Learning Lipschitz-Controlled Activation Functions in Neural Networks for Plug-and-Play Image Reconstruction Methods

Anonymous Author(s)
Affiliation
Address
email

Abstract

Ill-posed linear inverse problems are frequently encountered in image reconstruction tasks. Image reconstruction methods that combine the Plug-and-Play (PnP) priors framework with convolutional neural network (CNN) based denoisers have shown impressive performances. However, it is non-trivial to guarantee the convergence of such algorithms, which is necessary for sensitive applications such as medical imaging. It has been shown that PnP algorithms converge when deployed with a certain class of averaged denoising operators. While such averaged operators can be built from 1-Lipschitz CNNs, imposing such a constraint on CNNs usually leads to a severe drop in performance. To mitigate this effect, we propose the use of deep spline neural networks which benefit from learnable piecewise-linear spline activation functions. We introduce "slope normalization" to control the Lipschitz constant of these activation functions. We show that averaged denoising operators built from 1-Lipschitz deep spline networks consistently outperform those built from 1-Lipschitz ReLU networks.

1 Introduction

Linear inverse problems are ubiquitous in medical imaging. There, the goal is to reconstruct an image $\mathbf{x} \in \mathbb{R}^K$ from measurements $\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n} \in \mathbb{R}^M$. The linear operator $\mathbf{H}\colon \mathbb{R}^K \to \mathbb{R}^M$ models the acquisition system and $\mathbf{n} \in \mathbb{R}^M$ is an additive measurement noise. These problems are typically ill-posed and additional information about $\mathbf{x}$ is required to obtain meaningful solutions. In the variational framework for solving an inverse problem, one formulates it as an optimization task

$$\widehat{\mathbf{x}} = \underset{\mathbf{x}\in\mathbb{R}^K}{\arg\min} \left\{ E(\mathbf{y}, \mathbf{H}\mathbf{x}) + \lambda R(\mathbf{x}) \right\}, \tag{1}$$

where the data-fidelity term $E\colon \mathbb{R}^M \times \mathbb{R}^M \to \mathbb{R}_+$ imposes closeness between the solution estimate and the acquired measurements, the regularization term $R\colon \mathbb{R}^K \to \mathbb{R}_+$ imposes some prior knowledge on the image of interest, and $\lambda \in \mathbb{R}_+$ is a tunable hyperparameter. The cost functional in (1) is then typically minimized using proximal algorithms such as forward-backward splitting (FBS) [10] and the alternating direction method of multipliers (ADMM) [5].

The main idea in the Plug-and-Play (PnP) priors framework [24, 8] is to replace the proximal operator of R in the iterations of proximal algorithms with some denoiser, even though it might not correspond to an explicit regularization term. This implicit regularization approach has been shown to yield better results than conventional variational methods for a variety of inverse problems since it allows the use of powerful denoisers such as NLM [6], WNNM [13], BM3D [11], and neural networks [25, 17, 20], which have emerged as state-of-the-art. However, the delicate point that remains is ensuring the

Submitted to the NeurIPS Workshop on Deep Learning and Inverse Problems, 2021.

convergence of these algorithms, which is non-trivial but essential for sensitive applications such as the ones encountered in medical imaging.

There exist several works that analyze conditions on the denoiser under which PnP algorithms are guaranteed to converge [18, 21, 7, 19, 12]. For example, Ryu *et al.* [17] show that PnP-FBS and PnP-ADMM provably converge to fixed points if the denoiser obeys an appropriate Lipschitz condition. They then propose a practical way to enforce the derived Lipschitz constraint while training neural network denoisers. However, their analysis requires the data-fidelity term to be strongly convex and this unfortunately rules out ill-posed inverse problems. In order to design convergent PnP schemes for ill-posed problems, stricter conditions need to be enforced on the denoiser. More specifically, it has been shown that averagedness (firm nonexpansiveness) of the denoiser is sufficient to guarantee fixed point convergence of PnP-FBS (PnP-ADMM) [3, 14]. The design and training of constrained neural networks to satisfy the averagedness or firm nonexpansiveness conditions is a challenging task and is an emerging direction of research [22, 14].

In this work, we look at the problem of training 1-Lipschitz[1] (nonexpansive) neural networks in order to construct averaged denoisers. Several techniques have been proposed in the literature to control the Lipschitz constant of ReLU networks such as spectral normalization [16] and Parseval frames [9]. However, it has been observed that constraining ReLU networks may lead to a drop in denoising performance. While the nonexpansiveness constraint reduces the capacity of the model, the performance drop could also be imputed to the choice of the model and the training scheme. Here, we propose to mitigate this drop in performance by using more adaptable models. In particular, we consider deep spline neural networks [23] where the activation functions are learnable linear splines. They are known to improve the performance relative to ReLU networks [4] and are amenable to a control of the Lipschitz constant [2]. We present an efficient way to train 1-Lipschitz deep spline neural networks using "slope normalization", which can be viewed as the counterpart of spectral normalization for activation functions. We then apply our trained denoisers to the PnP-FBS framework for compressed sensing MRI and show the benefits of our approach.

Contributions. We provide proof of concept that 1-Lipschitz deep spline neural network denoisers with learnable activation functions improve the performance of provably convergent PnP algorithms for ill-posed linear inverse problems.

2 Methods

In this work, we focus on the PnP-FBS algorithm. At each iteration, the estimate is updated as

$$\mathbf{x}^{(t+1)} = \mathrm{D}\big(\mathbf{x}^{(t)} - \alpha \nabla_{\mathbf{x}} E(\mathbf{y}, \mathbf{H}\mathbf{x}^{(t)})\big), \tag{2}$$

where $\mathrm{D}\colon \mathbb{R}^K \to \mathbb{R}^K$ is the plugged-in denoiser and $\alpha \in \mathbb{R}_+$ is the gradient step size. We assume that $E\colon \mathbb{R}^M \times \mathbb{R}^M \to \mathbb{R}_+$ is convex in the second argument and is differentiable with L-Lipschitz continuous gradient. It has been shown that for $\alpha \in (0, 2/L)$, PnP-FBS is guaranteed to converge to a fixed point if $\mathrm{D}\colon \mathbb{R}^K \to \mathbb{R}^K$ is an averaged operator $\mathrm{D} = \beta\mathrm{R} + (1-\beta)\mathrm{Id}$, where R is a 1-Lipschitz (nonexpansive) operator and $\beta \in (0, 1)$ [3, 14].

Keeping this convergence result in mind, we now consider the problem of training an averaged denoiser where R is a neural network whose Lipschitz constant must be constrained to be at most one. We take R to be a simple convolutional deep spline network [23] consisting of a series of alternating convolutional layers and pointwise nonlinear transformations. In contrast to the standard ReLU network, we have a learnable piecewise-linear spline activation function for each output channel of the preceding convolutional layer. The spline nonlinearities are represented using linear B-spline basis functions that are compactly supported, allowing for an efficient computation of the forward and backward passes during training [4] (see Appendix for more details).

The Lipschitz constant of the deep spline network is upper-bounded by the product of the Lipschitz constants of the individual layers. Therefore, in order to ensure that the network is 1-Lipschitz, we constrain the Lipschitz constants of each convolutional and spline activation functions to be at most one. To do this during the training process, we use the argument that if an operator T is L-Lipschitz,

[1] An operator $\mathrm{T}\colon \mathbb{R}^K \to \mathbb{R}^K$ is L-Lipschitz if $\|\mathrm{T}(\mathbf{x}) - \mathrm{T}(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|$ for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^K$. The smallest value of L is called the Lipschitz constant of T.

then the normalized operator $(1/L)\mathrm{T}$ is 1-Lipschitz. The convolution operation can be viewed as a matrix-vector product and its Lipschitz constant is the spectral norm (maximum singular value) of this matrix. We rely on the real spectral normalization (real-SN) method [17] to control the Lipschitz constant of the convolutional layers. Real-SN uses power iterations (without having to build the convolution matrix explicitly) to efficiently compute the maximum singular value and normalizes the convolution kernel accordingly. The Lipschitz constant of a linear spline function is the maximum absolute value of its derivative. Since our learnable activation functions have a finite number of pieces, we are able to compute the maximum absolute slope efficiently and normalize the function by this constant to ensure that it is 1-Lipschitz. We term this process "slope normalization": it is the activation-function counterpart of spectral normalization.

3 Experimental Results

We now present some experimental results to demonstrate the advantages of the 1-Lipschitz deep spline networks over the corresponding ReLU versions. We tackle the compressed sensing MRI (CS-MRI) problem [17] using the PnP-FBS algorithm with neural networks trained for Gaussian noise suppression that are constrained to be averaged operators. We first evaluate the performances of the deep spline and ReLU based denoisers in Section 3.1. The performances of these denoisers for the CS-MRI problem are reported in Section 3.2.

3.1 Evaluation of Gaussian Denoisers

We train Gaussian denoisers of the form $\mathrm{D} = \beta\mathrm{R} + (1-\beta)\mathrm{Id}$, with R being a CNN. The CNN is chosen to be of the form $\mathrm{R}(\mathbf{x}) = \mathbf{C}_Q \circ \cdots \circ \boldsymbol{\sigma}_q \circ \mathbf{C}_q \circ \cdots \circ \boldsymbol{\sigma}_1 \circ \mathbf{C}_1(\mathbf{x})$, where $\mathbf{C}_q$ denotes a linear convolutional layer and $\boldsymbol{\sigma}_q$ denotes a pointwise nonlinear activation function. We consider three different denoisers characterized by three CNNs: an unconstrained ReLU network (ReLU-U), a 1-Lipschitz ReLU network (ReLU-L), and a 1-Lipschitz deep spline network (DS-L). Because the ReLU-U denoiser is not guaranteed to be an averaged operator (unconstrained CNN), it only serves as a baseline for comparing constrained cases (ReLU-L and DS-L). We fix $\beta = 0.5$ for all models. This value could have been optimized by grid search, but the goal is to compare the relative performance between ReLU-L and DS-L.

The training dataset consists of 400 images from the BSD500 dataset [1] divided into $238,400$ non-overlapping patches of size 40×40. From the remaining images, we create a validation dataset of 12 images and a test dataset of 68 images. The images take values between $[0, 1]$. We consider three noise levels $\sigma = 5/255, 10/255$ and $20/255$, where σ is the standard deviation of the Gaussian noise added to the datasets. For each level of noise, we trained all denoising models considered (ReLU-U, ReLU-L, and DS-L) with $Q = \{3, 5, 7, 9\}$ (number of layers). The kernels for the convolutional layers are of size (3×3) and the number of channels in the intermediate convolutional layers is fixed to 32. For DS-L, the grid size for the learnable nonlinearities is 0.2 and the number of knots is 51 (refer to Appendix A for details). All denoisers were trained for 200 epochs using an ADAM optimizer [15] with a learning rate of 10^{-5} and a batch size of 128. The loss function is the mean-square error (MSE). For DS-L, the loss function has a regularization term that penalizes the second-order derivative of the learnable spline activation functions (refer to Appendix A for details). The corresponding regularization parameter is set to 10^{-5}.

The denoising results for $\sigma = 5/255$ are shown in Figure 1 while the results for $\sigma = 10/255$ and $20/255$ are shown in Appendix B (Figures 2 and 3). As expected, we observe that for any value of σ and Q, enforcing the 1-Lipschitz condition on the ReLU CNN leads to a drop in performance (ReLU-U vs. ReLU-L in Figure 1). The DS-L denoiser is a more adaptable model and improves in performance over the ReLU-L denoiser. Interestingly, the DS-L denoiser with only three layers consistently outperforms the ReLU-L denoiser with nine layers even though it has fewer learnable parameters. This shows that increasing the capacity of the model by learning activation functions seems to be more beneficial than increasing the number of layers for such constrained training tasks.

3.2 Compressed Sensing MRI

We now look at the CS-MRI problem of recovering an image $\mathbf{x} \in \mathbb{R}^K$ from its measurements $\mathbf{y} = \mathbf{M}\mathbf{F}\mathbf{x} + \mathbf{n} \in \mathbb{C}^M$, where $\mathbf{M}$ is a subsampling mask (identity matrix with some missing entries),

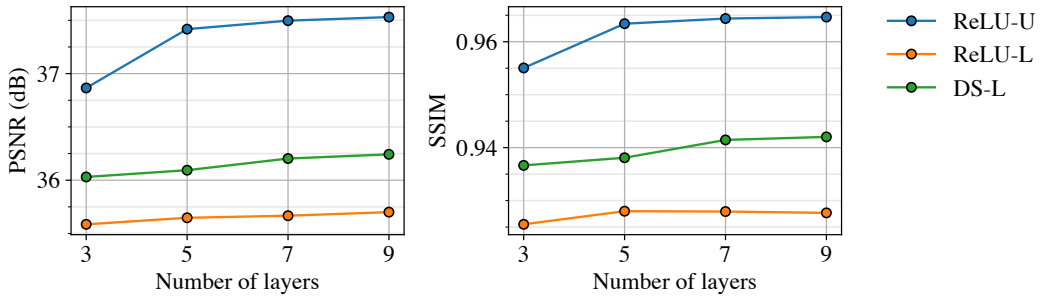

Figure 1: Denoising test performances (PSNR and SSIM) of the three models considered (ReLU-U, ReLU-L, and DS-L) for the noise level $\sigma = 5/255$.

Table 1: PSNR values for the CS-MRI experiment for $\sigma_{\mathbf{n}} = 10/255$, $Q = 5$, and $\sigma = 5/255$

Subsampling mask	Random		Radial		Cartesian	
Image type	Brain	Bust	Brain	Bust	Brain	Bust
Zero-filling	12.72	11.49	12.15	9.51	10.99	9.15
ReLU-L	20.25	17.05	19.02	16.22	13.70	12.85
DS-L	20.78	17.76	19.51	17.00	14.23	13.53

$\mathbf{F}$ is the discrete Fourier transform and $\mathbf{n}$ is a complex Gaussian noise with variance $\sigma_{\mathbf{n}}^2$ (not to be confused with σ, the noise variance used for training the denoisers) for both the real and imaginary parts. We use PnP-FBS with the trained denoisers described in the previous section to solve this inverse problem. Since we are interested in a convergent PnP-FBS algorithm, we compare the performances of the ReLU-L and DS-L denoisers only.

The data-fidelity term is chosen as $E(\mathbf{y}, \mathbf{MFx}) = (1/2)\|\mathbf{y} - \mathbf{MFx}\|^2$. We run the CS-MRI experiment on two images of size 256×256 and which take values between $[0, 1]$. We consider three kinds of subsampling masks (random, radial and cartesian) [17], each with a subsampling ratio of 0.3. Further, we also consider three levels of noise $\sigma_{\mathbf{n}} = 10/255, 20/255$, and $30/255$. The step size α for PnP-FBS is set to 10^{-5}. This value is chosen based on the Lipschitz constant of the gradient of the data-fidelity term (Section 2). The PnP-FBS algorithm is initialized with the zero-filled estimate, obtained by taking the inverse discrete Fourier transform of the zero-filled subsampled measurements.

We use ReLU-L and DS-L denoisers with $Q = 5$. For each experimental setup (subsampling mask and image type), we evaluate the performance of these denoisers in terms of PSNR for different values of σ (denoising strength) and report the best models. The results for $\sigma_{\mathbf{n}} = 10/255$ are presented in Table 1. The results for the other two noise levels as well the reconstructed images are provided in Appendix B (Tables 2 and 3 and Figures 4, 5 and 6). We observe that the DS-L denoiser systematically yields better image reconstructions than the ReLU-L denoiser. Thus, the improvement in denoising performance is translated to the CS-MRI problem, highlighting the benefits of the proposed approach.

4 Conclusion

In this work, we looked at the problem of building averaged (denoising) operators using 1-Lipschitz CNNs for provably convergent PnP algorithms that are used for solving ill-posed linear inverse problems. To mitigate the drop in performance resulting from such a constraint, we proposed the use of deep spline networks whose piecewise-linear spline activation functions can be trained while guaranteeing a controlled Lipschitz constant. To this end, we introduced "slope normalization" as the counterpart to spectral normalization. We showed that averaged denoising operators built from 1-Lipschitz deep spline networks consistently outperformed those built from 1-Lipschitz ReLU networks for both denoising and CS-MRI. Our findings suggest that deep spline networks have higher expressivity under Lipschitz constraints than ReLU networks, even with fewer trainable parameters.

References

[1] Pablo Arbeláez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, May 2011.

[2] Shayan Aziznejad, Harshit Gupta, Joaquim Campos, and Michael Unser. Deep neural networks with trainable activations and controlled Lipschitz constant. *IEEE Transactions on Signal Processing*, 68:4688–4699, August 2020.

[3] Heinz H. Bauschke and Patrick L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. CMS Books in Mathematics. Springer, Cham, 2nd edition, February 2017.

[4] Pakshal Bohra, Joaquim Campos, Harshit Gupta, Shayan Aziznejad, and Michael Unser. Learning activation functions in deep (spline) neural networks. *IEEE Open Journal of Signal Processing*, 1:295–309, November 2020.

[5] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, (1):1–122, 2011.

[6] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. Non-local means denoising. *Image Processing On Line*, 1:208–212, 2011.

[7] Gregery T. Buzzard, Stanley H. Chan, Suhas Sreehari, and Charles A. Bouman. Plug-and-play unplugged: Optimization-free reconstruction using consensus equilibrium. *SIAM Journal on Imaging Sciences*, 11(3):2001–2020, 2018.

[8] Stanley H Chan, Xiran Wang, and Omar A Elgendy. Plug-and-play ADMM for image restoration: Fixed-point convergence and applications. *IEEE Transactions on Computational Imaging*, 3(1):84–98, 2016.

[9] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *International Conference on Machine Learning*, pages 854–863. PMLR, 2017.

[10] Patrick L. Combettes and Valérie R. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005.

[11] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, August 2007.

[12] Ruturaj G. Gavaskar and Kunal N. Chaudhury. Plug-and-play ISTA converges with kernel denoisers. *IEEE Signal Processing Letters*, 27:610–614, 2020.

[13] Shuhang Gu, Lei Zhang, Wangmeng Zuo, and Xiangchu Feng. Weighted nuclear norm minimization with application to image denoising. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2862–2869, 2014.

[14] Johannes Hertrich, Sebastian Neumayer, and Gabriele Steidl. Convolutional proximal neural networks and plug-and-play algorithms. *Linear Algebra and its Applications*, 631:203–234, December 2021.

[15] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[16] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

[17] Ernest Ryu, Jialin Liu, Sicheng Wang, Xiaohan Chen, Zhangyang Wang, and Wotao Yin. Plug-and-play methods provably converge with properly trained denoisers. In *International Conference on Machine Learning*, pages 5546–5557. PMLR, 2019.

[18] Suhas Sreehari, Singanallur V. Venkatakrishnan, Brendt Wohlberg, Gregery T. Buzzard, Lawrence F. Drummy, Jeffrey P. Simmons, and Charles A. Bouman. Plug-and-play priors for bright field electron tomography and sparse interpolation. *IEEE Transactions on Computational Imaging*, 2(4):408–423, 2016.

[19] Yu Sun, Brendt Wohlberg, and Ulugbek S. Kamilov. An online plug-and-play algorithm for regularized image reconstruction. *IEEE Transactions on Computational Imaging*, 5(3):395–408, 2019.

[20] Yu Sun, Zihui Wu, Xiaojian Xu, Brendt Wohlberg, and Ulugbek S. Kamilov. Scalable plug-and-play ADMM with convergence guarantees. *IEEE Transactions on Computational Imaging*, 7:849–863, 2021.

[21] Afonso M. Teodoro, José M. Bioucas-Dias, and Mário A. T. Figueiredo. Scene-adapted plug-and-play algorithm with convergence guarantees. In *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, 2017.

[22] Matthieu Terris, Audrey Repetti, Jean-Christophe Pesquet, and Yves Wiaux. Building firmly nonexpansive convolutional neural networks. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8658–8662. IEEE, 2020.

[23] Michael Unser. A representer theorem for deep neural networks. *Journal of Machine Learning Research*, 20(110):1–30, 2019.

[24] Singanallur V. Venkatakrishnan, Charles A. Bouman, and Brendt Wohlberg. Plug-and-play priors for model based reconstruction. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 945–948, 2013.

[25] Dong Hye Ye, Somesh Srivastava, Jean-Baptiste Thibault, Ken Sauer, and Charles Bouman. Deep residual learning for model-based iterative CT reconstruction using plug-and-play framework. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6668–6672, 2018.

A Deep Spline Neural Networks

In deep spline neural networks, besides the linear weights or convolution kernels, we also have learnable nonlinearities that are piecewise-linear splines. This choice stems from a functional formulation of the training of a neural network with free-form activation functions. If the cost functional for the training process includes a regularization term that penalizes the second-order total-variations of the activation functions, then the optimal activation functions are known to be adaptive piecewise-linear splines [23].

A.1 B-Spline Representation

For each nonlinearity in the network, we consider a linear spline s with K knots on a finite grid of size T. We assume that K is odd. Let $k_{\text{min}} = -(K-1)/2$ and $k_{\text{max}} = (K-1)/2$. The function s is then represented as

$$s(x) = \begin{cases} c_{k_{\text{min}}} + \frac{1}{T}(c_{k_{\text{min}}} - c_{k_{\text{min}}-1})(x - k_{\text{min}}T), & x \in (-\infty, k_{\text{min}}T) \\ \sum\limits_{k=k_{\text{min}-1}}^{k_{\text{max}+1}} c_k \varphi_T(x - kT), & x \in [k_{\text{min}}T, k_{\text{max}}T] \\ c_{k_{\text{max}}} + \frac{1}{T}(c_{k_{\text{max}}+1} - c_{k_{\text{max}}})(x - k_{\text{max}}T), & x \in (k_{\text{max}}T, \infty), \end{cases} \tag{3}$$

where φ_T is the triangle-shaped B-spline

$$\varphi_T(x) = \begin{cases} 1 - \left|\frac{x}{T}\right|, & -T \leq x \leq T, \\ 0, & \text{otherwise}, \end{cases} \tag{4}$$

and the B-spline coefficients $\mathbf{c} = (c_k)$ are the adjustable quantities during the training process. The last two coefficients on either side - $(k_{\text{min}-1}, k_{\text{min}})$ and $(k_{\text{max}}, k_{\text{max}+1})$ - are responsible for handling the linear extensions beyond the interval $[k_{\text{min}}T, k_{\text{max}}T]$.

The second-order total-variations for the regularization term in the cost function can be computed as $\text{TV}^{(2)}(s) = \|\mathbf{L}\mathbf{c}\|_1$, where

$$\mathbf{L} = \frac{1}{T} \begin{bmatrix} 1 & -2 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & -2 & 1 & 0 & \cdots & 0 \\ \vdots & & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & 1 & -2 & 1 \end{bmatrix}. \tag{5}$$

This regularization term promotes sparsification of the knots in the linear spline.

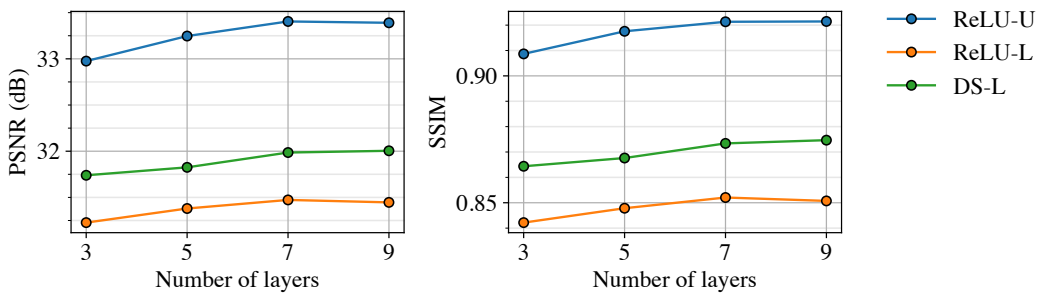

Figure 2: Denoising test performances (PSNR and SSIM) of the three models considered (ReLU-U, ReLU-L, and DS-L) for the noise level $\sigma = 10/255$.

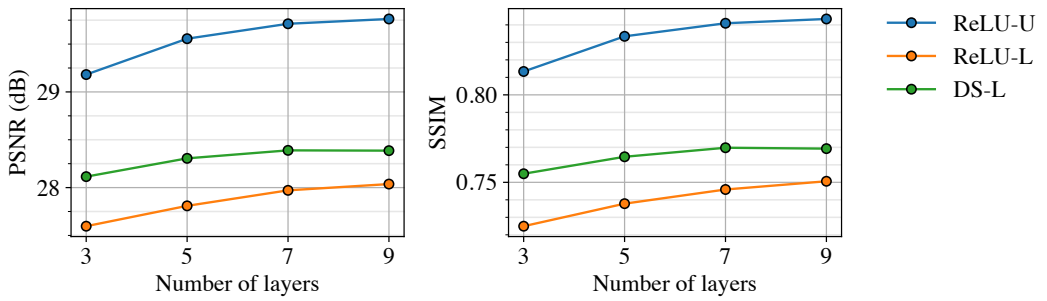

Figure 3: Denoising test performances (PSNR and SSIM) of the three models considered (ReLU-U, ReLU-L, and DS-L) for the noise level $\sigma = 20/255$.

A.2 Computation of the Lipschitz Constant

The Lipschitz constant of the linear spline s is the maximum absolute value of its gradient and can be computed as $s_{\text{Lip}} = \|\mathbf{D}\mathbf{c}\|_\infty$, where

$$\mathbf{D} = \frac{1}{T}\begin{bmatrix} -1 & 1 & 0 & 0 & \cdots & 0 \\ 0 & -1 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & -1 & 1 \end{bmatrix}. \tag{6}$$

Thus, the Lipschitz constant of the linear spline activation functions can be computed in an efficient manner in order to perform slope normalization.

B Additional Results

Here, we present additional results for both the Gaussian denoising and compressed sensing MRI tasks.

B.1 Evaluation of Gaussian Denoisers

Figures 2 and 3 show the denoising results for Gaussian noises with variances $\sigma = 10/255$ and $\sigma = 20/255$, respectively.

B.2 Compressed Sensing MRI

In Tables 2 and 3, we present the results for CS-MRI for $\sigma_{\mathbf{n}} = 20/255$ and $\sigma_{\mathbf{n}} = 30/255$, respectively. Also, we show the reconstructed images for all the different settings in Figures 4, 5 and 6.

Table 2: PSNR values for the CS-MRI experiment for $\sigma_{\mathbf{n}} = 20/255$, $Q = 5$, and $\sigma = 5/255$

Subsampling mask	Random		Radial		Cartesian	
Image type	Brain	Bust	Brain	Bust	Brain	Bust
Zero-filling	11.27	9.21	10.86	7.95	9.98	7.70
ReLU-L	18.21	14.93	17.40	14.39	13.23	12.10
DS-L	18.60	15.55	17.84	15.00	13.73	12.67

Table 3: PSNR values for the CS-MRI experiment for $\sigma_{\mathbf{n}} = 30/255$, $Q = 5$, and $\sigma = 5/255$

Subsampling mask	Random		Radial		Cartesian	
Image type	Brain	Bust	Brain	Bust	Brain	Bust
Zero-filling	9.58	7.04	9.27	6.18	8.66	6.01
ReLU-L	16.09	12.82	15.51	12.56	12.55	11.02
DS-L	16.39	13.27	15.83	13.04	12.94	11.48

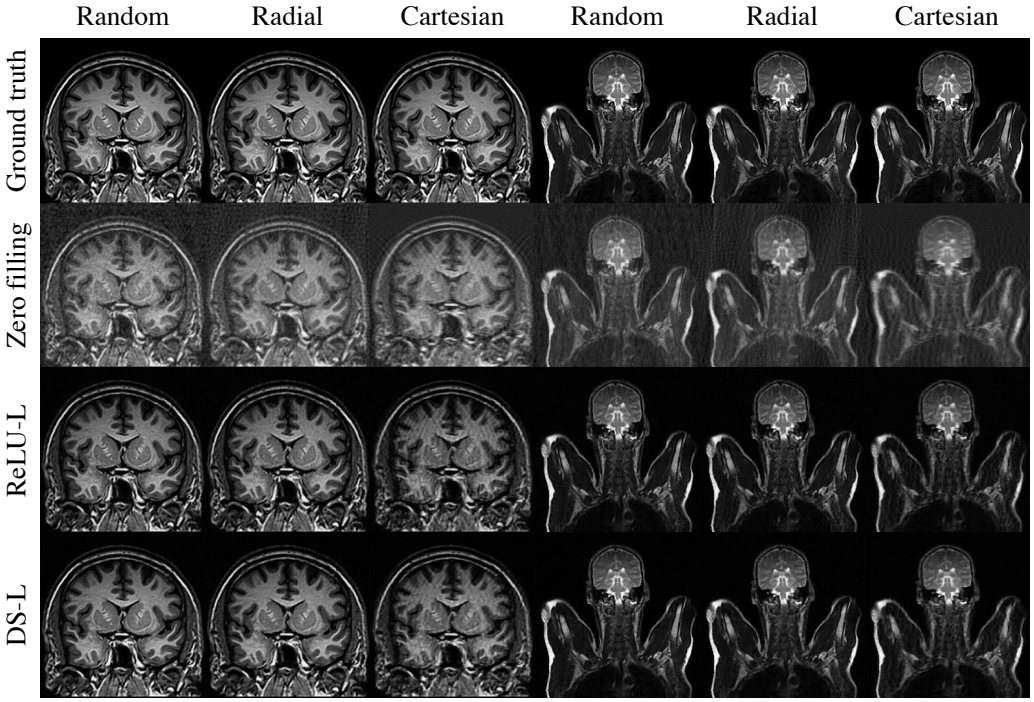

Figure 4: Two types of images (brain and bust) reconstructed for the CS-MRI experiment with three subsampling masks (random, radial and Cartesian) for $\sigma_{\mathbf{n}} = 10/255$ and $\sigma = 5/255$.

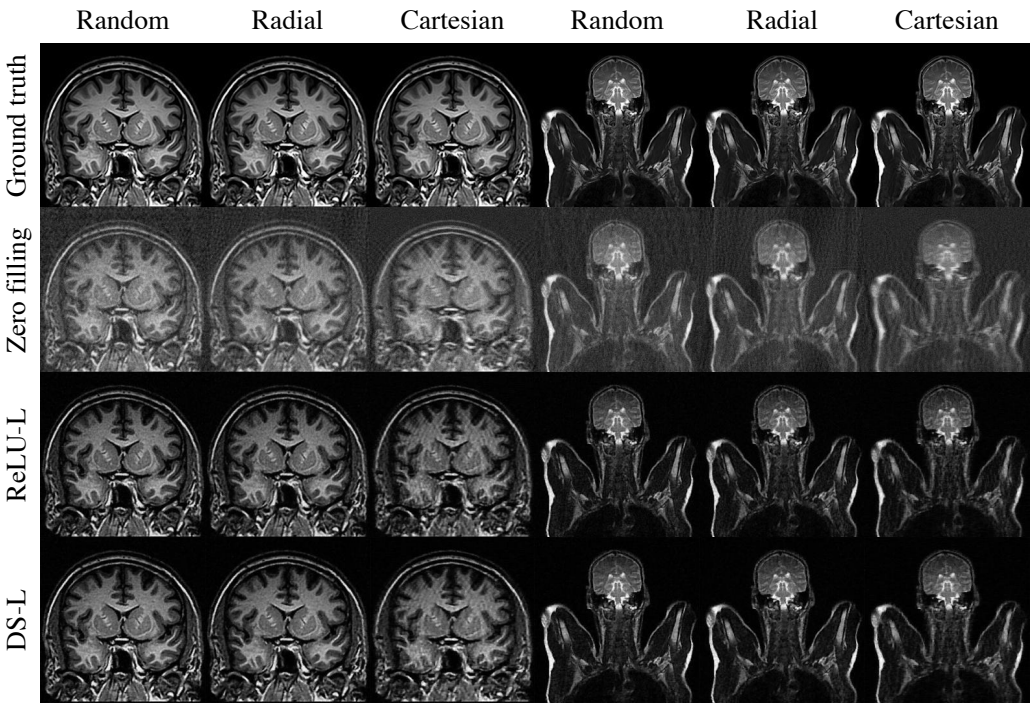

Figure 5: Two types of images (brain and bust) reconstructed for the CS-MRI experiment with three subsampling masks (random, radial and Cartesian) for $\sigma_{\mathbf{n}} = 20/255$ and $\sigma = 5/255$.

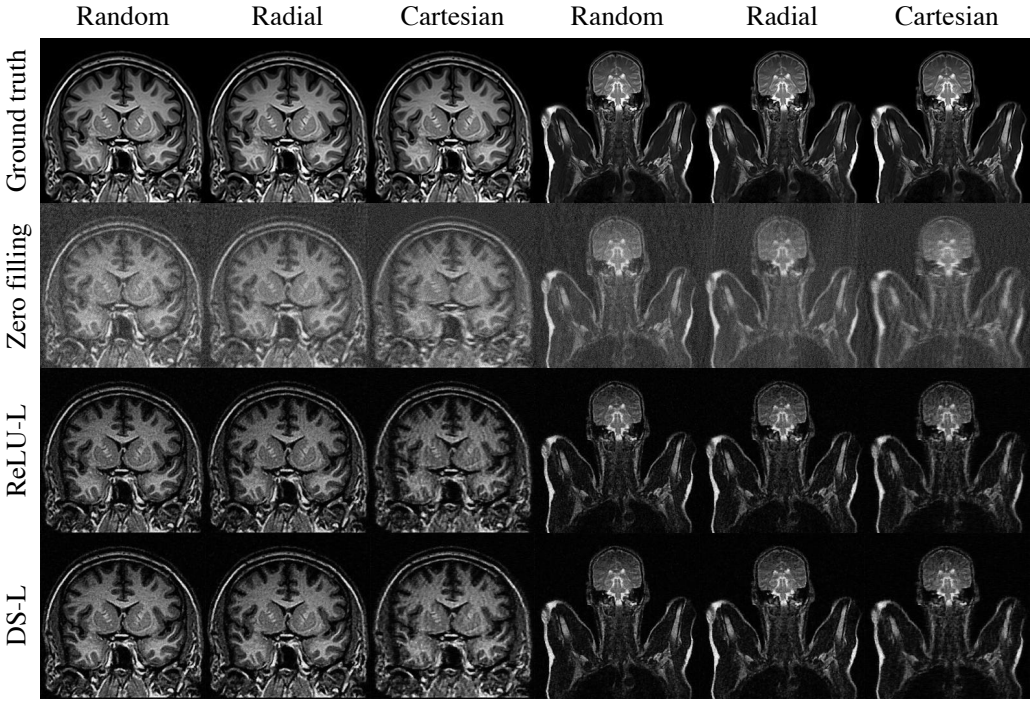

Figure 6: Two types of images (brain and bust) reconstructed for the CS-MRI experiment with three subsampling masks (random, radial and Cartesian) for $\sigma_{\mathbf{n}} = 30/255$ and $\sigma = 5/255$.