

Analogously to Figure 1, the PSNR convergence of EDIP on the Lotus root for the Limited 45 setting is shown in Figure 11; the reconstructions are reported in Figure 12. From the numerical results, one can draw analogous conclusions as for the case of Sparse 20.

Table 4 reports overall tabular results for the Lotus Sparse 20 and the Lotus Limited 45. The following observations can be drawn: pretraining (on the ellipses dataset) can substantially accelerate and stabilize the convergence of DIP. The acceleration factor is more substantial, when considering the FBP as input. The maximum PSNR (Max. PSNR) and steady PSNR suggest that pretraining also contribute to ameliorate the quality of the reconstruction. However, the focus of our work is the speed-up induced by the pretraining, and we abstain from drawing any conclusive remarks on the quality of the reconstructed image when reconstructing from real-measured data. The performance of EDIP-FE is largely comparable to EDIP, but with fewer parameters to update. EDIP-FE is thus computationally lighter than EDIP.

![Ellipses-Lotus Limited 45](image)

**Figure 11.** PSNR convergence of EDIP (pretrained on ellipses dataset) compared to standard DIP on Limited 45 Lotus measurement data. All traces are the mean PSNR of 5 repetitions of the same experimental run (varying the random seed); the standard deviation is also reported.

![Lotus reconstruction of EDIP](image)

**Figure 12.** Lotus reconstruction of EDIP (pretrained on ellipses dataset) compared to standard DIP on Lotus Limited 45 measurement data. From the 5 runs (varying the seed), the one with the (closest to) median PSNR was selected for each method. Note that the reported reconstructions are the best reconstruction (i.e. reconstruction at the minimum loss value).

![Figure 13](image)

**Figure 13.** Shows the convergence of the loss in (1) and of the PSNR, but this time, the PSNR is computed using the network output with minimum loss reached until the current iteration. Using the minimum loss output is a practical way to overcome the instability of the DIP optimization scheme. However, in our main analysis, we consider the raw data, which shows the inherent instability of DIP. The pretraining can greatly accelerate and stabilize the subsequent unsupervised training of EDIP, when compared to the standard...
DIP. This indicates a (plausibly) more favorable optimization landscape of EDIP / EDIP-FE than that of DIP. A stable convergence in practice is important for designing stopping rules for DIP / EDIP.

| Ellipses-Lotus Sparse 20 | Rise time | Max. PSNR | Steady PSNR | Init. PSNR |
|--------------------------|-----------|-----------|-------------|------------|
| DIP (noise)              | 3848      | 31.17     | 31.10       | 11.17      |
| DIP (FBP)                | 3622      | 31.25     | 31.17       | 11.33      |
| DIP-FE (noise)           | 6118      | 31.10     | 31.00       | 11.17      |
| DIP-FE (FBP)             | 4516      | 31.19     | 31.13       | 11.33      |
| EDIP (FBP)               | 195       | 31.65     | 31.21       | 27.04      |
| EDIP (noise)             | 723       | 31.53     | 31.39       | 14.28      |
| EDIP-FE (FBP)            | 226       | 31.59     | 31.26       | 27.04      |
| EDIP-FE (noise)          | 1414      | 31.46     | 31.39       | 14.28      |
| TV                       |           |           |             | 30.73      |

| Ellipses-Lotus Limited 45 | Rise time | Max. PSNR | Steady PSNR | Init. PSNR |
|----------------------------|-----------|-----------|-------------|------------|
| DIP (noise)                | 5470      | 29.85     | 29.69       | 11.17      |
| DIP (FBP)                  | 5419      | 29.84     | 29.69       | 11.32      |
| DIP-FE (noise)             | 5142      | 29.82     | 29.69       | 11.17      |
| DIP-FE (FBP)               | 5056      | 29.83     | 29.67       | 11.32      |
| EDIP (FBP)                 | 524       | 29.83     | 29.68       | 27.55      |
| EDIP (noise)               | 682       | 29.94     | 29.80       | 14.34      |
| EDIP-FE (FBP)              | 245       | 29.85     | 29.72       | 27.55      |
| EDIP-FE (noise)            | 1279      | 29.95     | 29.86       | 14.34      |
| TV                        |           |           |             | 29.62      |

Table 4. Quantitative evaluation for Lotus Sparse 20 and Lotus Limited 45 with EDIP being pretrained on ellipses data. Rise time is defined to be the minimal number of iterations after which the PSNR reaches steady PSNR of DIP (noise) minus 0.1 dB. Both maximum PSNR and steady PSNR are computed using the iteration-wise median PSNR history over the 5 repeated runs (varying the random seed). For steady PSNR, the median value of the median PSNR history over the last 5k iterations is considered. The convergence of TV is very stable, and we report the final PSNR. Initial PSNR is the mean value over the 5 repeated runs. All PSNR values are in dB.

Figure 13. Loss and PSNR convergence for Lotus Sparse 20, with EDIP pretrained on ellipses data. (Left) Loss and PSNR using the network output with minimum loss reached until the current iteration; the original loss curves are shown in the background. Stopping times obtained with a simple stopping criterion are shown by the bars on the top: we select the iteration at which the moving average of \(|l_t(\theta^{i+1}) - l_t(\theta^i)|\), computed with a window size of 100 iterations, becomes less than \(5e^{-6}\). (Right) Variation of the loss, measured by the absolute differences of loss values between subsequent iterations \(|l_t(\theta^{i+1}) - l_t(\theta^i)|\). The moving averages (top) are computed with a window size of 100 iterations, and the mean curve for the 5 runs (with varying random seed) is shown. Histograms (bottom) of \(|l_t(\theta^{i+1}) - l_t(\theta^i)|\) are shown for different iteration ranges.
C.2. The Walnut (Continued)

Figure 14 compares EDIP (FBP) and EDIP (noise), pretrained on the ellipses data, to standard DIP by showing the reconstructions obtained with the parameters after 70, 100, 150 and 250 iterations, respectively. A video showing the full reconstruction process is included in [7] and available at the project page5. We observe that EDIP (FBP) yields a better initial reconstruction, and that both EDIP (FBP) and EDIP (noise) are by far better than DIP (noise). It is worth noting that EDIP (noise) reconstruction after the 70th iteration exhibits some structure resembling the Walnut.

![Intermediate reconstructions](image)

Figure 14. **Intermediate reconstructions** of Walnut Sparse 120, comparing EDIP (FBP) and EDIP (noise), pretrained on ellipses data, to standard DIP. A video showing the full reconstruction process is available at the project page [https://educateddip.github.io/docs.educated_deep_image_prior/](https://educateddip.github.io/docs.educated_deep_image_prior/).

Analogous to Figure 10, Figure 15 shows exemplary reconstructions on the test-fold of the synthetic ellipses dataset used in the pretraining of EDIP for the Walnut Sparse 120 setting. The quantitative results are reported in Table 5, which validates our findings on the Lotus root. Similarly to Figure 13, Figure 16 shows the convergence and stability of the loss in (1), and the PSNR convergence using minimum loss network output.

![Exemplary reconstructions](image)

Figure 15. **Exemplary reconstructions** from the synthetic training dataset of ellipses images for Walnut Sparse 120.

| Method          | Rise time | Max. PSNR | Steady PSNR | Init. PSNR |
|-----------------|-----------|-----------|-------------|------------|
| DIP (noise)     | 20 373    | 34.02     | 33.87       | 6.88       |
| DIP (FBP)       | 13 778    | 34.07     | 33.90       | 6.26       |
| DIP-FE (noise)  | 14 289    | 34.02     | 33.88       | 6.88       |
| DIP-FE (FBP)    | 13 421    | 34.19     | 33.97       | 6.26       |
| EDIP (FBP)      | 4496      | 33.92     | 33.56       | 25.67      |
| EDIP (noise) *  | 9561      | 34.12     | 33.95       | 12.22      |
| EDIP-FE (FBP)   | 4384      | 33.91     | 33.70       | 25.67      |
| EDIP-FE (noise) * | 21 760   | 33.89     | 33.75       | 12.22      |
| TV              |           |           | 31.67       |            |

Table 5. **Quantitative evaluation** for Walnut Sparse 120 with EDIP being pretrained on ellipses data. Rise time is defined to be the minimal number of iterations after which the PSNR reaches steady PSNR of DIP (noise) minus 0.1 dB. Both maximum PSNR and steady PSNR are computed using the iteration-wise median PSNR history over the 5 repeated runs (varying the random seed). For steady PSNR, the median value of the median PSNR history over the last 5k iterations is considered. The convergence of TV is very stable, and we report the final PSNR. Initial PSNR is the mean value over the 5 repeated runs. All PSNR values are in dB. For the experiments marked with “*” a higher initial learning rate was used (see Table 1).

---

5[https://educateddip.github.io/docs.educated_deep_image_prior/]
Figure 16. Loss and PSNR convergence of minimum loss outputs for Walnut Sparse 120, with EDIP pre-trained on ellipses data. (Left) Loss and PSNR using the network output with minimum loss reached until the current iteration; the original loss curves are shown in the background. Stopping times obtained with a simple stopping criterion are shown by the bars on the top; we select the iteration at which the moving average of \(l_t(\theta^{[i+1]}) - l_t(\theta^{[0]})\), computed with a window size of 100 iterations, becomes less than 5e-8. (Right) Variation of the loss, measured by the absolute differences of loss values between subsequent iterations \(l_t(\theta^{[i+1]}) - l_t(\theta^{[i]})\). The moving averages (top) are computed with a window size of 100 iterations, and the mean curve for the 5 runs (with varying random seed) is shown. Histograms (bottom) of \(l_t(\theta^{[i+1]}) - l_t(\theta^{[i]})\) are shown for different iteration ranges.

Appendix D. Validating Pretraining

Figure 17 shows the convergence of the pretraining on the ellipses datasets for the respective Lotus and the Walnut settings, along with the learning rate scheduling.

After pretraining, we select the network parameters to be used for EDIP through a validation stage; each parameter set (i.e. checkpoint) is evaluated by performing EDIP fine-tuning on simulated measurements of the Shepp-Logan phantom. The validation runs for Lotus Sparse 20, Lotus Limited 45, and Walnut Sparse 120 are shown in Figure 18 and Figure 20, respectively. We repeat the pretraining three times (varying the seed) and collect checkpoints after every 20 epochs for Lotus Sparse 20 and Lotus Limited 45, training for a maximum of 100 epochs. We also include the checkpoint for which the model shows minimum validation loss. For Walnut Sparse 120 we pretrain for 20 epochs, and retain only the minimum validation loss checkpoint. In the Lotus settings, we observe that starting EDIP fine-tuning using checkpoints from a later epoch (e.g. 60, 80, 100) is more beneficial. However, it is worth noting that even pretraining for less epochs (e.g. 20) can already greatly benefit the EDIP fine-tuning, although to a lesser degree.

We notice that pretraining considerably ameliorates the quality of the reconstruction of the Shepp-Logan phantom for both the Lotus and Walnut settings. Especially for the Lotus Limited 45 setting a substantial increase in reconstruction quality is observed.

We then investigate whether the selected checkpoints that then are used for the test data — both the Lotus and
Validation runs on the Shepp-Logan phantom for selecting the initial EDIP (FBP) model parameters pretrained on the ellipses dataset for measurement data in the Lotus Sparse 20 and Limited 45 geometry. For Sparse 20 the model from training run 2 after 100 epochs is selected because it has the shortest rise time (with a sufficiently high steady PSNR), whilst, for Limited 45 run 1 after 100 epochs is selected.

The Walnut could be considered an out-of-distribution image class — are still optimal as we switch from the simulated measurements of the Shepp-Logan phantom to the real-measured test data. Figure 19 and Figure 21 show the PSNR convergence using different checkpoints. While we observe a different behavior between validation and test data, the validation selects one of the best two checkpoint.

Validation runs on the Shepp-Logan phantom for selecting the initial EDIP (FBP) model parameters pretrained on the ellipses dataset for measurement data in the Walnut Sparse 120 geometry. The model from training run 1 is selected because it has the shortest rise time (with a sufficiently high steady PSNR).

PSNR convergence using different checkpoints considered during validation (see Figure 18) for EDIP (FBP) pretrained on ellipses dataset on Lotus Sparse 20 measurement data. The parameters from run 2 after 100 epochs are the ones selected by the validation.
puting the full Jacobian matrix $F_\Omega \in \mathbb{R}^{m \times \ell}$ can be executed very efficiently with random sampling.

Stage #1: Randomized Range Finder. We construct a low-rank matrix $B = Q^\top F'(\theta_0) \in \mathbb{R}^{p \times \ell}$. Its computation is then turned into $B^\top = F'(\theta_0)^\top Q$, which is computed through backpropagation. We then approximate the singular values and right singular vectors of $F'(\theta_0)$ by the right singular vectors of $B \approx U\Sigma V^\top$ (with the last few discarded as oversampling: default choice 5). Note that the size of $B \in \mathbb{R}^{p \times \ell}$ is much smaller than that of $F'(\theta_0)$, and a direct SVD computation is feasible.

Algorithm 1 rSVD for Non-linear Forward Map

Require: the Jacobian matrix $F'(\theta_0)$, the target rank $\kappa$, and oversampling parameter $\rho$

1: Draw a $p \times (\kappa + \rho)$ Gaussian random matrix $\Omega = \{\omega_{ij}\}$
2: Form $F_\Omega = F'(\theta_0)\Omega$;
3: Construct an orthonormal basis $Q$ of range($\hat{F}$) using QR decomposition
4: Form the smaller matrix $B = Q^\top F_\Omega$;
5: Compute the SVD of $B = W\Sigma_k\hat{V}_k$;
6: Return $\Sigma_k, \hat{V}_k$.

The singular value spectrum of the linearized forward map with respect to $\theta$ for DIP / EDIP pretrained on ellipses images are shown in Figure 22. During iterations, one observes a dramatic shift of the spectrum for DIP, but for EDIP, the spectrum does not change much. Nonetheless, asymptotically, the singular values decay algebraically for all the cases.

Stage #2: Direct SVD We then directly approximate the SVD of $F'(\theta_0)$ using the information contained in the basis $Q$. We construct a low-rank matrix $B = Q^\top F'(\theta_0) \in \mathbb{R}^{p \times \ell}$. Its computation is then turned into $B^\top = F'(\theta_0)^\top Q$, which is computed through backpropagation. We then approximate the singular values and right singular vectors of $F'(\theta_0)$ by the right singular vectors of $B \approx U\Sigma V^\top$ (with the last few discarded as oversampling: default choice 5). Note that the size of $B \in \mathbb{R}^{p \times \ell}$ is much smaller than that of $F'(\theta_0)$, and a direct SVD computation is feasible.

Appendix E. Diagnostic Analysis

E.1. Approximate Spectral Decomposition

To gain insight into the mechanism of pretraining, we linearize the non-linear forward map $F(\theta) = A\varphi_\theta (A^\dagger y_\delta)$ at the initialization $\theta_0$:

$$F'(\theta) = F(\theta_0) + F'(\theta_0)(\theta - \theta_0),$$

with $F'(\theta_0)$ being the Jacobian matrix of $F(\theta)$ evaluated at $\theta_0$, and $\varphi_\theta$ the network, and $A^\dagger y_\delta$ the input to the network. It follows that, $F'(\theta_0) = A\varphi'_{\theta_0} \in \mathbb{R}^{m \times \ell}$ with $\varphi'_{\theta_0} = \partial \varphi_{\theta}/\partial \theta|_{\theta = \theta_0} \in \mathbb{R}^{n \times \ell}$ denoting the Jacobian of the network’s output w.r.t. its parameters $\theta$, evaluated at $\theta_0$. A naive implementation of $F'(\theta_0)$ is computationally challenging due to the high-dimensionality of the Jacobian matrix $\varphi'_{\theta_0}$. For the purpose of our analysis, we are only interested in recovering the first $\ell$ leading singular values and the corresponding right singular vectors, which are used to represent the parameters (i.e. solution space). Thus, we resort to the randomized Singular Value Decomposition (rSVD) [21, 54]; that is, a randomized algorithm for constructing a low-rank matrix approximation to the matrix $F'(\theta_0)$ (see Algorithm 1). The task of computing a low-rank approximation to $F'(\theta_0)$ can be divided into two computational stages:

Stage #1: Randomized Range Finder. We construct a subspace that captures most of the action of $F'(\theta_0)$. This can be executed very efficiently with random sampling methods. Specifically, we draw a Gaussian random matrix $\Omega \in \mathbb{R}^{p \times \ell}$ and form $\hat{F} = F'(\theta_0)\Omega \in \mathbb{R}^{m \times \ell}$. However, the computation of $F'(\theta_0)\Omega$ requires some care, since computing the full Jacobian matrix $\varphi'_{\theta_0}$ is computationally and memory-wise expensive. We approximate the Jacobian-vector product by means of finite difference approximation, most specifically, we use the following central difference approximation of the first-order derivative:

$$\varphi'_{\theta_0} \omega = (\varphi_{\theta_0+\epsilon \omega} - \varphi_{\theta_0-\epsilon \omega})/(2\epsilon),$$

where the random vector $\omega \in \mathbb{R}^p$ (i.e. one column of the Gaussian random matrix $\Omega$), and $\epsilon > 0$ is a small constant. To compute an orthonormal matrix $Q \in \mathbb{R}^{m \times \ell}$, whose range approximates the range of $\hat{F}$, we use the standard QR factorization [21, 54].
Appendix F. Ablation Study and Limitations

Here we showcase one potential pitfall of the “supervised pretraining + unsupervised fine-tuning” paradigm in the context of DIP.

F.1. An Unexpectedly “Bad Education”

As an alternative training image dataset to the Ellipses dataset, we attempt to design a dataset for the supervised learning stage that we believe to be a better proxy for the Walnut. We then choose human brain images (as ground truths) under the impression that brains are more similar to a walnut [14]. In particular, we consider MRI images of the human brain from the ACRIN-FMISO-Brain (ACRIN 6684) dataset [10, 19, 35, 47]. For the synthetic dataset, we normalize the extracted 2D slices and (mis)interpret the values to be X-ray attenuation coefficients. We use a random data split on patient level, leading to 30,917 training images and 4,524 validation images. Both training and validation images are augmented by random rotations. Figure 23 shows an exemplary reconstruction of the brain dataset, whilst Figure 24 reports the pretraining convergence.

In Figure 25, we show the validation on the Shepp-Logan. Figure 26 shows the comparison between DIP and EDIP trained on the brain dataset. We observe that EDIP deteriorates its performances. We also observe (with surprise!) that EDIP pretrained on the brain dataset reports a lower initial PSNR than when trained on the ellipses dataset (i.e. 25.49 dB vs. 25.67 dB). The brain dataset results, indeed, in inferior input-robustness.

Figure 27 suggests that other pretrained parameters’ configurations also lead to similar subpar results. We observe the inadequacy of the brain dataset (of its education!), which we believed would have led to better transferable properties, than simple ellipses. Instead, it turns out that pretraining on the brain dataset induces “malignant” inductive biases (i.e. a “bad education”) from which EDIP fails to escape, leading to slow convergence and sub-optimal steady PSNR. Possibly the implicit regularization exerted by the
pretraining on the brain dataset essentially restricts the networks from leaving a "pretrained landscape" of sub-optimal parameters’ configurations.

We then check whether using earlier checkpoints would lead to better transferable performances. We, indeed, observe that an early-stopping of the pretraining stage on the brain dataset ameliorates EDIP as shown in Figure 29. It is worth noting that we do not observe a similar behavior when pretraining on the ellipses dataset. We also notice that pretraining for more epochs leads to better input-robustness — the longer one pretrains on the brain dataset, the higher the initial PSNR on the Walnut. However, the longer we pretrain on the brain dataset, the worse EDIP performs on the subsequent reconstruction. We conclude that determining whether a dataset is sufficiently similar to the target reconstruction task is highly nontrivial, and simple intuition might lead to misleading conclusion.

In conclusion, what we believed to make up for a better dataset (i.e. a better proxy knowledge to start with) actually resulted into a "bad education". A posteriori, we realized that the brain dataset is far from being a perfect dataset for supervised pretraining. One could hypothesize that it is by far more specific, and, indeed, less diverse. In contrast, the ellipses dataset is more general; thus, intuitively, it would enforce simple, yet more transferable reconstructive properties. This sheds light on the mechanism of the “supervised pretraining + unsupervised fine-tuning” paradigm.

Table 6. Quantitative evaluation results for EDIP on Walnut Sparse 120 after pretraining on the brain dataset for 20 epochs. No rise time can be reported, because the PSNR is not reaching the steady PSNR of DIP (noise) minus 0.1 dB within the 30k iterations. See Table 5 for the corresponding results from standard DIP and from pretraining on ellipses data.

|                | Rise time | Max. PSNR | Steady PSNR | Init. PSNR |
|----------------|-----------|-----------|-------------|------------|
| EDIP (FBP)     | -         | 33.51     | 33.35       | 25.49      |
| EDIP (noise)   | -         | 33.67     | 32.29       | 12.23      |
| EDIP-FE (FBP)  | -         | 33.43     | 33.24       | 25.49      |
| EDIP-FE (noise)| -         | 31.06     | 30.39       | 12.23      |

Figure 26. PSNR convergence of EDIP pretrained on the brain dataset compared to standard DIP on Walnut Sparse 120 measurement data. All traces are the mean PSNR of 5 repetitions of the same experimental run (varying the random seed); the standard deviation is also reported. See Tab. 6 for complementary tabular results.

Figure 27. PSNR convergence using parameters from different training runs considered during validation (see Figure 25) for EDIP (FBP), pretrained on the brain dataset, on Walnut Sparse 120 measurement data. The parameters from run 1 are the ones selected by the validation.

Figure 28. Walnut reconstruction of EDIP pretrained on brain dataset, compared to standard DIP. From the 5 runs (varying the seed), the one with the (closest to) median PSNR was selected for each method, except for EDIP (FBP) iter. 4500, which is taken from an additional run. See Figure 4 for the Walnut reconstruction with EDIP pretrained on the ellipses dataset.

In conclusion, what we believed to make up for a better dataset (i.e. a better proxy knowledge to start with) actually resulted into a “bad education”. A posteriori, we realized that the brain dataset is far from being a perfect dataset for supervised pretraining. One could hypothesize that it is by far more specific, and, indeed, less diverse. In contrast, the ellipses dataset is more general; thus, intuitively, it would enforce simple, yet more transferable reconstructive properties. This sheds light on the mechanism of the “supervised pretraining + unsupervised fine-tuning” paradigm.
F.2. Early-Stopping

DIP employs early-stopping to deliver satisfactory reconstructions. The use of a regularization term effectively alleviates the need for early-stopping, but a mild degradation of the quality of the reconstructed image is still observed. Our approach would benefit from being combined with an early-stopping criterion, and from tuning the hyperparameter $\gamma$ independently. We would be able to catch the overshooting phase, overshadowing the quality of the reconstruction with respect to DIP. With this regard, we showcase that EDIP leads to a more stable optimization (see Figure 13 and Figure 16). This opens up to the possibility of deploying simple, yet effective stopping criteria such as thresholding.