Building firmly nonexpansive convolutional neural networks

Matthieu Terris, Audrey Repetti, Jean-Christophe Pesquet, Yves Wiaux

BUILDING FIRMLY NONEXPANSIVE CONVOLUTIONAL NEURAL NETWORKS

Matthieu Terris[⋆⋆]*, Audrey Repetti*[⋆◇]*, Jean-Christophe Pesquet*[‡†]*, and Yves Wiaux*[⋆]

[⋆] Institute of Sensors, Signals and Systems, Heriot-Watt University, Edinburgh EH14 4AS, United Kingdom
[◇] Department of Actuarial Mathematics & Statistics, Heriot-Watt University, Edinburgh EH14 4AS, United Kingdom
[‡] Université Paris-Saclay, CentraleSupélec, Inria, Center for Visual Computing, 91190 Gif sur Yvette, France

ABSTRACT

Building nonexpansive Convolutional Neural Networks (CNNs) is a challenging problem that has recently gained a lot of attention from the image processing community. In particular, it appears to be the key to obtain convergent Plug-and-Play algorithms. This problem, which relies on an accurate control of the the Lipschitz constant of the convolutional layers, has also been investigated for Generative Adversarial Networks to improve robustness to adversarial perturbations. However, to the best of our knowledge, no efficient method has been developed yet to build nonexpansive CNNs. In this paper, we develop an optimization algorithm that can be incorporated in the training of a network to ensure the nonexpansiveness of its convolutional layers. This is shown to allow us to build firmly nonexpansive CNNs. We apply the proposed approach to train a CNN for an image denoising task and show its effectiveness through simulations.

***Index Terms*—** Neural networks, optimization, monotone operators, nonexpansive operator, image restoration.

1. INTRODUCTION

Plug-and-Play (PnP) methods have shown competitive results with state-of-the-art methods in image recovery [1–3]. These approaches consist in replacing the proximity operator in some optimization algorithms, such as ADMM [1] or forward-backward [4], by a denoiser often being a neural network (NN). Thus, these hybrid approaches take advantage of both the good image approximation properties of NNs, and the robustness of optimization methods. In particular, it is shown that a sufficient condition to get the convergence of PnP iterates is to ensure the firm nonexpansiveness of the denoiser [4]. Unfortunately, this assumption is generally not met in practice [5].

As a first contribution, leveraging monotone operator theory, we show that a firmly nonexpansive network can be obtained by building a nonexpansive network, i.e. a 1-Lipschitz network. Nevertheless, building such network is still a non trivial problem. This problem was investigated recently for Generative Adversarial Networks (GANs), where it was shown that controlling the Lipschitz constant improves the robustness to adversarial perturbations. One of the first dedicated study appeared with Parseval networks [6], where the Lispchitz constant is limited by regularizing the weights of the convolutional layers. This approach was further improved in [7], where the authors propose to iterate a projection on the Stiefel manifold, and in [8, 9], where the weight matrices are normalized by their spectral norm. However, limiting the spectral norm (or equivalently, the Lipschitz constant) of the weight matrices does not allow to constrain accurately the Lipschitz constant of the associated convolutional layer, making the building of nonexpansive convolutional layers difficult. Other approaches, developed in [10, 11], mention conditions in the Fourier domain in order to compute the singular values of convolutional layers.

The main contribution of this work is to develop a method to build nonexpansive networks, by tightly constraining the Lipschitz constant of feedforward CNNs to be smaller than 1. The proposed approach is based on a proximal optimization algorithm, namely the Douglas-Rachford (DR) algorithm [12, 13], which can directly be introduced as a projection step during training. We showcase our method with the training of a feedforward CNN as a denoiser. Unlike state-of-the-art normalization techniques [6, 8, 9], our Lipschitz constraint ensures the firm nonexpansiveness of the resulting denoiser.

The remainder of the paper is organised as follows: in Section 2 we describe the necessary conditions to build a nonexpansive network. The proposed algorithm to impose the nonexpansive condition is provided in Section 3. Simulation results for a denoising problem are given in Section 4. Finally, we draw our conclusions in Section 5.

2. NONEXPANSIVE NETWORKS

Let $(\mathcal{H}_j)_{0\leq j\leq m}$ be non-zero real-valued Hilbert spaces. Feedforward neural networks can be decomposed under the form $Q = T_m \circ \cdots \circ T_1$ with $(T_j)_{1\leq j\leq m}$ being the layers of

[⋆⋆]This work was partly supported by the UK Engineering and Physical Sciences Research Council (EPM008843/1 and EP/M019306/1) and used the Cirrus UK National Tier-2 HPC Service at EPCC (http://www.cirrus.ac.uk) funded by the University of Edinburgh and EPSRC (EP/P020267/1).

[†]This work was partly supported by Institut Universitaire de France

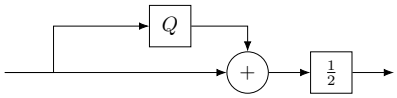


Fig. 1: Architecture of a firmly nonexpansive network given a nonexpansive network Q.

the network defined, for every $j \in \{1, \dots, m\}$, by

$$T_j : \mathcal{H}_{j-1} \to \mathcal{H}_j \colon x \mapsto R_j(W_j x + b_j), \tag{1}$$

where R_j is a non linear activation operator, W_j is a linear operator, and $b_j \in \mathcal{H}_j$ [5, 14]. In the following we will focus on the case when $(W_j)_{1 \leq j \leq m}$ are convolutional operators defined on suitable signal spaces.

Following the approach of [5], we see a network G as the resolvent of a multivalued operator acting on $\mathcal{H} = \mathcal{H}_0$. Any multivalued operator is fully characterized by its resolvent, that is for $A \colon \mathcal{H} \rightrightarrows \mathcal{H}$, the resolvent of A is $J_A = (\mathrm{Id} + A)^{-1}$, the inverse being here defined in the sense of the inversion of the graph of the operator. We refer the reader to [15] for a background on monotone operators. A main property for our purpose is the following one:

Proposition 2.1. *Let $A \colon \mathcal{H} \rightrightarrows \mathcal{H}$. A is a maximally monotone operator if and only if its resolvent is firmly nonexpansive, i.e. there exists a nonexpansive operator $Q \colon \mathcal{H} \to \mathcal{H}$ such that*

$$J_A \colon \mathcal{H} \to \mathcal{H} \colon x \mapsto \frac{x + Q(x)}{2}. \tag{2}$$

In turn, $A = 2(\mathrm{Id} + Q)^{-1} - \mathrm{Id}$.

Equation (2) shows that given a nonexpansive operator Q, one can deduce a firmly nonexpansive one. The architecture of our firmly nonexpansive neural network is depicted in Fig. 1. As a consequence of Proposition 2.1, by training the network $G = (\mathrm{Id} + Q)/2$ while ensuring the nonexpansiveness of Q, we end up with a firmly nonexpansive structure. By doing so, we actually learn the resolvent J_A of a maximally monotone operator A, which has an interesting echo in the literature [2] where it is usual to replace the proximity operator in some algorithm by a neural network. It is interesting to highlight that the proximity operator is a special case of the resolvent of a maximally monotone operator. Thus, our study goes in the same direction, but in a more general setting.

A sufficient condition to ensure the nonexpansiveness of a feedforward NN is to ensure that, for every layer j, the operator T_j has a Lipschitz constant L_j lower than 1. We emphasize that we restrict our attention to *feedforward* networks, which means that we do not use any other skip or residual connections than the one already present in Fig. 1. In order to constrain the Lipschitz constant of a network, classical approaches in the literature consist of adding some constraints on the weight operators or the convolution kernels [6–9]. This however leads to Lipschitz constant estimates which are either loose or difficult to compute, especially for 2D applications. The following proposition, which follows from standard signal processing arguments, links the kernel and the Lipschitz constant of a convolutional layer.

Proposition 2.2. *Let $\mathcal{H} = \ell^2(\mathbb{Z}^2)$ be the space of square summable discrete 2D fields defined on $\mathbb{Z}^2$. Let $T = R(W \cdot + b)$ where $R \colon \mathcal{H}^d \to \mathcal{H}^d$ is nonexpansive, $b \in \mathcal{H}^d$, and W is a 2D-convolutive operator with $c \in \mathbb{N}^*$ input channels and $d \in \mathbb{N}^*$ output channels, that is*

$$W \colon \mathcal{H}^c \to \mathcal{H}^d \colon (x_k)_{1 \leq k \leq c} \mapsto \Big(\sum_{k=1}^{c} h_{l,k} * x_k \Big)_{1 \leq l \leq d}, \tag{3}$$

where $(h_{l,k})_{1 \leq k \leq c, 1 \leq l \leq d}$ are finite 2D kernels. At each frequency $\nu \in [0, 1[^2$, let $\boldsymbol{H}(\nu) \in \mathbb{C}^{d \times c}$ denote the multi-input multi-output (MIMO) frequency response associated with these kernels. Then, T is nonexpansive if

$$(\forall \nu \in [0, 1[^2) \quad \|\boldsymbol{H}(\nu)\|_{\mathrm{S}} \leq 1, \tag{4}$$

where $\| \cdot \|_{\mathrm{S}}$ denotes the spectral norm.

In the following, we will focus on 2D convolutional layers having a single input, i.e. $c = 1$.

3. ALGORITHMS FOR IMPOSING THE NONEXPANSIVE CONDITION

In this section we provide a method to constrain the convolutive layer to be nonexpansive, relying on Proposition 2.2.

Let $\mathbf{h} = (h_l)_{1 \leq l \leq d}$ be the kernels of a convolutive layer with one input and d outputs. We will view these kernels as elements of the space $\mathcal{H} = \ell^2(\mathbb{Z}^2)$. We consider the following constraint set:

$$\mathcal{C} = \Big\{ \mathbf{h} \in \mathcal{H}^d \mid (\forall \nu \in [0, 1[^2) \sum_{l=1}^{d} |\mathcal{F}(h_l)(\nu)|^2 \leq 1 \Big\}, \tag{5}$$

where $\mathcal{F}$ denotes the 2D-Fourier transform defined on $\mathcal{H}$. In addition, each kernel is constrained to correspond to a finite impulse response filter, that is to belong to

$$\mathcal{D} = \Big\{ h \in \mathcal{H} \mid (\forall p \notin \{0, \dots, s_1 - 1\} \times \{0, \dots, s_2 - 1\}) \quad h(p) = 0 \Big\}, \tag{6}$$

where $(s_1, s_2) \in (\mathbb{N}^*)^2$ define the 2D support of the filters.

In order to constrain the convolutive layer to be nonexpansive, we propose to compute the projection of a kernel $\overline{\mathbf{h}} \in \mathcal{H}^d$ onto $\mathcal{C} \cap \mathcal{D}^d$. This will enable to satisfy condition (4) of Proposition 2.2. The projection problem is equivalent to

$$\underset{\mathbf{h} = (h_l)_{1 \leq l \leq d}}{\text{minimize}} \quad \iota_{\mathcal{C}}(\mathbf{h}) + \sum_{l=1}^{d} \iota_{\mathcal{D}}(h_l) + \frac{1}{2} \|\mathbf{h} - \overline{\mathbf{h}}\|_2^2, \tag{7}$$

Algorithm 1: DR algorithm to solve (7)

Initialization: Let $\epsilon \in]0,1]$ and $\mathbf{k}_0 \in \mathcal{H}^d$.
Iterations:
for $t = 0, 1, \ldots,$ **do**
$\mathbf{h}_t = \left(\mathsf{P}_{\mathcal{D}}(k_{t,l})\right)_{1 \leq l \leq d}$,
$\lambda_t \in [\epsilon, 2-\epsilon]$,
$\mathbf{k}_{t+1} = \mathbf{k}_t + \lambda_t \Big(\mathsf{P}_{\mathcal{C}} \left((2\mathbf{h}_t - \mathbf{k}_t + \overline{\mathbf{h}})/2 \right) - \mathbf{h}_t \Big)$.

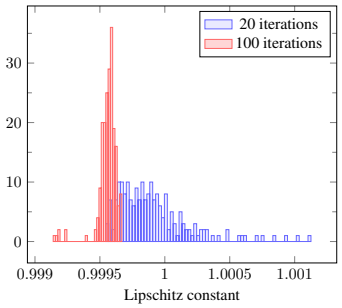


Fig. 2: Histogram of the Lipschitz constant of a convolution operator considering randomly sampled kernels of support size $s_1 \times s_2 = 3 \times 3$, with $(c, d) = (1, 64)$ (i.e. `Conv2d(1,64,3)` using PyTorch implementation) after a normalization with Algorithm 1.

where $\iota_{\mathcal{S}}$ denotes the indicator function of a set $\mathcal{S}$ (equal to 0 when its argument belongs to $\mathcal{S}$, and $+\infty$ otherwise). This problem can be efficiently solved by the DR algorithm [12], described in Algorithm 1. Since $\mathcal{C}$ and $\mathcal{D}$ are non-empty, closed, convex sets, the sequence $(\mathbf{h}_n)_{n \in \mathbb{N}}$ generated by this algorithm is guaranteed to converge to a solution to (7) [13].

In the practical implementation of this method, it is important to note that the support of the kernels $(\mathbf{h}_n)_{n \in \mathbb{N}}$ is of size $L_1 \times L_2$ while a 2D Fast Fourier Transform (FFT) is used to approximate $\mathcal{F}$ on a fine enough spectral grid. The projection onto $\mathcal{D}$ is then performed by truncating the result of an inverse FFT and the projection onto $\mathcal{C}$ reduces to a finite set of projections onto unit balls in the frequency domain.

In Fig. 2, we show the histogram of the Lipschitz constants of one layer of a convolutional network Q, obtained after performing 20 and 100 iterations of Algorithm 1. This algorithm can be included as an additional projection step in a standard Stochastic Gradient Descent (SGD) algorithm to build a nonexpansive network Q.

4. AN APPLICATION TO DENOISING

We use the methodology described in the previous sections to develop a firmly nonexpansive network in the context of image denoising. To this aim, we use the facts that a firmly nonexpansive network corresponds to a resolvent operator (see Proposition 2.1) and that proximal operators are particular cases of resolvent operators. From a signal processing viewpoint, proximity operators can be interpreted as denoisers. We hence propose to train our CNN as a denoiser.

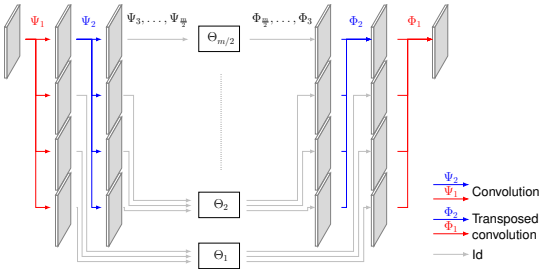


Fig. 3: Architecture of our convolutional neural network Q. The notation $Q_{\mathbf{d}}$ indicates the (ordered) number $\mathbf{d} = \{d_1, \ldots, d_{m/2}\}$ of output channels for each convolution; in this example we have $d_1 = d_2 = 4$. $q = \sum_{j=1}^{m/2} d_j - m/2 + 1$ is the total number of channels.

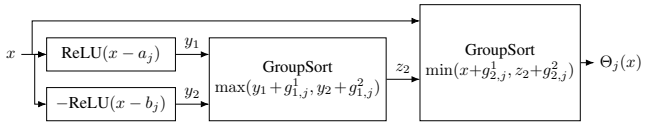


Fig. 4: Structure of a Θ_j nonlinearity; $a_j, b_j, g^1_{1,j}, g^2_{1,j}, g^1_{2,j}$ and $g^2_{2,j}$ are learnable parameters.

4.1. Architecture of the nonexpansive network

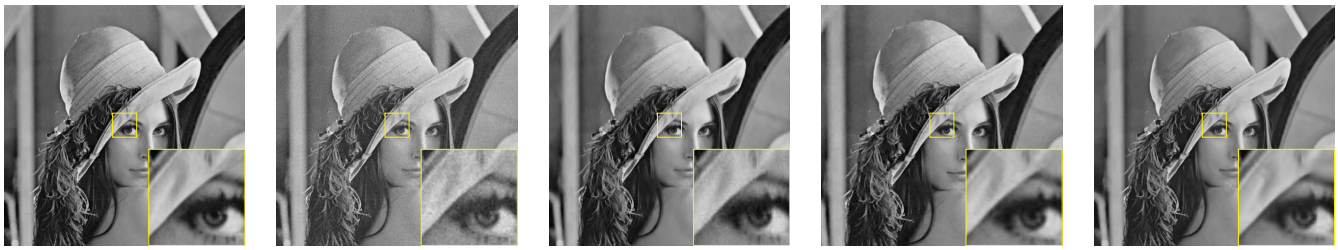
We consider images of dimension $n_1 \times n_2$, and we aim to build a nonexpansive convolutional network Q. The proposed network architecture consists of a succession of operators $(\Psi_j, \Theta_j, \Phi_j)_{1 \leq j \leq m/2}$ as depicted in Fig. 3, where m is even. For every $j \in \{1, \ldots, m/2\}$, $\Psi_j \colon \mathbb{R}^{n_1 n_2} \to (\mathbb{R}^{n_1 n_2})^{d_j}$ is a Single Input Multiple Output (SIMO) convolutive layer with d_j outputs, whereas $\Phi_j \colon (\mathbb{R}^{n_1 n_2})^{d_j} \to \mathbb{R}^{n_1 n_2}$ is a Multiple Input Single Output (MISO) convolutive layer with d_j inputs. The whole structure is an instance of the general model introduced in Section 2. To ensure the nonexpansiveness of Q, we build each Θ_j, Ψ_j and Φ_j as nonexpansive operators. Concerning $(\Theta_j)_{1 \leq j \leq m/2}$, we propose the structure described in Fig. 4, relying on both ReLU and GroupSort operations [7, 16]. According to Proposition 2.1, the resulting firmly nonexpansive network is then given by $G = (\mathrm{Id} + Q)/2$.

Note that denoising operators resulting from soft theshholding of the coefficients of the decomposition on a tight frame are particular case of G as defined by Fig. 3. Then, for every $j \in \{1, \ldots, m/2\}$, $\Psi_j = \Phi_j^\top$. In the case of wavelet frames, $m/2$ represents the number of resolution levels.

Dropping the indices j for clarity, by choosing the parameters in Fig. 4 as $a = -b = -g^1_1 = g^2_1 = \lambda > 0$, $g^1_2 = 2\lambda$, and $g^2_2 = 0$, the operator Θ boils down to perform a soft thresholding operator, i.e. $\Theta = 2\mathrm{soft}_{[-\lambda,\lambda]} - \mathrm{Id}$.

4.2. Training strategy

We train our network on the ILSVRC2012 ImageNet dataset [17], containing $N = 5 \times 10^4$ images. The training set is built as follows. We first convert each image in grayscale and randomly crop and resize it in patches of size 158×158. Secondly, for every image x_i, with $i \in \{1, \ldots, N\}$, we build



(a) (b) (c) (d) (e)

Fig. 5: (a) Original image `Lena` of size 512×512, (b) noisy image (PSNR $= 28.1$ dB), and reconstructions by (c) the proximal denoiser (PSNR $= 32.9$ dB, SSIM $= 0.75$), (d) our method (PSNR $= 33.9$ dB, SSIM $= 0.90$) and (e) BM3D (PSNR $= 35.1$ dB, SSIM $= 0.91$).

Algorithm 2: Algorithm to solve (8)

Initialization: Set $\mathbf{h}_0$ and $\mathbf{u}_0$.
Iterations:
for $t = 0, 1, \ldots,$ **do**

Randomly select $\mathcal{I}_t \subset \{1, \ldots, N\}$,

$$\mathbf{h}_{t+1} = \mathsf{P}_{\mathcal{C}\cap\mathcal{D}^d}\Big(\mathbf{h}_t - \gamma_{\mathbf{h},t}\sum_{i\in\mathcal{I}_t}\nabla_{\mathbf{h}}\phi_i(\mathbf{h}_t,\mathbf{u}_t) + \mu_t(\mathbf{h}_t - \mathbf{h}_{t-1})\Big),$$

$$\mathbf{u}_{t+1} = \mathbf{u}_t - \gamma_{\mathbf{u},t}\sum_{i\in\mathcal{I}_t}\nabla_{\mathbf{u}}\phi_i(\mathbf{h}_t,\mathbf{u}_t).$$

a noisy version y_i by adding to x_i a realization of an i.i.d. zero-mean Gaussian noise with variance $\sigma^2 = 0.04$ (PSNR $= 28.1$ dB). Eventually, to avoid border effects, the network output is cropped and its final size is $n_1 \times n_2 = 128 \times 128$. Only 49×10^3 images of the dataset are used for training while the 10^3 remaining images are used for testing. The same process as described above is applied to the testing set.

In our experiments, we choose Q with $m = 12$ and $\mathbf{d} = \{64, 64, 16, 16, 4, 4\}$ (see Fig. 3 and the description in Section 4.1). The firmly nonexpansive network $G_{(\mathbf{h}_j,\mathbf{u}_j)_{1\le j\le m}} = (\mathrm{Id}+Q)/2$ is parametrized by $(\mathbf{h}_j)_{1\le j\le m}$ and $(\mathbf{u}_j)_{1\le j\le m/2}$, where, for every $j \in \{1, \ldots, m\}$, $\mathbf{h}_j$ are the kernels of the convolutional layers $(\Psi_j, \Phi_j)_{1\le j\le m/2}$ in the nonexpansive network Q, and for every $j \in \{1, \ldots, m/2\}$, $\mathbf{u}_j = (a_j, b_j, g^1_{1,j}, g^1_{2,j}, g^2_{1,j}, g^2_{2,j})$ are the nonlinearity parameters of $(\Theta_j)_{1\le j\le m/2}$ defined in Fig. 4. We train the network as a denoiser, i.e. we

$$\underset{\mathbf{h},\mathbf{u}}{\text{minimize}} \sum_{i=0}^{N}\phi_i(\mathbf{h},\mathbf{u}) \tag{8}$$

$$\text{with} \quad \phi_i(\mathbf{h},\mathbf{u}) = \|x_i - G_{\mathbf{h},\mathbf{u}}(y_i)\|_2^2. \tag{9}$$

In a manner akin to [18,19], we use a heuristic projected SGD algorithm described in Algorithm 2, involving positive learning rates $(\gamma_t)_{t\in\mathbb{N}}$ and momentum parameters $(\mu_t)_{t\in\mathbb{N}}$. The projection is computed with Algorithm 1. In practice, we observe that 20 iterations of Algorithm 1 are sufficient to ensure the Lipschitz condition during training.

For SIMO (resp. MISO) convolutional layers with d_j output (resp. input) channels, we set the kernel size to $s \times s$ with $s = \sqrt{d_j}+1$. Convolutions are initialized with Hadamard filters, ensuring good denoising properties of G at initialization. We train the network for 20 epochs and decrease the learning rate every 3 epochs. Training plots are shown in Fig. 6.

	PSNR (dB)	SSIM	Firmly nonexpansive
Proximal method	31.4	0.86	yes
Proposed	32.6	0.90	yes
BM3D	33.3	0.91	no

Table 1: Results of the various methods over the BSDS300 dataset.

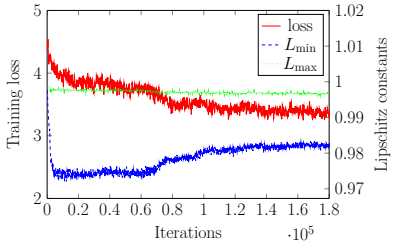


Fig. 6: Training loss (red), minimum (blue) and maximum (green) Lipschitz constant values of the convolutional layers during training.

4.3. Results

The average PSNR of the reconstructed images is 34.3 dB (resp. 34.2 dB) on the training (resp. testing) dataset. In addition, we compare our network with both a proximal method and the BM3D denoiser [20] on the BSDS300 image dataset [21]. For the proximal denoiser, we propose to compute $x^* = \mathrm{prox}_g(y)$ where $g = \iota_{[0,1]^{n_1 n_2}} + \eta\|\Lambda L \cdot\|_1, \eta > 0$ is a regularization parameter, Λ a positive weight matrix, and L a wavelet basis decomposition. This proximity operator does not admit a closed form solution but can be efficiently computed using a dual forward-backward algorithm [22], yielding a slower reconstruction time than with our method.

Table 1 shows that our firmly nonexpansive network provides better reconstruction results than the classical proximal denoiser and compares with BM3D in terms of SSIM. Visual results on the `Lena` test image are provided in Fig. 5.

5. CONCLUSION

In this paper, we developed a method to build firmly nonexpansive CNNs. Precisely, a proximal optimization algorithm is leveraged to perform the projection of the convolution kernels onto the underlying constraint set. In the context of an image denoising example, we plugged the proposed approach into the training of a feedforward CNN, to build a firmly nonexpansive denoiser. Future works include generalization to MIMO convolutions, in order to extend our method to any feedforward CNN architecture.

6. REFERENCES

[1] J. Rick Chang, C.-L. Li, B. Poczos, B. Vijaya Kumar, and A. C. Sankaranarayanan, "One network to solve them all–solving linear inverse problems using deep projection models," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5888–5897.

[2] T. Meinhardt, M. Moller, C. Hazirbas, and D. Cremers, "Learning proximal operators: Using denoising networks for regularizing inverse imaging problems," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1781–1790.

[3] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep cnn denoiser prior for image restoration," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3929–3938.

[4] E. Ryu, J. Liu, S. Wang, X. Chen, Z. Wang, and W. Yin, "Plug-and-play methods provably converge with properly trained denoisers," in *International Conference on Machine Learning*, 2019, pp. 5546–5557.

[5] P. L. Combettes and J.-C. Pesquet, "Deep neural network structures solving variational inequalities," *Set-Valued Var. Anal.*, to appear.

[6] M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier, "Parseval networks: Improving robustness to adversarial examples," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR.org, 2017, pp. 854–863.

[7] C. Anil, J. Lucas, and R. Grosse, "Sorting out lipschitz function approximation," *arXiv preprint arXiv:1811.05381*, 2018.

[8] Y. Yoshida and T. Miyato, "Spectral norm regularization for improving the generalizability of deep learning," *arXiv preprint arXiv:1705.10941*, 2017.

[9] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *International Conference on Learning Representations*, 2018.

[10] H. Sedghi, V. Gupta, and P. M. Long, "The singular values of convolutional layers," in *International Conference on Learning Representations*, 2019.

[11] A. Bibi, B. Ghanem, V. Koltun, and R. Ranftl, "Deep layers as stochastic solvers," in *International Conference on Learning Representations*, 2019.

[12] P. L. Combettes and J.-C. Pesquet, "A Douglas-Rachford splitting approach to nonsmooth convex variational signal recovery," *IEEE Journal of Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 564, 2007.

[13] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-point algorithms for inverse problems in science and engineering*, pp. 185–212. Springer, 2011.

[14] C. Bertocchi, E. Chouzenoux, M.-C. Corbineau, J.-C. Pesquet, and M. Prato, "Deep unfolding of a proximal interior point method for image restoration," *Inverse Problems*, 2019.

[15] H. H. Bauschke and P. L. Combettes, *Convex analysis and monotone operator theory in Hilbert spaces*, Springer, 2017.

[16] R. H. Hahnloser, R. Sarpeshkar, M. A. Mahowald, R. J. Douglas, and H. S. Seung, "Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit," *Nature*, vol. 405, no. 6789, pp. 947, 2000.

[17] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[18] L. Rosasco, S. Villa, and B. C. Vũ, "A stochastic inertial forward–backward splitting algorithm for multivariate monotone inclusions," *Optimization*, vol. 65, no. 6, pp. 1293–1314, 2016.

[19] P. L. Combettes and J.-C. Pesquet, "Stochastic forward-backward and primal-dual approximation algorithms with application to online image restoration," in *2016 24th European Signal Processing Conference (EUSIPCO)*. IEEE, 2016, pp. 1813–1817.

[20] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Transactions on image processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

[21] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*. IEEE, 2001, vol. 2, pp. 416–423.

[22] P. L. Combettes, Đ. Dũng, and B. C. Vũ, "Proximity for sums of composite functions," *Journal of Mathematical Analysis and applications*, vol. 380, no. 2, pp. 680–688, 2011.