SVD-DIP: Overcoming the Overfitting Problem in DIP-based CT Reconstruction

Author name(s) withheld EMAIL(S) WITHHELD
Address withheld

Editors: Under Review for MIDL 2023

Abstract

The deep image prior (DIP) is a well-established unsupervised deep learning method for image reconstruction; yet it is far from being *flawless*. The *DIP overfits to noise* if not early stopped, or optimized via a regularized objective. We build on the regularized fine-tuning of a pretrained DIP, by adopting a novel strategy that restricts the learning to the adaptation of *singular values*. The proposed SVD-DIP uses *ad hoc* convolutional layers whose pretrained parameters are decomposed via the singular value decomposition. Optimizing the DIP then solely consists in the fine-tuning of the singular values, while keeping the left and right singular vectors fixed. We thoroughly validate the proposed method on real-measured µCT data of a lotus root as well as two medical datasets (LoDoPaB and Mayo). We report significantly improved stability of the DIP optimization, by *overcoming the overfitting* to noise.

Keywords: Deep Image Prior, Fine-Tuning, Computed Tomography, Singular Value Decomposition

1. Introduction

In medical imaging, we are often interested in inverse problems of the form $y = Ax + \nu$, with $y \in \mathbb{R}^m$ being a noisy measurement, $x \in \mathbb{R}^n$ the unknown image of interest, A a linear forward operator, and $\nu \sim \mathcal{N}(0, \sigma^2 I)$ i.i.d. noise. This problem is often ill-posed, and regularization is needed to recover a sensible image (Engl et al., 1996; Ito and Jin, 2015).

In recent years, deep learning has been applied successfully to many imaging modalities, often via a supervised learning paradigm (Arridge et al., 2019; Ongie et al., 2020). However, supervised learning tends to require a large amount of paired training data to be effective (Baguer et al., 2020). Deep Image Prior (DIP) (Ulyanov et al., 2020) is an unsupervised alternative to applying deep learning to image reconstruction. Its main advantage over supervised methods is that it requires no training data, and learns only on the observed data sample, relying on the rich structure of convolutional neural networks (CNNs) to have a regularizing effect on the image. However, it is not without limitations. The network can overfit to the noise, and has to be freshly trained for every image we intend to reconstruct. The Educated Deep Image Prior (EDIP) (Barbano et al., 2022) is a variant of DIP that addresses some of these issues. It uses pretraining by initializing the network architecture with a pretrained (warm-start) parameter setting, rather than randomly. However, EDIP still suffers from overfitting even if equipped with a regularized objective; iterating beyond a certain point leads to deteriorated quality.

The problem of overfitting is not unique to DIP. Recently, Sun et al. (2022) suggested a novel way of adapting pretrained parameters in a CNN using the singular value decomposition (SVD) to address overfitting and the generalization ability of a CNN specifically in the

© 2023 Anonymous.

context of image segmentation. In this paper, our contribution is to integrate the idea of the SVD fine-tuning into the EDIP framework, achieving remarkable stability. Specifically, we first train a CNN on synthetic data, and before warm-starting the DIP with the pretrained parameter setting, we replace some or all of the layers of the network using the SVD. In our experiments, the resulting DIP (i.e., SVD-DIP) does not overfit to noise even after 100k iterations. The SVD-DIP reports an almost non-existent drop in the maximal PSNR value when compared to EDIP or the vanilla DIP, which start overfitting after a few thousand iterations.

2. The Deep Image Prior

Given a measurement y, the DIP (Ulyanov et al., 2020) finds an x that minimizes $||Ax-y||_2^2$. The method trains a CNN, typically deploying a U-Net architecture (Ronneberger et al., 2015), to fit to the single data sample (z, y), where z is a randomly initialized input (e.g., with i.i.d. Gaussian noise). The DIP finds a parameter setting θ such that the output of the neural network $\varphi_\theta(z) = x$ minimizes the error $||A\varphi_\theta(z) - y||_2^2$. This requires training a CNN for each y; the optimization can take hours or even days, depending on the complexity (esp. high-dimensionality) of the reconstruction task (Barbano et al., 2022). The method relies on the observation that the CNN structure already captures a sufficient amount of low-level image statistics, such that it can reconstruct an image well even without being trained on any data save for the input image (Dittmer et al., 2020). When data is scarce or expensive to acquire, this represents a major upside. However, the Achilles' heel of the DIP is overfitting to noise. When optimizing for too long, the network fits the noise. There are several approaches to alleviate this issue. The use of an explicit regularization term (Liu et al., 2018) is one of them. Instead of $||A \cdot -y||_2^2$, the objective function becomes

$$||A \cdot -y||_2^2 + \gamma \mathrm{TV}(\cdot) \tag{1}$$

where $\gamma > 0$ and $\mathrm{TV}(x) := \sum_{i,j} |x_{i+1,j} - x_{i,j}| + \sum_{i,j} |x_{i,j+1} - x_{i,j}|$ is the anisotropic total variation. The latter only partially alleviates overfitting to the noise, often not being sufficient to prevent it completely. Other methods rely on the Stein's unbiased risk estimator (Jo et al., 2021), or on hand-crafted early-stopping criteria (Liu et al., 2018; Wang et al., 2021). Additionally, one may constrain the network to an under-parameterized regime such that its ability to fit to noise is reduced, e.g., deep decoder (Heckel and Hand, 2019).

Freshly training DIP on multiple measurements is time-consuming. To speed-up DIP, pretraining has been introduced, leading to EDIP (Barbano et al., 2022). The idea is to first train the CNN architecture using available or synthetic data. The obtained parameter setting $\theta^\star$ is used as the starting point for the subsequent DIP recontructive task on y. This "warm-start" allows the network to reconstruct the desired image in fewer iterations, exploiting benign inductive biases learned in the pretraining phase.

3. Overcoming Overfitting via an SVD-based Pretrained CNN

Intuitively, we can think of the warm-start parameter setting obtained via pretraining as a list of tensors W, each coding the weights for a convolutional layer in the CNN. In EDIP, this parameter setting is used as the initialization. Thus, the only mechanism in place to

stop from eventually overfitting to noise is the TV term in the loss. Our approach, the SVD-DIP, integrates the idea from (Sun et al., 2022) of adapting the pretrained parameter setting, into the DIP framework. Specifically, the pretrained tensor $W \in \mathbb{R}^{C_{out} \times C_{in} \times K \times K}$ (which represents $K \times K$ convolutions) is first folded into a matrix $W' \in \mathbb{R}^{C_{out} \times C_{in} K^2}$. Then, the SVD yields three matrices, $U' \in \mathbb{R}^{C_{out} \times R}$, $S' \in \mathbb{R}^{R \times R}$ (diagonal), $V' \in \mathbb{R}^{R \times C_{in} K^2}$ $(R = \min\{C_{out}, C_{in} K^2\})$,

$$W' = U' S' V'.$$

These matrices can be "unfolded" back into three tensors $U \in \mathbb{R}^{C_{out} \times R \times 1 \times 1}$, $S \in \mathbb{R}^{R \times R \times 1 \times 1}$, $V \in \mathbb{R}^{R \times C_{in} \times K \times K}$ such that composing the tensors' induced convolutions is equivalent to taking the original tensor's induced convolution. For the technical details, see Appendix A. The key idea of the proposed approach is to represent the network parameters in this form, then freeze the tensors U and V, keeping their convolutions fixed, but allowing the tuning of the singular values (SVs), i.e. the diagonal of S'.

Freezing U, V and only varying the SVs retains the structure learned by pretraining the network, but the weighting of these structural components can be fine-tuned for a specific task at hand: increasing the weight of more important structures, or reducing less important ones. Figure 2 shows that the SVs of the pretrained CNN are modified by SVD-DIP, but still retain the overall trend of the original values.

Prior to the DIP fine-tuning, the network encodes the same function whether it is warm-started with the pretrained parameter setting, or its SVD-adapted counterpart, since with or without the SVD replacement, each layer performs exactly the same convolutional operation. Once fine-tuning begins, the two methods diverge from each other with SVD-DIP being more constrained due to having fewer parameters. The way the SVD is used to replace convolutional layers can be viewed as parameter compression; if $C_{in} = C_{out}$, the number of SVs is a fraction of the square root of the number of parameters of the original tensor. This drastic reduction in the number of parameters, while still retaining enough expressive power thanks to the structure learned during pretraining being encoded in U and V, greatly alleviates the overfitting issue and stabilizes the convergence, as demonstrated by the experiments below.

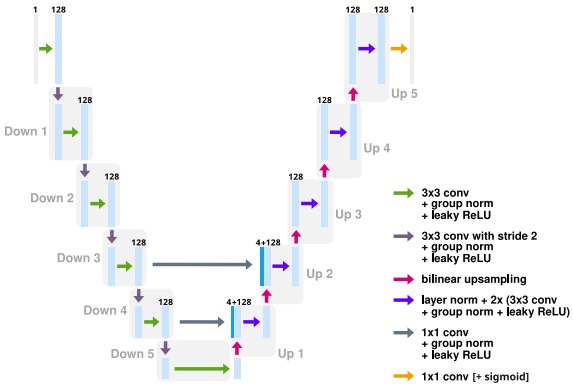


Figure 1: U-Net architecture used for experiments with pretraining on Ellipses.

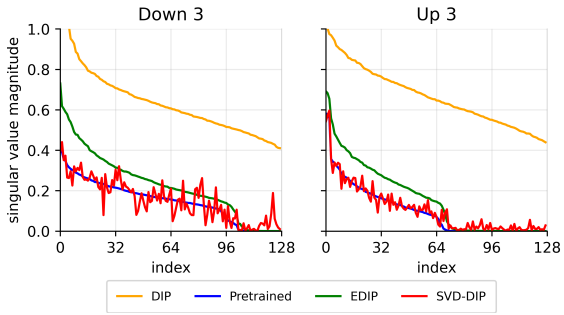


Figure 2: SVs taken from the 1st convolutions of Down layer 3 and Up layer 3 of the U-Net on the lotus.

4. Experiments and Results

4.1. The Experiment Setup

We conduct our experiment on both µCT and medical CT image datasets. During pretraining, the U-Net learns to post-process filtered back-projections of noisy CT projections, that are simulated using the geometry of the target data. We consider the following scenarios:

Pretraining on Ellipses → fine-tuning on Lotus and LoDoPaB's Chest Pretraining is performed on a dataset consisting of synthetically generated ellipses, which is commonly used for inverse problems arising in imaging (Adler and Öktem, 2017) and used to warm-start DIP for CT reconstruction in (Barbano et al., 2022). Synthetic data is advantageous in applications where data is scarce. 5% Gaussian noise is added to the simulated projection data. We use this approach for two target datasets: (i) real-measured µCT data of a lotus root (Bubba et al., 2016) (fan-beam geometry with 20 angles, 429 detector pixels and image size $128 \times 128\,\mathrm{px}^2$); (ii) simulated medical chest CT data from LoDoPaB (Leuschner et al., 2021) (parallel-beam geometry with 200 angles, 513 detector pixels and image size $362 \times 362\,\mathrm{px}^2$). It is simulated to include Poisson noise corresponding to 4096 photons per pixel before attenuation. 10 images were selected via manual inspection, by looking for ground truth images with low artifact corruption. Note that there is a shift in noise distribution between pretraining that uses 5% Gaussian noise and the target data which is either real-measured or simulated to contain Poisson noise. The U-Net architecture is shown in Figure 1, as in (Barbano et al., 2022). The SVD replacements reduce the number of parameters from $128^2 \cdot 3^2$ to 128 in each convolution layer, i.e., reducing by a factor of $\frac{1}{1152}$. We refer to the two transfer settings as "Ellipses-Lotus" and "Ellipses-LoDoPaB".

Pretraining on Chest → fine-tuning on Mayo's Head, Abdomen and Chest Pretraining is performed on LoDoPaB (Leuschner et al., 2021). The dataset contains ca. 40k chest CT images of size $362 \times 362\,\mathrm{px}^2$ and simulated parallel-beam projections with 1000 angles, 513 detector pixels and Poisson noise corresponding to 4096 photons per pixel before attenuation. We experiment with both the original 1000-angle geometry as well as a sub-sampled 200-angle geometry. We use images of different body parts from (Moen et al., 2021) as the images to be reconstructed and simulate observations using the same geometry and noise setting that was used in LoDoPaB. The used U-Net architecture is shown in Appendix B Figure 8(b). We refer to this setting as "LoDoPaB-Mayo".

Then we use the resulting parameters as our pretrained parameters. For the target data containing Poisson noise (Ellipses-LoDoPaB and LoDoPaB-Mayo), we replace the squared error $\|A \cdot -y\|_2^2$ in the objective (1) by the Poisson regression loss matching the noise distribution. We repeat three times for each image from the data sets, for each variant of DIP. We test the classical (randomly initialized) DIP, EDIP and SVD-DIP.

4.2. Comparison of Methods

4.2.1. µCT of lotus root (Ellipses-Lotus)

Figure 3 shows the convergence behaviour of the different DIP frameworks for the lotus. EDIP peaks very early, attaining a maximum PSNR (max PSNR) value of 31.66 dB. However, after achieving the peak value, its PSNR steadily decreases, indicating insufficient

regularization to prevent overfitting. The vanilla DIP is outperformed by the (pretrained) SVD-DIP in terms of the attained maximal PSNR, and is far less stable; its PSNR falls also after peaking, while the PSNR of SVD-DIP does not decline. The DIP also has a lower maximal PSNR. While EDIP has similar stability issues as DIP, it also has better max PSNR. Irrespective of the variant, or the performance metric, pretraining improves on the DIP for the lotus. After around 10k iterations, SVD-DIP and EDIP are at a similar PSNR, and after many more iterations, SVD-DIP reaches and marginally surpasses peak values of EDIP. Thus, the pretrained SVD-DIP performs as well as EDIP in terms of max PSNR and drastically outperforms EDIP in terms of stability (and thus, the PSNR of the final iteration). EDIP has the benefit of the speed at which it reaches its best PSNR value.

As an ablation study, we also include the results for SVD-DIP without pretraining, which performs much worse than the rest. This is not surprising, since it can only learn a weighting of the singular vectors possessing no useful structure, due to the random initialization. The SVD decomposition can't make use of its ability to compress information, so there is no reason to expect the SVD at random initialization to be beneficial.

SVD Truncation Since before and after the fine-tuning, SVs close to zero generally remain so, we observe that the SVD-DIP can be further reduced in the number of parameters by the truncated SVD.

Figure 4 contains a comparison of SVD-DIP settings with different approaches to truncation. For (50% rd.), we truncated all but the top 50% of the SVs. For (10% thresh.), we truncated all SVs which are below 10% of the largest SV. The graph shows that the different

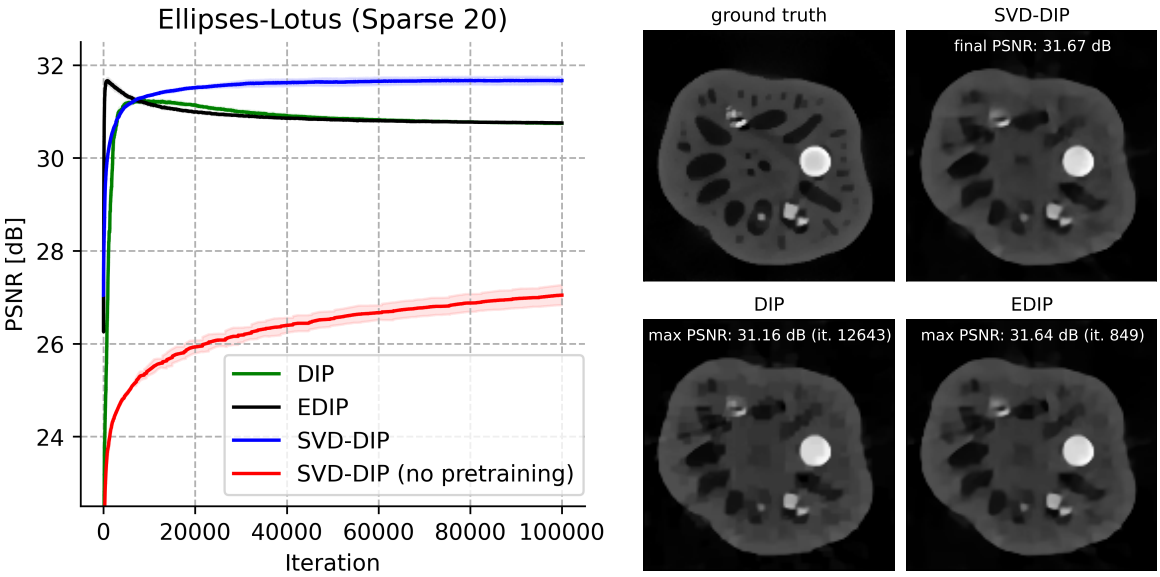


Figure 3: Optimization of DIP, EDIP and SVD-DIP on the lotus, pretraining on ellipses. Each line represents the mean over 3 runs, the colored area the standard deviation. The final SVD-DIP reconstruction is shown on the right, along with the max PSNR reconstructions of DIP and EDIP, which would require ideal early stopping.

variations of SVD-DIP perform very similarly, the difference between their PSNRs being within one standard deviation of the SVD-DIP without truncation. Thus, the lotus root image is simple enough to allow for a low-dimensional representation of the parameter space by only the upper 50% of the SVs (50% rd.), or only the values greater or equal to 10% of the largest (10% thresh.). In this experiment, the truncated DIPs have reduced variance between runs, though this would need to be tested more rigorously. Other percentages, both for the threshold and the reduction approach, were tested, but generally performed worse or were on par with these.

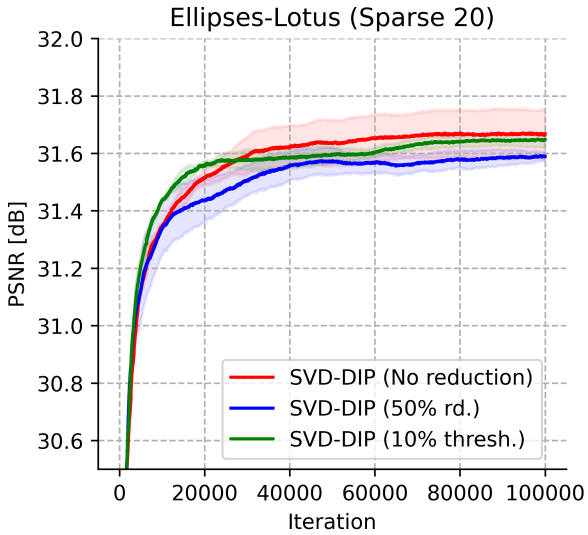


Figure 4: Optimization of SVD-DIP for multiple forms of truncation on the lotus, pretraining on ellipses. Each line represents the mean over 3 runs, the colored area the standard deviation.

4.2.2. Chest CT (Ellipses-LoDoPaB)

Table 1: Mean PSNR values for Ellipses-LoDoPaB (Sparse 200) over 3 runs and 10 images. Max−Final and Final−Init denote the decrease from maximum to final PSNR and the PSNR gain over the pretraining initialization, respectively. Values are in dB.

	DIP		Pretrain.	EDIP			SVD-DIP		
	Final	Max−Final	Init	Final	Max−Final	Final−Init	Final	Max−Final	Final−Init
Mean	32.08	3.26	30.51	32.39	2.70	1.88	34.65	0.11	4.14

As observed above, EDIP peaks very early, after a few thousand iterations. After attaining the maximum, the PSNR starts to decline, cf. Figure 5. DIP has a better max

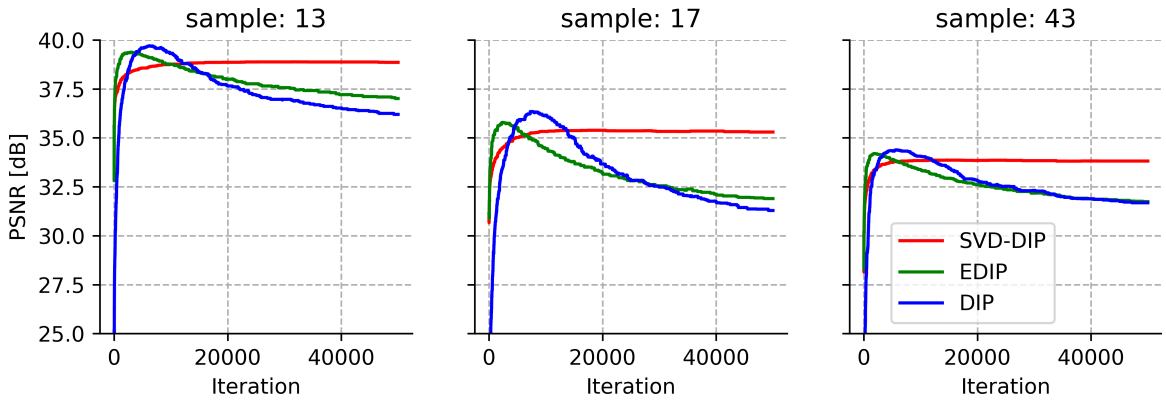


Figure 5: Optimization of DIP, EDIP and SVD-DIP on LoDoPaB (Sparse 200) test samples 13, 17 and 43, pretraining on ellipses. Each line represents the mean over 3 runs.

PSNR on average (see Table 1). SVD-DIP reaches on average a lower max PSNR than EDIP and DIP, but it is much more stable, with the PSNR trending upwards, rather than declining. SVD-DIP outperforms DIP and EDIP, arriving at a slightly lower value stably. Here, for SVD-DIP we replace the convolutional layers in all down- and up-blocks by SVD representations, except the first three down-blocks. We can treat the number of down-blocks that remain unchanged as a hyperparameter, which can be adjusted depending on the need to adapt to different input data. For example, we replace fewer blocks here because we experience a major change in the noise distribution, and want to give the network more flexibility.

4.2.3. CT of different body parts (LoDoPaB-Mayo)

Both pretrained methods, EDIP and SVD-DIP, have the largest gain in the unsupervised optimization stage on the head data, compared to chest and abdomen, cf. Figure 6. This is expected, because the pretraining was done on chest images from LoDoPaB, which are somewhat similar to the chest and abdomen scans of the Mayo dataset, whereas more adaptation is needed for the head scans. SVD-DIP consistently demonstrates both a larger gain in PSNR than EDIP, and a smaller decrease from maximal PSNR to the final value. The max PSNR of DIP is often higher than that of EDIP and SVD-DIP on the head data, while it is on average close or lower for abdomen and chest (see Figure 7). While less visible within the 40k iterations, DIP's performance deteriorates when iterating for longer; on N005, we exemplarily tested running it for 200k iterations, where it fell below the final PSNR values at 40k iterations of SVD-DIP (37.72 dB) and EDIP (37.06 dB) after 84k and 141k iterations, respectively.

5. Conclusion

In this work we built on the regularized fine-tuning of a pretrained DIP and adopted a novel strategy that restricts the learning to the adaptation of singular values of the unfolded network parameter tensor. We proposed a variant of the DIP, named SVD-DIP, that overall

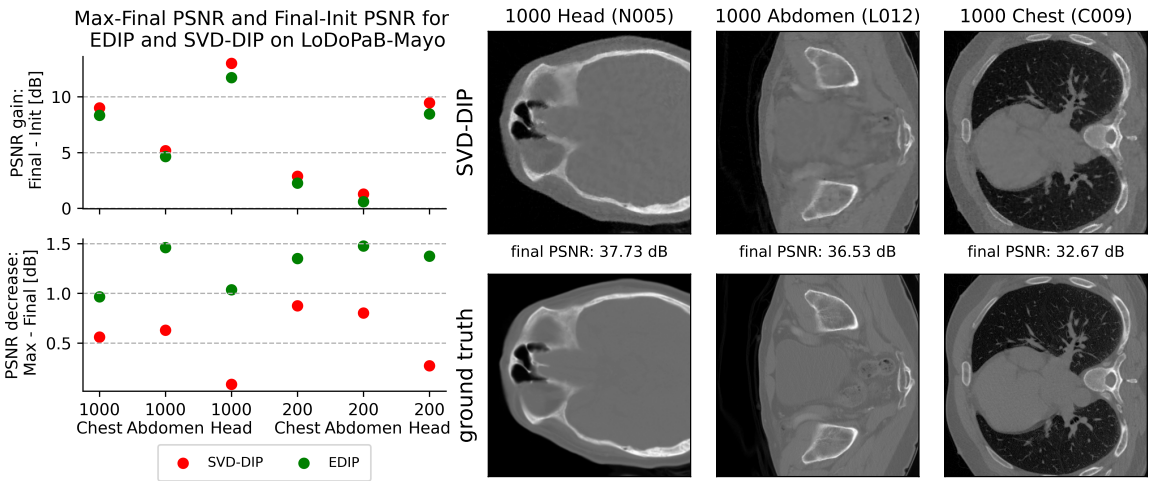


Figure 6: PSNR gains and decreases for EDIP and SVD-DIP on different body parts from the LoDoPaB-Mayo dataset, pretraining on LoDoPaB. Both 1000 and 200 angle settings are considered, indicated on the horizontal axis. Final reconstructions of SVD-DIP are shown for the 1000 angle setting, along with the ground truth.

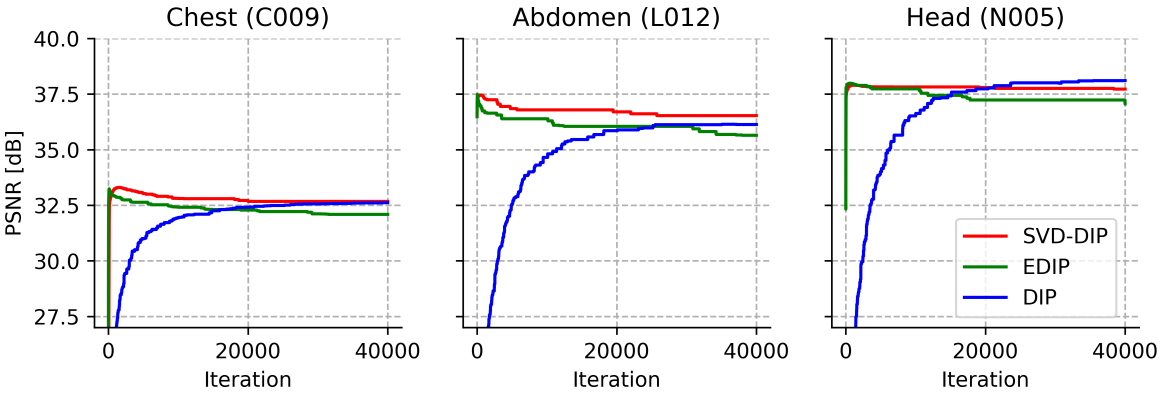


Figure 7: Optimization of DIP, EDIP and SVD-DIP on samples from different body parts from the LoDoPaB-Mayo dataset, pretraining on LoDoPaB. The 1000 angle setting is used. Each line represents the mean over 3 runs.

overcomes the need for early stopping. This approach yields a more stable (less prone to overfit to noise) DIP optimization. The empirical results suggest that while the SVD-DIP reconstructive properties are on par with (or slightly worse than) those of DIP or EDIP, it loses very little in terms of PSNR, even after iterating for a long time.

Acknowledgments

Acknowledgments withheld.

References

Jonas Adler and Ozan Öktem. Solving ill-posed inverse problems using iterative deep neural networks. *Inverse Problems*, 33(12):124007, nov 2017. doi: 10.1088/1361-6420/aa9581. URL https://doi.org/10.1088%2F1361-6420%2Faa9581.

Simon Arridge, Peter Maass, Ozan Öktem, and Carola-Bibiane Schönlieb. Solving inverse problems using data-driven models. *Acta Numerica*, 28:1–174, 2019. doi: 10.1017/S0962492919000059.

Daniel Otero Baguer, Johannes Leuschner, and Maximilian Schmidt. Computed tomography reconstruction using deep image prior and learned reconstruction methods. *Inverse Problems*, 36(9):094004, sep 2020. doi: 10.1088/1361-6420/aba415. URL https://doi.org/10.1088%2F1361-6420%2Faba415.

Riccardo Barbano, Johannes Leuschner, Maximilian Schmidt, Alexander Denker, Andreas Hauptmann, Peter Maass, and Bangti Jin. An educated warm start for deep image prior-based micro ct reconstruction. *IEEE Transactions on Computational Imaging*, pages 1–12, 2022. doi: 10.1109/TCI.2022.3233188.

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

Sören Dittmer, Tobias Kluth, Peter Maass, and Daniel Otero Baguer. Regularization by architecture: a deep prior approach for inverse problems. *J. Math. Imaging Vision*, 62 (3):456–470, 2020. ISSN 0924-9907. doi: 10.1007/s10851-019-00923-x. URL https://doi.org/10.1007/s10851-019-00923-x.

Heinz W. Engl, Martin Hanke, and Andreas Neubauer. *Regularization of Inverse Problems*. Kluwer Academic Publishers Group, Dordrecht, 1996. ISBN 0-7923-4157-0.

Reinhard Heckel and Paul Hand. Deep decoder: Concise image representations from untrained non-convolutional networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rylV-2C9KQ.

Kazufumi Ito and Bangti Jin. *Inverse Problems: Tikhonov Theory and Algorithms*. World Scientific, Hackensack, NJ, 2015. ISBN 978-981-4596-19-0.

Yeonsik Jo, Se Young Chun, and Jonghyun Choi. Rethinking deep image prior for denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5087–5096, October 2021.

Johannes Leuschner, Maximilian Schmidt, Daniel Otero Baguer, and Peter Maass. Lodopab-ct, a benchmark dataset for low-dose computed tomography reconstruction. *Scientific Data*, 8(1):109, 2021. ISSN 2052-4463. doi: 10.1038/s41597-021-00893-z. URL https://doi.org/10.1038/s41597-021-00893-z.

Jiaming Liu, Yu Sun, Xiaojian Xu, and Ulugbek S. Kamilov. Image restoration using total variation regularized deep image prior, 2018. URL https://arxiv.org/abs/1810.12864.

Taylor R. Moen, Baiyu Chen, David R. Holmes III, Xinhui Duan, Zhicong Yu, Lifeng Yu, Shuai Leng, Joel G. Fletcher, and Cynthia H. McCollough. Low-dose ct image and projection dataset. *Medical Physics*, 48(2):902–911, 2021. doi: https://doi.org/10.1002/mp.14594. URL https://aapm.onlinelibrary.wiley.com/doi/abs/10.1002/mp.14594.

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.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. URL https://arxiv.org/abs/1505.04597.

Yanpeng Sun, Qiang Chen, Xiangyu He, Jian Wang, Haocheng Feng, Junyu Han, Errui Ding, Jian Cheng, Zechao Li, and Jingdong Wang. Singular value fine-tuning: Few-shot segmentation requires few-parameters fine-tuning. In *NeurIPS 2022*, 2022. URL https://openreview.net/forum?id=LEqYZz7cZOI.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. *International Journal of Computer Vision*, 128(7):1867–1888, mar 2020. doi: 10.1007/s11263-020-01303-4. URL https://doi.org/10.1007%2Fs11263-020-01303-4.

Hengkang Wang, Taihui Li, Zhong Zhuang, Tiancong Chen, Hengyue Liang, and Ju Sun. Early stopping for deep image prior, 2021. URL https://arxiv.org/abs/2112.06074.

Appendix A. The Singular Value Decomposition is Sensible

We denote by $[N]$ the set $\{1, \dots, N\}$.
Let $W := (w_{n,m,k_1,k_2})_{(n,m,k_1,k_2)\in[N]\times[M]\times[K]^2} \in \mathbb{R}^{N\times M\times K\times K}$,
and $X := (x_{m,j_1,j_2})_{(m,j_1,j_2)\in[M]\times[J]^2} \in \mathbb{R}^{M\times J\times J}$ be two tensors, with K being an odd integer.
Then we can define the 2d convolution of X with W as a tensor
$\mathrm{conv2d}(W, X) := (y_{n,j_1,j_2})_{\dots} \in \mathbb{R}^{N\times J\times J}$ with

$$y_{n,j_1,j_2} := \sum_{m=1}^{M}\sum_{k_1=1}^{K}\sum_{k_2=1}^{K} w_{n,m,k_1,k_2} \cdot X_{m,j_1+k_1-\frac{K-1}{2}-1,j_2+k_2-\frac{K-1}{2}-1}$$

with $X_{m,i_1,i_2} := 0$ when either of the indices i_1, i_2 is outside of $[J]$.

The following is a simple fact about 2D convolutions; for a given weight tensor $W \in \mathbb{R}^{N\times M\times K\times K}$ that can be represent by a matrix $A'B' = W' \in \mathbb{R}^{N\times MK^2}$, and a given feature map $X \in \mathbb{R}^{M\times J\times J}$, taking the 2-dimensional convolution of X with respect to W is identical to convolving X by appropriate tensor representations of $B' \in \mathbb{R}^{R\times M\cdot K\cdot K}$ and (then) $A' \in \mathbb{R}^{N\times R\cdot 1\cdot 1}$. Note that the second tensor only codes 1×1 convolutions. To prove this result, we need the following lemma:

Lemma 1 *Let $M, K \in \mathbb{N}_{>0}$. Given a bijection between indices $f : [M] \times [K]^2 \to [MK^2]$, we can define a function that folds tensors into matrices:*

$$\Phi_f : \biguplus_{N=1}^{\infty} \mathbb{R}^{N\times M\times K\times K} \to \biguplus_{N=1}^{\infty} \mathbb{R}^{N\times MK^2}$$
$$(x_{n,m,k_1,k_2})_{(n,m,k_1,k_2)\in[N]\times[M]\times[K]^2} \mapsto (x_{n,f^{-1}(i)})_{(n,i)\in[N]\times[MK^2]}$$

Let $g : [M] \times [K]^2 \to [MK^2]$ be a bijection, $N, R, J \in \mathbb{N}_{>0}$ be tensor dimensions, $X \in \mathbb{R}^{M\times J\times J}$ an input feature map. Then the following diagram commutes:

$$\begin{tikzcd}
\mathbb{R}^{N\times R} \times \mathbb{R}^{R\times MK^2} \arrow[r, "\textit{Matrix Multiplication}"] \arrow[d, "(\Phi^{-1}_{(Id_{[1]},Id_{[1]})^{-1}}, \Phi_g^{-1})"] & \mathbb{R}^{N\times MK^2} \arrow[r, "\Phi_g^{-1}"] & \mathbb{R}^{N\times M\times K\times K} \arrow[d, "conv2d(\cdot,X)"] \\
\mathbb{R}^{N\times R\times 1\times 1} \times \mathbb{R}^{R\times M\times K\times K} \arrow[r, "(Id, conv2d(\cdot,X))"] & \mathbb{R}^{N\times R\times 1\times 1} \times \mathbb{R}^{R\times J\times J} \arrow[r, "conv2d(\cdot,\cdot)"] & \mathbb{R}^{N\times J\times J}
\end{tikzcd}$$

Proof Let $W := (w_{n,m,k_1,k_2})_{(n,m,k_1,k_2)\in[N]\times[M]\times[K]^2} \in \mathbb{R}^{N\times M\times K\times K}$ be a tensor. Fix a bijection $g : [M] \times [K]^2 \to [MK^2]$. We define $W' := \Phi_g^{-1}(W)$. Let $R \in \mathbb{N}_{>0}$, $A' \in \mathbb{R}^{N\times R}$, $B' \in \mathbb{R}^{R\times MK^2}$ such that $W' = A'B'$. Then we define two tensors

$$(a_{n,r,k_1,k_2})_{(n,r,k_1,k_2)\in[N]\times[R]\times[1]\times[1]} := A := \Phi^{-1}_{(Id_{\{1\}},Id_{\{1\}})^{-1}}(A'),$$
$$(b_{r,m,k_1,k_2})_{(r,m,k_1,k_2)\in[R]\times[M]\times[K]^2} := B := \Phi_g^{-1}(B').$$

Let $X \in \mathbb{R}^{M\times J\times J}$ be an arbitrary tensor. We claim

$$\mathrm{conv2d}(W, X) = \mathrm{conv2d}(A, \mathrm{conv2d}(B, X)).$$

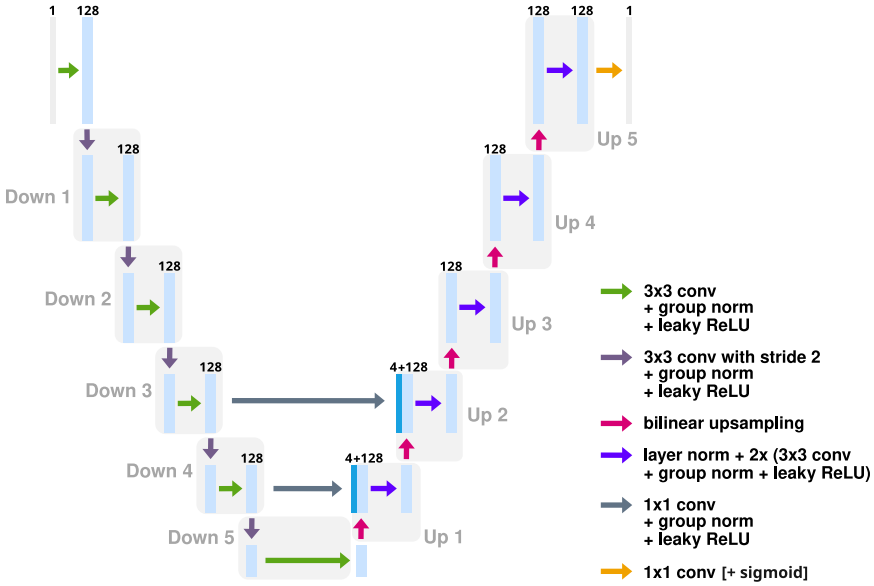
To this end, let $Y := (y_{r,j_1,j_2})_{(r,j_1,j_2)\in[R]\times[J]^2} := \mathrm{conv2d}(B, X)$, and let π_3, π_4 be the mappings that extract the third (resp. fourth) entry of a tuple. Then the proof reduces to the following elementary argument, exploiting the linearity of the operators involved:

$$
\begin{aligned}
&(\mathrm{conv2d}(A, \mathrm{conv2d}(B, X)))_{n,j_1,j_2} = (\mathrm{conv2d}(A, Y))_{r,j_1,j_2} \\
&= \sum_{r=1}^{R} \sum_{k_1=1}^{1} \sum_{k_2=1}^{1} a_{n,r,k_1,k_2} \cdot Y_{r,j_1+k_1-\frac{1-1}{2}-1,j_2+k_2-\frac{1-1}{2}-1} \\
&= \sum_{r=1}^{R} a_{n,r,1,1} \cdot Y_{r,j_1,j_2} \\
&= \sum_{r=1}^{R} a_{n,r,1,1} \cdot \sum_{m=1}^{M} \sum_{k_1=1}^{K} \sum_{k_2=1}^{K} b_{r,m,k_1,k_2} \cdot X_{m,j_1+k_1-\frac{K-1}{2}-1,j_2+k_2-\frac{K-1}{2}-1} \\
&= \sum_{r=1}^{R} a_{n,r} \cdot \sum_{i=1}^{MK^2} b_{r,i} \cdot X_{m,j_1+\pi_3(\Phi_g^{-1}(i))-\frac{K-1}{2}-1,j_2+\pi_3(\Phi_g^{-1}(i))-\frac{K-1}{2}-1} \\
&= \sum_{i=1}^{MK^2} \sum_{r=1}^{R} a_{n,r} \cdot b_{r,i} \cdot X_{m,j_1+\pi_3(\Phi_g^{-1}(i))-\frac{K-1}{2}-1,j_2+\pi_3(\Phi_g^{-1}(i))-\frac{K-1}{2}-1} \\
&= \sum_{i=1}^{MK^2} w_{n,i} \cdot X_{m,j_1+\pi_3(\Phi_g^{-1}(i))-\frac{K-1}{2}-1,j_2+\pi_3(\Phi_g^{-1}(i))-\frac{K-1}{2}-1} \\
&= \sum_{m=1}^{M} \sum_{k=1}^{K} \sum_{k_2=1}^{K} w_{n,m,k_1,k_2} \cdot X_{m,j_1+k_1-\frac{K-1}{2}-1,j_2+k_2-\frac{K-1}{2}-1} \\
&= (\mathrm{conv2d}(W, X))_{n,j_1,j_2}
\end{aligned}
$$

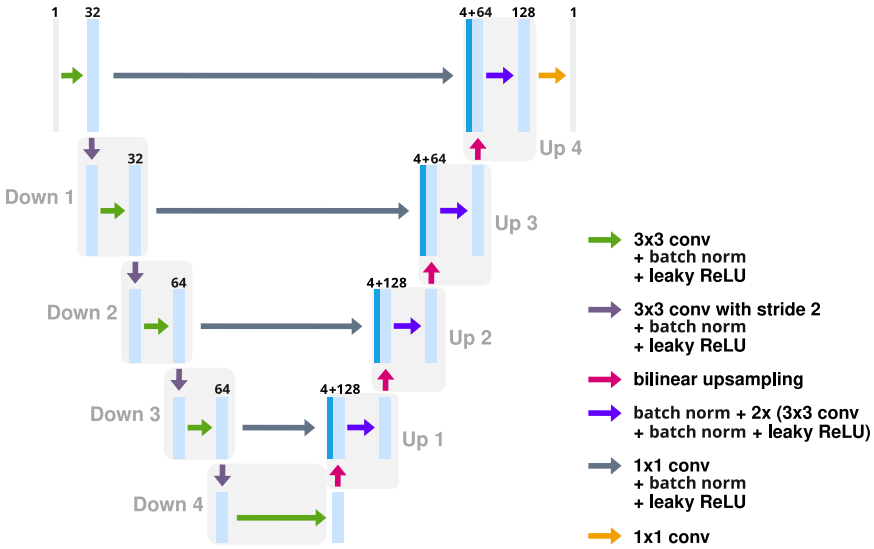
This completes the proof of the claim and hence also the lemma. ■

By applying the lemma twice, we deduce that we can replace a convolutional layer with the three induced by the SVD decomposition.

Appendix B. Experimental setup details



(a) U-Net used for Ellipses-Lotus and Ellipses-LoDoPaB. Linear output activation is used for Ellipses-Lotus, while Ellipses-LoDoPaB uses sigmoid output activation.



(b) U-Net used for LoDoPaB-Mayo

Figure 8: U-Net architectures

Appendix C. Additional results

Table 2: PSNR values for Ellipses-Lotus (Sparse 20)

Method	Final	Max (Iteration)	Max−Final	Final−Init
DIP	30.75	31.24 (8527)	0.4970	16.50
EDIP	30.75	31.66 (832)	0.9090	3.71
SVD-DIP (No reduction)	31.67	31.67 (94662)	0.0028	4.63
SVD-DIP (50% rd.)	31.59	31.59 (98941)	0.0027	4.83
SVD-DIP (10% thresh.)	31.65	31.65 (98339)	0.0010	5.26
SVD-DIP (50% rd. no pretraining)	27.31	27.31 (99540)	0.0000	10.58

Table 3: PSNR values for Ellipses-LoDoPaB (Sparse 200) samples

	DIP		Pretrain.	EDIP			SVD-DIP		
sample	Final	Max−Final	Init	Final	Max−Final	Final−Init	Final	Max−Final	Final−Init
6	34.81	2.96	32.60	35.07	2.55	2.47	37.24	0.04	4.64
9	30.67	3.92	30.22	30.98	3.44	0.76	33.87	0.06	3.65
11	29.69	4.24	28.65	30.55	3.05	1.91	33.05	0.06	4.40
13	36.17	3.57	35.57	36.97	2.42	1.41	38.86	0.04	3.30
17	31.63	4.76	31.12	31.81	4.04	0.69	35.29	0.16	4.18
19	31.33	2.72	30.48	31.47	2.29	1.00	33.52	0.19	3.04
22	30.56	2.90	31.14	30.97	2.36	−0.17	32.86	0.18	1.72
29	32.76	3.25	31.43	32.94	2.92	1.51	35.27	0.15	3.84
34	31.47	1.59	24.76	31.40	1.45	6.64	32.73	0.12	7.97
43	31.75	2.67	29.13	31.70	2.50	2.57	33.78	0.13	4.65
Mean	32.08	3.26	30.51	32.39	2.70	1.88	34.65	0.11	4.14

Table 4: PSNR values for LoDoPaB-Mayo (Full 1000), Chest/C samples

	DIP		Pretrain.	EDIP			SVD-DIP		
sample	Final	Max− Final	Init	Final	Max− Final	Final− Init	Final	Max− Final	Final− Init
C001	33.64	0.00	23.43	32.95	2.16	9.52	33.54	1.34	10.18
C002	30.46	0.00	24.95	30.55	0.61	5.60	30.77	0.29	6.11
C004	31.76	0.00	24.14	31.89	0.65	7.75	32.13	0.31	8.21
C009	32.62	0.00	20.67	32.09	1.15	11.42	32.67	0.63	11.97
C012	31.10	0.00	27.73	31.14	1.05	3.41	31.53	0.65	5.79
C016	31.66	0.00	25.71	32.11	0.66	6.40	32.21	0.35	6.67
C019	34.06	0.00	22.29	33.25	1.20	10.96	34.02	0.81	11.99
C021	32.05	0.00	19.62	32.36	0.13	12.75	32.35	0.29	12.64
C023	31.02	0.00	23.33	30.93	0.61	7.60	31.18	0.29	7.98
C027	33.10	0.00	24.58	32.56	1.45	7.97	33.11	0.64	8.49
Mean	32.15	0.00	23.64	31.98	0.97	8.34	32.35	0.56	9.00

Table 5: PSNR values for LoDoPaB-Mayo (Sparse 200), Chest/C samples

	DIP		Pretrain.	EDIP			SVD-DIP		
sample	Final	Max− Final	Init	Final	Max− Final	Final− Init	Final	Max− Final	Final− Init
C001	30.25	0.00	32.27	30.80	2.26	−1.47	31.56	1.56	−0.77
C002	27.89	0.00	29.72	28.90	1.10	−0.83	29.26	0.82	−0.46
C004	28.79	0.00	28.09	30.04	1.13	1.94	30.44	0.70	2.34
C009	29.80	0.00	30.17	30.16	1.45	−0.02	30.78	0.86	0.79
C012	28.13	0.00	29.84	29.30	1.41	−0.54	29.87	0.94	0.09
C016	28.21	0.00	27.91	29.89	1.11	1.98	30.24	0.76	2.49
C019	31.11	0.00	26.95	31.24	1.18	4.29	31.98	0.77	5.09
C021	28.43	0.00	17.02	30.14	0.93	13.11	30.51	0.35	14.06
C023	29.02	0.00	24.22	29.34	0.95	5.12	29.75	0.51	5.55
C027	29.33	0.00	31.52	30.45	1.97	−1.07	31.01	1.47	−0.46
Mean	29.10	0.00	27.77	30.03	1.35	2.25	30.54	0.87	2.87

Table 6: PSNR values for LoDoPaB-Mayo (Full 1000), Abdomen/L samples

	DIP		Pretrain.	EDIP			SVD-DIP		
sample	Final	Max− Final	Init	Final	Max− Final	Final− Init	Final	Max− Final	Final− Init
L004	36.75	0.00	31.75	35.92	1.54	4.17	36.86	0.40	5.41
L006	37.66	0.00	35.23	36.50	2.26	1.27	37.45	1.41	2.37
L012	36.13	0.01	36.48	35.65	1.84	−0.83	36.53	0.93	0.01
L014	37.72	0.00	36.14	36.45	2.53	0.31	37.58	1.26	1.66
L019	35.76	0.00	36.27	35.18	2.72	−1.09	36.42	1.58	0.14
L024	36.50	0.00	20.07	36.29	0.25	16.22	36.41	0.00	13.86
L027	35.81	0.00	26.60	35.51	0.40	8.90	35.79	0.00	8.96
L030	34.57	0.01	23.74	34.71	0.50	10.97	34.94	0.04	11.37
L033	34.82	0.00	32.20	34.68	1.07	2.48	35.07	0.27	3.23
L035	36.51	0.00	32.31	36.25	1.52	3.95	37.09	0.39	4.60
Mean	36.22	0.00	31.08	35.71	1.46	4.63	36.41	0.63	5.16

Table 7: PSNR values for LoDoPaB-Mayo (Sparse 200), Abdomen/L samples

	DIP		Pretrain.	EDIP			SVD-DIP		
sample	Final	Max− Final	Init	Final	Max− Final	Final− Init	Final	Max− Final	Final− Init
L004	33.38	0.00	34.48	33.80	1.50	−0.68	34.98	0.58	0.19
L006	32.47	0.00	33.20	34.00	2.14	0.80	35.19	1.21	1.93
L012	31.77	0.00	34.10	33.39	1.75	−0.70	34.09	0.93	0.12
L014	32.90	0.00	33.77	34.02	2.45	0.25	35.05	1.47	1.41
L019	31.75	0.00	33.78	32.90	2.29	−0.89	33.67	1.47	−0.03
L024	32.48	0.00	28.28	33.97	0.09	5.70	33.80	0.00	5.62
L027	31.72	0.04	30.90	32.70	1.03	1.80	33.28	0.62	2.24
L030	31.56	0.00	31.85	32.45	0.95	0.60	33.06	0.55	1.01
L033	31.31	0.00	32.64	32.04	1.23	−0.60	32.82	0.48	0.09
L035	32.96	0.00	34.42	34.00	1.33	−0.42	34.88	0.71	0.11
Mean	32.23	0.00	32.74	33.33	1.48	0.59	34.08	0.80	1.27

Table 8: PSNR values for LoDoPaB-Mayo (Full 1000), Head/N samples

	DIP		Pretrain.	EDIP			SVD-DIP		
sample	Final	Max−Final	Init	Final	Max−Final	Final−Init	Final	Max−Final	Final−Init
N001	38.77	0.05	28.68	37.44	1.49	8.76	38.94	0.25	10.33
N003	36.62	0.04	17.54	35.61	0.60	18.07	36.29	0.00	18.65
N005	38.10	0.02	32.49	37.06	0.93	4.57	37.72	0.18	5.10
N012	37.15	0.02	30.70	36.20	1.78	5.50	37.20	0.18	9.77
N017	36.69	0.08	16.94	35.93	0.88	18.99	37.02	0.01	20.11
N021	38.08	0.03	19.84	37.12	1.00	17.29	37.86	0.00	18.46
N024	37.16	0.07	24.67	36.08	0.82	11.40	36.80	0.12	12.15
N025	38.28	0.08	26.31	36.93	0.84	10.63	38.04	0.01	11.50
N029	38.85	0.03	23.49	37.22	1.06	13.73	38.12	0.04	14.65
N030	38.43	0.02	28.92	37.28	0.97	8.36	38.39	0.05	9.24
Mean	37.81	0.04	24.96	36.69	1.04	11.73	37.64	0.08	13.00

Table 9: PSNR values for LoDoPaB-Mayo (Sparse 200), Head/N samples

	DIP		Pretrain.	EDIP			SVD-DIP		
sample	Final	Max−Final	Init	Final	Max−Final	Final−Init	Final	Max−Final	Final−Init
N001	35.14	0.04	27.53	34.38	2.07	6.85	36.20	0.25	7.91
N003	32.87	0.04	20.00	32.49	0.72	12.50	33.70	0.02	13.91
N005	34.31	0.10	30.43	33.50	1.64	3.07	34.48	0.51	4.07
N012	33.51	0.01	27.62	32.94	1.45	5.32	33.90	0.41	5.39
N017	33.52	0.06	21.16	33.14	0.97	11.98	34.23	0.04	12.89
N021	34.77	0.12	23.23	34.23	1.17	10.99	35.46	0.00	12.53
N024	33.40	0.08	22.70	32.89	1.34	10.18	34.07	0.26	11.44
N025	34.36	0.12	25.69	33.19	1.65	7.50	35.13	0.43	9.47
N029	35.28	0.09	24.82	34.25	1.33	9.44	35.20	0.40	9.58
N030	34.44	0.03	27.34	34.12	1.41	6.77	35.25	0.39	7.51
Mean	34.16	0.07	25.05	33.51	1.37	8.46	34.76	0.27	9.47

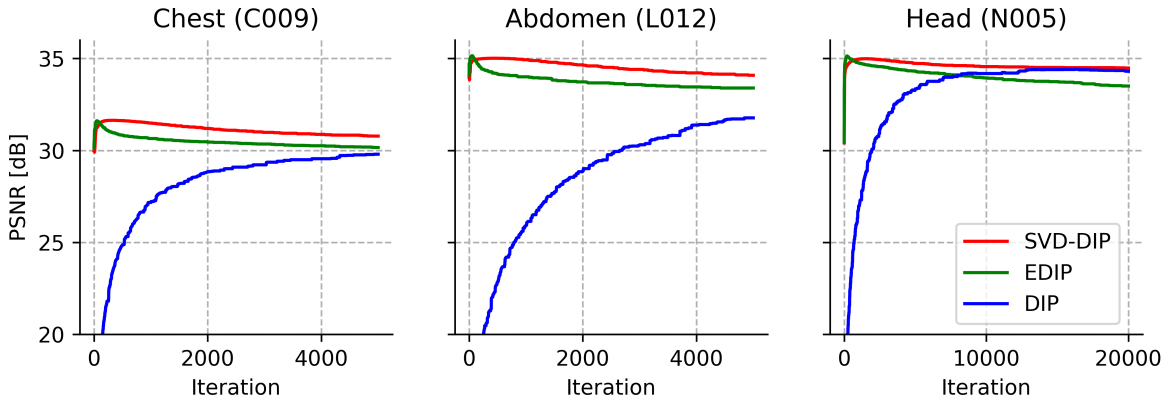


Figure 9: Optimization of DIP, EDIP and SVD-DIP on LoDoPaB-Mayo (Sparse 200) for samples C009, L012 and N005. Each line represents the mean over 3 runs.