Stability of Unfolded Forward-Backward to Perturbations in Observed Data

Cécile de Valle, Ezequiel Centofanti, Emilie Chouzenoux and Jean-Christophe Pesquet
Center for Visual Computing
Université Paris-Saclay, CentraleSupélec, Inria
Gif sur Yvette, France

***Abstract*—We consider a neural network architecture to solve inverse problems, which is built by unfolding a forward-backward algorithm. This algorithm is based on the minimization of an objective function which corresponds to a penalized least squares problem. In this context, ensuring stability is consistent with inverse problem theory since it guarantees both the continuity of the inversion method and its insensitivity to small noise. The latter is a critical property as deep neural networks have been shown to be vulnerable to adversarial perturbations. The main novelty of our work is to analyze the robustness of this inversion method with respect to a perturbation of the bias parameter of the network. In our architecture, the bias accounts for the observed data in the inverse problem. The analysis is performed by using tools of fixed point theory. Our theoretical results are illustrated by numerical simulations on a problem of signal restoration.**

***Index Terms*—neural networks, unfolding, stability, forward-backward algorithm, inverse problems**

I. Introduction

Inverse problems are commonly encountered in signal/image restoration [1], tomography [2], or inverse Laplace transform [3]. They consist in finding $x \in \mathscr{X}$ from observed data

$$y = Tx + w \tag{1}$$

where T is a bounded linear operator from a Hilbert space $\mathscr{X}$ to a Hilbert space $\mathscr{Y}$ and w corresponds to an additive measurement noise. The above problem is often ill-posed i.e., a solution might not exist, might not be unique, or might not depend continuously on the data.

The ill-posedness of the inverse problem can be addressed by regularization. Let $(\tau, \mu) \in]0, +\infty[^2$ be regularization parameters. Solving the inverse problem (1) often leads to the resolution of the following optimization problem:

$$\underset{x \in \mathscr{X}}{\text{minimize}}\, J_\tau(x) + \mu\, g(x) \;, \tag{2}$$

where

$$(\forall x \in \mathscr{X}) \quad J_\tau(x) = \frac{1}{2}\|Tx - y\|^2 + \frac{\tau}{2}\|Dx\|^2, \tag{3}$$

D is a bounded linear operator from $\mathscr{X}$ to some Hilbert space $\mathscr{Z}$, and g is a proper lower-semicontinuous convex function from $\mathscr{X}$ to $]-\infty, +\infty]$. To impose smoothness of the solution, D is usually chosen as a differential operator, while g may be the indicator function of a nonempty closed and convex set encoding some prior knowledge, e.g. some range constraint or sparsity pattern.

Optimization techniques [4] are classically used to solve Problem (2) but they require to set regularization parameters, which is a tedious task. In addition, optimization algorithms may be slow and their convergence behavior may strongly depend on the choice of some parameters.

The use of neural networks for solving inverse problems has become increasingly popular, especially in the image processing community. A rich panel of approaches have been proposed, either adapted to the sparsity of the data [5], [6], or mimicking variational models [7], [8], or iterating learned operators [9]–[13].

In iterative approaches, a regularization operator is learned, either in the form of a proximity operator as in [9], [10], [13], of a regularization term [14], of a pseudodiffential operator [15], or of its gradient [2], [16]. Strong connections also exist with Plug and Play methods [11], [17], [18], where the regularization operator is a pre-trained neural network.

Other recent works solve linear inverse problems by unrolling the optimization iterative process in the form of a network architecture as in [19], [20]. Here the number of iterations is fixed, instead of iterating until convergence, and the network is often trained in an end-to-end fashion. Since neural network frameworks offer powerful differential programming capabilities, they are also used for learning hyper-parameters in an unrolled optimization algorithm as in [21], [22].

All of the above strategies have shown very good numerical results. However, few studies have been conducted on their theoretical properties, especially their stability. In this paper, we propose an algorithm based on a neural network architecture to invert (1). One of its main advantages is that the structure of the neural network is interpretable and contains few parameters which are learned. We study the stability of the so-built neural network. The sensitivity analysis is performed with respect to the observed data y which correspond to a bias term in each of the layers of the proposed architecture. This analysis is more general than the one performed in [21], in which only the impact of the initialization was considered.

The outline of the paper is as follows: In Section II, we describe our proposed unrolled neural network architecture. In

The work by J.-C. Pesquet was supported by the ANR Chair in Artificial Intelligence BRIDGEABLE. E. Centofanti and E. Chouzenoux acknowledge support from the European Research Council Starting Grant MAJORIS ERC-2019-STG-850925.

Section III, we state our new stability results. In Section IV, we provide some numerical experiments, before giving some concluding remarks in Section V.

II. Proposed neural architecture

A. Unrolled forward-backward algorithm

We define the solution to the inverse problem (1) as the output of a neural network, whose structure is similar to a recurrent network [23], [24]. Namely, by setting an initial value x_0, we are interested in the following m-layers neural network with $m \in \mathbb{N} \setminus \{0\}$:

$$\begin{cases} \textbf{Initialization:} \\ \quad b_0 = T^* y, \\ \textbf{Layer } n \in \{1, \dots, m\}\textbf{:} \\ \quad x_n = R_n(W_n x_{n-1} + V_n b_0), \end{cases} \tag{4}$$

where, for every $n \in \{1, \dots, m\}$,

$$R_n = \mathrm{prox}_{\lambda_n \mu_n g}, \tag{5}$$
$$W_n = \mathbb{1} - \lambda_n T^* T - \lambda_n \tau_n D^* D, \tag{6}$$
$$V_n = \lambda_n \mathbb{1}. \tag{7}$$

Hereabove, prox_φ stands for the proximity operator of a lower-semicontinuous proper convex function φ [25, Chapter 9], $\mathbb{1}$ denotes the identity operator, and for every $n \in \{1, \dots, m\}$, λ_n, μ_n, and τ_n are positive constants, which are learned during training. Throughout this paper, L^* denotes the adjoint of a bounded linear operator L defined on Hilbert spaces.

Model (4) can be viewed as unrolling m iterations of an optimization algorithm. Inded, when $\mu_n \equiv \mu$ and $\tau_n \equiv \tau$, we recognize a forward-backward algorithm [4], [26] applied to variational problem (2).

B. Leakage factor

In order to gain more flexibility, we introduce positive multiplicative factors $(\eta_n)_{n \geq 1}$ on the bias. More specifically, we replace the operator V_n in (7) by

$$V_n = \lambda_n \eta_{n-1} \cdots \eta_1 \mathbb{1} \tag{8}$$

and $\eta_0 = 1$. When $n \geq 1$ and $\eta_n < 1$, the parameter η_n can be interpreted as a leakage factor. In the original forward-backward algorithm, the introduction of $(\eta_n)_{n \geq 1}$ amounts to introducing an error e_n in the gradient step, at iteration n, which is equal to $e_n = \lambda_n(\eta_{n-1} \cdots \eta_1 - 1) b_0$.

C. Virtual neural network

To facilitate our theoretical analysis, we will introduce a virtual network making use of new variables $(z_n)_{n \in \mathbb{N}}$. For every $n \in \mathbb{N} \setminus \{0\}$, we define the n-th layer of our virtual network by the following state-space model:

$$z_n = \begin{pmatrix} x_n \\ b_n \end{pmatrix}, \quad z_n = Q_n(U_n z_{n-1}), \tag{9}$$

with

$$\begin{cases} Q_n = \begin{pmatrix} R_n \\ \mathbb{1} \end{pmatrix}, \\ \\ U_n = \begin{pmatrix} W_n & \lambda_n \mathbb{1} \\ 0 & \eta_n \mathbb{1} \end{pmatrix}. \end{cases} \tag{10}$$

When we cascade the layers of the virtual neural network, the following triangular linear operator plays a prominent role:

$$U = U_m \circ \cdots \circ U_1 = \begin{pmatrix} W_{1,m} & \widetilde{W}_{1,m} \\ 0 & \eta_{1,m} \mathbb{1} \end{pmatrix}, \tag{11}$$

where, for every $n \in \{1, \dots, m\}$ and $i \in \{1, \dots, n\}$,

$$\widetilde{W}_{i,n} = \sum_{j=i}^{n} \lambda_j \eta_{i,j-1} W_{j+1,n} \tag{12}$$

and, for every $i \in \{1, \dots, m+1\}$ and $j \in \{0, \dots, m\}$,

$$W_{i,j} = \begin{cases} W_j \circ \cdots \circ W_i & \text{if } j \geq i \\ \mathbb{1} & \text{otherwise,} \end{cases} \tag{13}$$

$$\eta_{i,j} = \begin{cases} \eta_j \cdots \eta_i & \text{if } j \geq i \\ 1 & \text{otherwise.} \end{cases} \tag{14}$$

III. Stability analysis

A. Elements of fixed point theory

Our analysis will be grounded on tools of fixed point theory [27]. We recall some fundamental definitions.
Let us consider the Hilbert space $\mathscr{X}$ endowed with the norm $\|\cdot\|$ and the scalar product $\langle \cdot, \cdot \rangle$.
An operator $S\colon \mathscr{X} \to \mathscr{X}$ is θ-Lipschitz with $\theta \in]0, +\infty[$ if

$$(\forall (x, y) \in \mathscr{X} \times \mathscr{X}) \qquad \|Sx - Sy\| \leq \theta \|x - y\|.$$

If $\theta = 1$, S is said to be nonexpansive. Moreover, S is said to be α-averaged with $\alpha \in]0, 1[$ if, for every $(x, y) \in \mathscr{X} \times \mathscr{X}$,

$$\|Sx - Sy\|^2 + \frac{1 - \alpha}{\alpha} \|(\mathbb{1} - S)x - (\mathbb{1} - S)y\|^2 \leq \|x - y\|^2. \tag{15}$$

If S has a fixed point and it is averaged, then the iterates $(S^n x)_{n \in \mathbb{N}}$ converge weakly to a fixed point of S.
If $\alpha = 1/2$, we say that S is firmly nonexpansive. Let $\Gamma_0(\mathscr{X})$ be the set of proper lower semicontinuous convex functions from $\mathscr{X}$ to $]-\infty, +\infty]$. The proximity operator of any function $\varphi \in \Gamma_0(\mathscr{X})$ is firmly nonexpansive. In the investigated neural network (4), the activation operator is a proximity operator. In practice, this is the case for most activation operators, as shown in [28]. The neural network (4) is thus a cascade of firmly nonexpansive operators and linear operators.

B. Assumptions

We will make the assumption that the degradation operator T defined in (1) and the differential operator D are compact operators, so that we can define their singular value expansions. Furthermore, we will assume that D^*D and T^*T commute. This arises in particular when T and D correspond to filtering operations performed in a space of compactly supported signals.
Based on the above assumptions, operators D^*D and T^*T can be diagonalized in the same orthonormal set of eigenvectors $(v_p)_p$. We define their respective eigenvalues $(\beta_{T,p})_p$ and $(\beta_{D,p})_p$, as well as the following quantities, for every indices $p \in \mathbb{N}$, $n \in \{1, \dots, m\}$, and $i \in \{1, \dots, n\}$,

$$\beta_p^{(n)} = 1 - \lambda_n(\beta_{T,p} + \tau_n \beta_{D,p}) \ , \tag{16}$$

$$\beta_{i,n,p} = \prod_{j=i}^{n} \beta_p^{(j)} \ , \tag{17}$$

$$\widetilde{\beta}_{i,n,p} = \sum_{j=i}^{n-1} \beta_p^{(n)} \cdots \beta_p^{(j+1)} \lambda_j \eta_{i,j-1} + \lambda_n \eta_{i,n-1} \ , \tag{18}$$

with the convention $\sum_{i=n}^{n-1} \cdot = 0$. Note that $(\beta_p^{(n)}, v_p)_p$, $(\beta_{i,n,p}, v_p)_p$, and $(\widetilde{\beta}_{i,n,p}, v_p)_p$ are the eigensystems of W_n, $W_{i,n}$ and $\widetilde{W}_{i,n}$, defined by (6), (13), and (12), respectively.

C. Lipschitz regularity

The virtual network acts on input (x_0, b_0) and delivers output (x_m, b_m). Investigating the stability properties of such a system is possible [29] but it is not very meaningful from a practical viewpoint. A more insightful scenario consists in setting $x_0 = b_0$ and looking at the behaviour of the output x_m. This means that the system of interest is

$$R_m \circ \overline{U}_m \circ Q_{m-1} \circ U_{m-1} \cdots Q_1 \circ \widehat{U}_1, \tag{19}$$

where

$$\widehat{U}_1 = U_1 \begin{bmatrix} \mathbb{1} \\ \mathbb{1} \end{bmatrix} \tag{20}$$

$$\overline{U}_m = \mathsf{D}_{\mathrm{x}} \circ U_m \tag{21}$$

and D_{x} is the decimation operator

$$\mathsf{D}_{\mathrm{x}} = [\mathbb{1} \quad 0]. \tag{22}$$

As a preliminary result based on the assumptions in Section III-B, we quantify the spectral norms of the linear operators involved in the considered multivariate model.

Lemma 1

Let $m \in \mathbb{N} \setminus \{0\}$ be the total number of layers. For every layer indices $n \in \{1, \dots, m\}$ and $i \in \{1, \dots, n\}$,

- *the norm of $U_n \circ \cdots \circ U_i$ is equal to $\sqrt{a_{i,n}}$ with*

$$a_{i,n} = \frac{1}{2} \sup_p \Big(\beta_{i,n,p}^2 + \widetilde{\beta}_{i,n,p}^2 + \eta_{i,n}^2 + \sqrt{(\beta_{i,n,p}^2 + \widetilde{\beta}_{i,n,p}^2 + \eta_{i,n}^2)^2 - 4\beta_{i,n,p}^2 \eta_{i,n}^2} \Big) \ ; \tag{23}$$

- *the norm of $U_n \circ \cdots \circ U_2 \circ \widehat{U}_1$ is equal to $\sqrt{\widehat{a}_{1,n}}$ with*

$$\widehat{a}_{1,n} = \sup_p \Big((\beta_{1,n,p} + \widetilde{\beta}_{1,n,p})^2 \Big) + \eta_{1,n}^2 \ ; \tag{24}$$

- *the norm of $\overline{U}_m \circ U_{m-1} \circ \cdots \circ U_i$ is equal to $\sqrt{\overline{a}_{i,n}}$ with*

$$\overline{a}_{i,n} = \sup_p \Big(\beta_{i,n,p}^2 + \widetilde{\beta}_{i,n,p}^2 \Big) \ ; \tag{25}$$

- *the norm of $\overline{U}_m \circ U_{m-1} \circ \cdots \circ U_2 \circ \widehat{U}_1$ is equal to $\sqrt{\widehat{a}_{1,m}}$ with*

$$\widehat{a}_{1,m} = \sup_p (\beta_{1,m,p} + \widetilde{\beta}_{1,m,p})^2 \ . \tag{26}$$

The calculation details can be found in [29].

By applying the results in [30, Theorem 4.2], we deduce the following characterization of the stability of our unfolded network through its Lipschitz properties.

Proposition 2

Let $m \in \mathbb{N} \setminus \{0, 1\}$. For every $i \in \{2, \dots, n\}$ and $n \in \{1, \dots, m-1\}$, let $a_{i,n}$ be defined by (23) and let $\overline{a}_{i,m}$ be given by (25). For every $n \in \{1, \dots, m\}$, let $\widehat{a}_{1,n}$ be defined by (24) and (26). Define $(\widehat{\theta}_n)_{1 \le n \le m}$ recursively by

$$(\forall n \in \{1, \dots, m-1\}) \quad \widehat{\theta}_n = \sqrt{\widehat{a}_{1,n}} + \sum_{i=2}^{n} \widehat{\theta}_{i-1} \sqrt{a_{i,n}} \ , \tag{27}$$

$$\widehat{\theta}_m = \sqrt{\widehat{a}_{1,m}} + \sum_{i=2}^{m} \widehat{\theta}_{i-1} \sqrt{\overline{a}_{i,m}} \ . \tag{28}$$

Then network (19) is $\widehat{\theta}_m / 2^{m-1}$-Lipschitz.

D. Averagedness properties

As shown in [28], averagedness may also be a desirable property for neural networks since it is at the core of the convergence proofs of many iterative fixed point strategies.

Proposition 3

Let $m \in \mathbb{N} \setminus \{0, 1\}$. Let $\widehat{a}_{1,m}$ be defined in Lemma 1 and let $\widehat{\theta}_m$ be defined in Proposition 2. Let $\alpha \in [1/2, 1]$. Define

$$\widehat{b}_\alpha = \sup_p |\beta_{1,m,p} + \widetilde{\beta}_{1,m,p} - 2^m(1-\alpha)|. \tag{29}$$

If

$$\widehat{b}_\alpha - \sqrt{\widehat{a}_{1,m}} \le 2^m \alpha - 2\widehat{\theta}_m \ , \tag{30}$$

then network (19) is α-averaged.

Note that this proposition only provides a sufficient condition for the unfolded network to be α-averaged.

IV. Numerical experiments

A. Inverse problem

In this section, we present numerical tests carried out on the class of Abel integral operators. The Abel integral operator

Noise δ	Lowess		Neural Network		Fourier	
	$a=1$	$a=1/2$	$a=1$	$a=1/2$	$a=1$	$a=1/2$
0.1	3.96%	2.52%	2.70%	0.51%	3.72%	0.75%
0.05	2.66%	2.54%	1.55%	0.18%	2.25%	0.23%
0.01	2.49%	2.68%	0.31%	0.03%	0.43%	0.01%

TABLE I

AVERAGED RELATIVE ERROR OOBTAINED FOR DIFFERENT NOISE STANDARD DEVIATION VALUES δ AND DIFFERENT TYPES OF SIGNALS.

operates from $\mathscr{X} = L^2(0,1)$ to $\mathscr{Y} = L^2(0,1)$ and associates to a signal $x \in \mathscr{X}$ a signal v such that

$$(\forall t \in [0,1]) \quad v(t) = \frac{1}{\Gamma(a)} \int_0^t (t-s)^{(a-1)} x(s)\,\mathrm{d}s\,, \quad (31)$$

where $a > 0$ and Γ is the classical Gamma function. The Abel operator T is injective, linear, and compact.

Recovering x from a noisy measurement $y = v + w$ is an inverse problem linked to a large variety of experimental contexts in physics. Indeed, the operator T allows to define derivatives of fractional order for $a < 1$ and integrals of arbitrary order for $a > 1$. The most common case is the semi-derivative, when $a = 1/2$. A large number of physical applications have been documented in [31].

B. Implementation

The differential operator D is here chosen equal to a power B^r of the Laplacian denoted by B ($r = 1/2$). This choice ensures that the continuous operators T^*T and D^*D commute. To generate synthetic data, the Abel integral has been approximated by the trapezoidal rule on a fine regular mesh of $N = 2000$ points $(t_i)_{0\le i\le N-1} \subset [0,1[$. The operators T^*T and D^*D which are used in our neural network have been approximated by projection onto the span of their $K = 50$ first eigenvectors.

To create a diverse dataset of positively distributed functions, we found convenient to use histograms of images from a standard image dataset (BSDS500). We consider that 400 examples are enough to train the networks and 200 are used to test it. In order to properly reflect the prior regularity, the signals are first smoothed using a Savitzky-Golay filter, with filter length 21 and polynomial order 5. Then, to ensure that such signals are in the range of T^*T, the signals are projected into the eigenvector basis described above. After applying the discretized Abel operator, a zero-mean white Gaussian noise with a preset standard deviation δ is added.

C. Neural network

The activation function, namely operator R_n, corresponds the proximal operator associated with g appearing in (2). To reflect prior information, such a function can be chosen equal to the indicator function of a nonempty closed convex set. In this case, the proximity operator reduces to a projection onto this set. However, such activation functions may show vanishing gradient problems during the training procedure. To alleviate this issue, we chose to consider instead a logarithmic barrier g to enable prior knowledge in the algorithm, as proposed in [21]. We have then

$$\begin{cases} C = \{x \in L^2(0,1) \mid c_i(x) \ge 0,\ 1 \le i \le p\}\,, \\ (\forall x \in L^2(0,1))\ \ g(x) = \begin{cases} -\sum_{i=1}^p \ln\,(c_i(x)) & \text{if } x \in \operatorname{int} C \\ +\infty & \text{otherwise}\,, \end{cases} \end{cases} \quad (32)$$

where $(c_i)_{1\le i\le p}$ are suitable functions allowing us to describe the constraint set. The computation of the proximity operator associated to logarithmic barrier functions [32], after discretization, can be found in [21].

We experimented two possible choices for set C. First, we consider that the signal x has a minimum value $x_{\min}$ and a maximum value $x_{\max}$. Then C can be rewritten as

$$C = \left\{x \in L^2(0,1) \mid x \ge x_{\min},\ -x \ge -x_{\max}\right\}\ . \quad (33)$$

Secondly, we consider an affine constraint such as, for $j \ge 0$,

$$C = \left\{x \in L^2 \mid\ 0 \le \int_0^1 t^j x(t)\,\mathrm{d}t \le 1\right\}\ . \quad (34)$$

This constraint reflects the fact that a physical quantity, e.g. the total mass, linked to the signal is bounded.

For each layer n, three parameters (stepsize λ_n, quadratic regularization parameter τ_n, and barrier parameter μ_n) are then learned by proceeding similarly to [21]. The leakage factors are set to 1, in these experiments.

D. Results

We compare the results obtained with our unfolded structure with those provided by a Fourier technique [33] and temporal filtering techniques, e.g. Lowess filtering [34]. If $\widehat{x}$ denotes the restored signal, we report in Table I the averaged value of the relative error $\|\widehat{x} - x\|/\|x\|$. In our approach, we used $m = 20$ layers and Constraint (34) with $j = 1$, which turned out to provide slightly better results. We observe a lower error when a is smaller, which is related to the fact that the higher the order of integration a in T, the higher the order of differentiation performed in the inversion, hence the stronger the impact of the noise. By using Proposition 2, we have also been able to evaluate the Lipschitz constant of the network which is close to 4.93×10^{-2} when $x_0 = b_0 = T^*y$.

V. CONCLUSION

This paper unfolded an algorithm derived from a variational formulation of inverse problems. We focused on the forward-backward algorithm which offers a versatile solution for a variety of regularized quadratic fidelity terms. The advantage of the resulting neural network is that it has a limited number of layers and a small number of parameters. Its training can thus be performed in a few minutes while its inference time is extremely fast.

We additionally performed a theoretical analysis of robustness with respect to the observed data, which ensures the reliability of the proposed inverse method.

Finally, we showed the applicability of this approach for solving continuous one-dimensional inverse problems involving an Abel integral operator.

In future work, more sophisticated neural network structures could be considered or additional parameters (such as the leakage factors we introduced in Section II-B) could be learned. Also, training sets which would better suited to specific applications could be employed.

References

[1] M. Bertero, P. Boccacci, G. Desiderà, and G. Vicidomini, "Image deblurring with Poisson data: from cells to galaxies," *Inverse Problems*, vol. 25, no. 12, p. 123006, Nov. 2009. [Online]. Available: https://doi.org/10.1088/0266-5611/25/12/123006

[2] Z. Wu, Y. Sun, A. Matlock, J. Liu, L. Tian, and U. S. Kamilov, "SIMBA: Scalable inversion in optical tomography using deep denoising priors," *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 6, pp. 1163–1175, Oct. 2020. [Online]. Available: https://doi.org/10.1109/jstsp.2020.2999820

[3] A. Cherni, E. Chouzenoux, and M.-A. Delsuc, "PALMA, an improved algorithm for DOSY signal processing," *The Analyst*, vol. 142, no. 5, pp. 772–779, 2017. [Online]. Available: https://doi.org/10.1039/c6an01902a

[4] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*. Springer, 2011, pp. 185–212.

[5] S. Antholzer, M. Haltmeier, and J. Schwab, "Deep learning for photoacoustic tomography from sparse data," *Inverse Problems in Science and Engineering*, vol. 27, no. 7, pp. 987–1005, 2019.

[6] A. Kofler, M. Haltmeier, C. Kolbitsch, M. Kachelrieß, and M. Dewey, "A u-nets cascade for sparse view computed tomography," in *Proceedings of the International Workshop on Machine Learning for Medical Image Reconstruction (MLMIR 2018)*. Springer, 2018, pp. 91–99.

[7] K. Hammernik, T. Klatzer, E. Kobler, M. P. Recht, D. K. Sodickson, T. Pock, and F. Knoll, "Learning a variational network for reconstruction of accelerated mri data," *Magnetic Resonance in Medicine*, vol. 79, no. 6, pp. 3055–3071, 2018.

[8] J. Adler and O. Öktem, "Solving ill-posed inverse problems using iterative deep neural networks," *Inverse Problems*, vol. 33, no. 12, p. 124007, 2017.

[9] H. K. Aggarwal, M. P. Mani, and M. Jacob, "Modl: Model-based deep learning architecture for inverse problems," *IEEE Transactions on Medical Imaging*, vol. 38, no. 2, pp. 394–405, 2018.

[10] 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 (ICCV 2017)*, 2017, pp. 1781–1790.

[11] J.-C. Pesquet, A. Repetti, M. Terris, and Y. Wiaux, "Learning maximally monotone operators for image recovery," *SIAM Journal on Imaging Sciences*, vol. 14, no. 3, pp. 1206—-1237, 2021.

[12] M. Hasannasab, J. Hertrich, S. Neumayer, G. Plonka, S. Setzer, and G. Steidl, "Parseval proximal neural networks," *Journal of Fourier Analysis and Applications*, vol. 26, no. 4, Jul 2020. [Online]. Available: http://dx.doi.org/10.1007/s00041-020-09761-7

[13] M. Galinier, M. Prato, E. Chouzenoux, and J.-C. Pesquet, "A hybrid interior point - deep learning approach for Poisson image deblurring," in *Proceedings of the 30th International Workshop on Machine Learning for Signal Processing (MLSP 2020)*. IEEE, Sep. 2020, pp. 1–6. [Online]. Available: https://doi.org/10.1109/mlsp49062.2020.9231876

[14] H. Li, J. Schwab, S. Antholzer, and M. Haltmeier, "NETT: Solving inverse problems with deep neural networks," *Inverse Problems*, vol. 36, no. 6, p. 065005, 2020.

[15] T. A. Bubba, M. Galinier, M. Lassas, M. Prato, L. Ratti, and S. Siltanen, "Deep neural networks for inverse problems with pseudodifferential operators: an application to limited-angle tomography," *SIAM Journal on Imaging Sciences*, vol. 14, no. 2, pp. 470–505, 2021.

[16] D. Gilton, G. Ongie, and R. Willett, "Neumann networks for linear inverse problems in imaging," *IEEE Transactions on Computational Imaging*, vol. 6, pp. 328–343, 2019.

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

[18] Y. Sun, B. Wohlberg, and U. S. Kamilov, "An online plug-and-play algorithm for regularized image reconstruction," *IEEE Transactions on Computational Imaging*, vol. 5, no. 3, pp. 395–408, 2019.

[19] M. Borgerding and P. Schniter, "Onsager-corrected deep learning for sparse linear inverse problems," in *Proceedings of the IEEE Global Conference on Signal and Information Processing (GlobalSIP 2016)*. IEEE, 2016, pp. 227–231.

[20] K. H. Jin, M. T. McCann, E. Froustey, and M. Unser, "Deep convolutional neural network for inverse problems in imaging," *IEEE Transactions on Image Processing*, vol. 26, no. 9, pp. 4509–4522, 2017.

[21] 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*, vol. 36, no. 3, p. 034005, 2020.

[22] S. K. Natarajan and M. A. Caro, "Particle swarm based hyper-parameter optimization for machine learned interatomic potentials," *arXiv preprint arXiv:2101.00049*, 2020.

[23] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: LSTM cells and network architectures," *Neural Computation*, vol. 31, no. 7, pp. 1235–1270, 2019.

[24] W. Tang, E. Chouzenoux, J.-C. Pesquet, and H. Krim, "Deep transform and metric learning network: Wedding deep dictionary learning and neural network." *Neurocomputing*, vol. 509, pp. 244–256, Oct. 2022.

[25] H. H. Bauschke and P. L. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. New York: Springer, 2011, vol. 408.

[26] P. L. Combettes and V. R. Wajs, "Signal recovery by proximal forward-backward splitting," *Multiscale Modeling & Simulation*, vol. 4, no. 4, pp. 1168–1200, 2005.

[27] P. L. Combettes and J.-C. Pesquet, "Fixed point strategies in data science," *IEEE Transactions on Signal Processing*, vol. 59, pp. 3878–3905, 2021.

[28] ——, "Deep neural network structures solving variational inequalities," *Set-Valued and Variational Analysis*, pp. 1–28, 2020.

[29] E. Chouzenoux, C. Della Valle, and J.-C. Pesquet, "Inversion of integral models: a neural network approach," *arXiv preprint arXiv:2105.15044*, 2021.

[30] P. L. Combettes and J.-C. Pesquet, "Lipschitz certificates for layered network structures driven by averaged activation operators," *SIAM Journal on Mathematics of Data Science*, vol. 2, no. 2, pp. 529–557, 2020.

[31] R. Gorenflo and F. Mainardi, "Fractional calculus," in *Fractals and Fractional Calculus in Continuum Mechanics*. Springer, 1997, pp. 223–276.

[32] E. Chouzenoux, S. Moussaoui, and J. Idier, "Majorize–minimize linesearch for inversion methods involving barrier function optimization," *Inverse Problems*, vol. 28, no. 6, p. 065011, May 2012. [Online]. Available: https://doi.org/10.1088/0266-5611/28/6/065011

[33] S. Ma, H. Gao, and L. Wu, "Modified Fourier-Hankel method based on analysis of errors in Abel inversion using Fourier transform techniques," *Applied Optics*, vol. 47, no. 9, pp. 1350–1357, 2008.

[34] E. W. Hansen and P.-L. Law, "Recursive methods for computing the Abel transform and its inverse," *Journal of the Optical Society of America A*, vol. 2, no. 4, pp. 510–520, 1985.