Some Intriguing Observations on the Learnt Matrices of the Deep Unfolded Networks – Supplementary Material

Kartheek Kumar Reddy Nareddy †, Inbasekaran Perumal ‡, Chandra Sekhar Seelamantula †

† Department of Electrical Engineering, Indian Institute of Science, Bengaluru-560012

‡ Department of Electronics and Communication Engineering, National Institute of Technology Karnataka, Mangaluru-575025

Email:nareddyreddy@iisc.ac.in, inba2002.p@gmail.com, css@iisc.ac.in

I. Extended Background

Consider the quadratic majorizer $q(\cdot;\boldsymbol{x}^{(k)})$ of the data-fidelity loss at a point $\boldsymbol{x}^{(k)} \in \mathbb{R}^n$ given by

$$q\left(\boldsymbol{x};\boldsymbol{x}^{(k)}\right) = f\left(\boldsymbol{x}^{(k)}\right) + \left(\boldsymbol{x}-\boldsymbol{x}^{(k)}\right)^{\mathrm{T}} \nabla f\left(\boldsymbol{x}^{(k)}\right) + \frac{1}{2\eta}\|\boldsymbol{x}-\boldsymbol{x}^{(k)}\|_2^2, \tag{1}$$

where $\eta \leq \dfrac{1}{L}$, ensuring $q\left(\boldsymbol{x};\boldsymbol{x}^{(k)}\right) \geq f(\boldsymbol{x}),\ \forall \boldsymbol{x} \in \mathbb{R}^n$, and $q\left(\boldsymbol{x}^{(k)};\boldsymbol{x}^{(k)}\right) = f\left(\boldsymbol{x}^{(k)}\right)$. The majorizer can be rewritten by completing the squares as

$$q\left(\boldsymbol{x};\boldsymbol{x}^{(k)}\right) = \frac{1}{2}\left\|\boldsymbol{x} - \left(\boldsymbol{x}^{(k)} - \eta\mathbf{A}^{\mathrm{T}}(\mathbf{A}\boldsymbol{x}^{(k)}-\boldsymbol{y})\right)\right\|_2^2 + \text{constant}. \tag{2}$$

A. Updates of ISTA

Let $\boldsymbol{x}^{(k+1)}$ be the minimizer of the majorizer $q\left(\boldsymbol{x};\boldsymbol{x}^{(k)}\right) + \lambda\|\boldsymbol{x}\|_1 \geq F(\boldsymbol{x})$. For the unconstrained problem:

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}}\ \frac{1}{2}\|\mathbf{A}\boldsymbol{x}-\boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{x}\|_1, \tag{3}$$

the proximal gradient method update is:

$$\boldsymbol{x}^{(k+1)} = \mathcal{T}_{\eta\lambda}\left(\boldsymbol{x}^{(k)} - \eta\mathbf{A}^{\mathrm{T}}(\mathbf{A}\boldsymbol{x}^{(k)}-\boldsymbol{y})\right), \tag{4}$$

where $\mathcal{T}_{\eta\lambda}$ is the shrinkage function:

$$\mathcal{T}_{\eta\lambda}(\boldsymbol{x}) = \operatorname{sgn}(\boldsymbol{x}) \cdot \max(|\boldsymbol{x}| - \eta\lambda\mathbf{1}, \mathbf{0}). \tag{5}$$

B. Derivation of LISTA

The updates in (4) can be rewritten as:

$$\begin{aligned} \boldsymbol{x}^{(k+1)} &= \mathcal{T}_{\eta\lambda}\left(\boldsymbol{x}^{(k)} - \eta\mathbf{A}^{\mathrm{T}}\mathbf{A}\boldsymbol{x}^{(k)} + \eta\mathbf{A}^{\mathrm{T}}\boldsymbol{y}\right), \\ \boldsymbol{x}^{(k+1)} &= \mathcal{T}_{\eta\lambda}\left(\left(\mathbf{I} - \eta\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)\boldsymbol{x}^{(k)} + \eta\mathbf{A}^{\mathrm{T}}\boldsymbol{y}\right). \end{aligned} \tag{6}$$

The final equation can be interpreted as

$$\boldsymbol{x}^{(k+1)} = \mathcal{T}_{\eta^{(k)}\lambda^{(k)}}\left(\mathbf{S}\boldsymbol{x}^{(k)} + \mathbf{W}\boldsymbol{y}\right)$$

where $\mathbf{S} = \mathbf{I} - \eta\mathbf{A}^{\mathrm{T}}\mathbf{A}$ and $\mathbf{W} = \eta\mathbf{A}^{\mathrm{T}}$. This is the foundation for learned ISTA (LISTA). The iterations can be viewed as linear transformations in a fully connected layer, with a shrinkage-thresholding function acting as a non-linearity. It is also notable that the shrinkage-thresholding function can be expressed as a combination of two ReLU activation functions:

$$\mathcal{T}_{\eta\lambda}(\boldsymbol{x}) = \mathrm{ReLU}(\boldsymbol{x} - \eta\lambda\mathbf{1}) - \mathrm{ReLU}(-(\boldsymbol{x} + \eta\lambda\mathbf{1})). \tag{7}$$

LISTA is a recurrent neural network with learnable network weights $\mathbf{S}$, $\mathbf{W}$, and the nonlinearity $\mathcal{T}_{\eta^{(k)}\lambda^{(k)}}$. The network weights are typically shared between the layers. The network has a finite number of layers (L), and the output from the last layer ($\boldsymbol{x}^{(L)}$) is considered as the reconstructed signal. The network is trained on squared error (quadratic loss) $\|\boldsymbol{x}^{(L)} - \boldsymbol{x}^*\|_2^2$ [1]. Observations from [1] have shown that the number of iterations required for accurately reconstructing the sparse signal $\boldsymbol{x}^*$ has reduced from several hundred iterations in the case of ISTA to about a dozen iterations/layers in the case of LISTA. Moreover, the reconstruction accuracy has improved with LISTA, although it requires more extensive training [1].

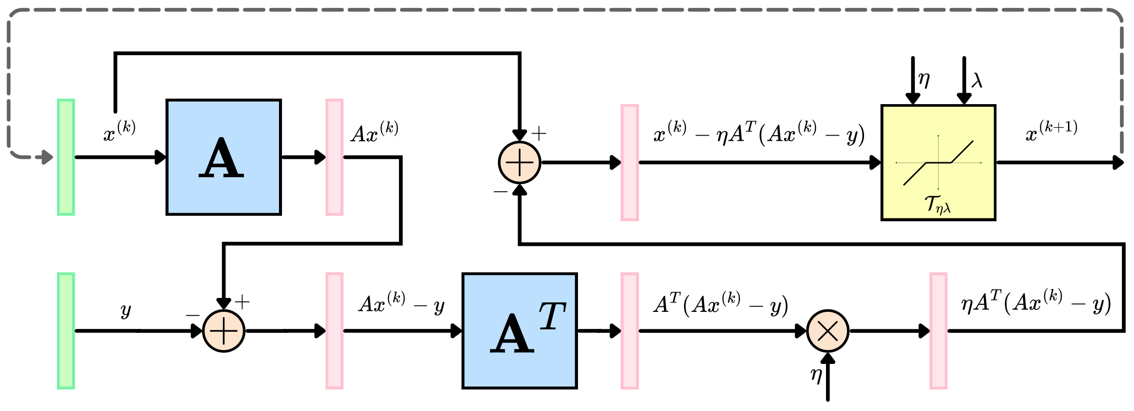


Fig. 1: This figure illustrates the computational workflow of ISTA. Here, λ and η are hyperparameters for thresholding and step-size, respectively.

C. Back-Projection Loss or Tight-Frame Loss Formulation

The back-projection (BP) loss [2]–[4] or the tight-frame (TF) data-fidelity loss [5] are proposed in conjunction with various regularization terms for solving image reconstruction and image restoration tasks [6]. In this work, we

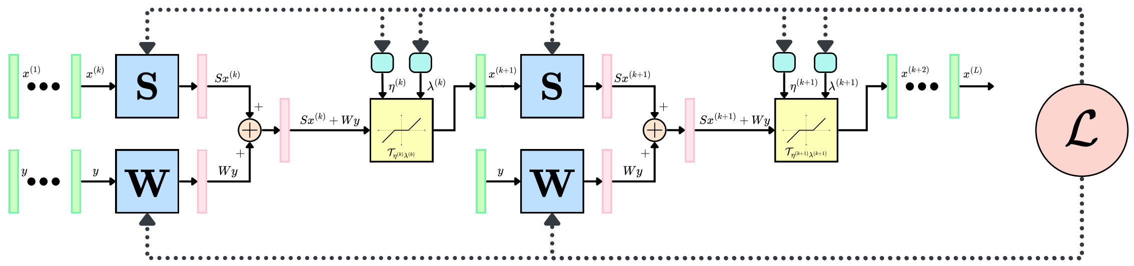
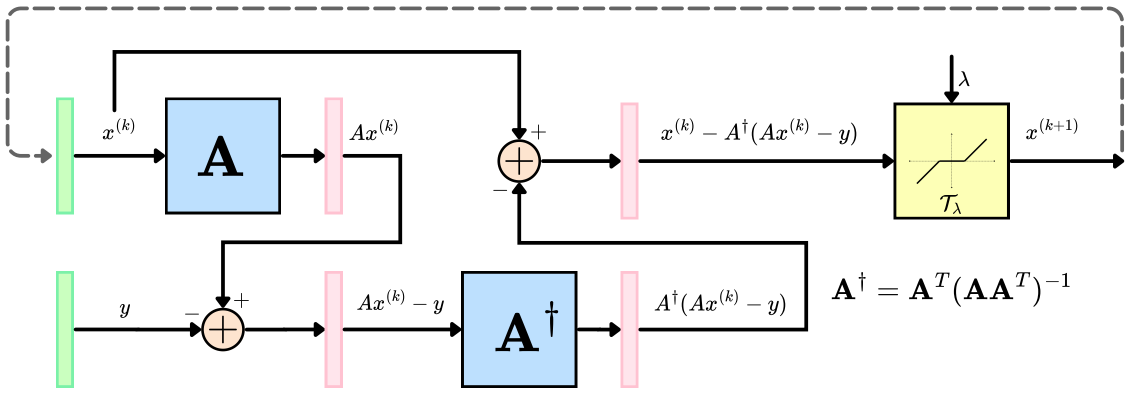


Fig. 2: This schematic shows the architecture of LISTA, which is the deep-unfolded version of ISTA. The matrices $\mathbf{S}$ and $\mathbf{W}$ are common across all layers similar to weight sharing in recurrent neural networks (RNNs), while $\eta^{(k)}$ and $\lambda^{(k)}$ are optimized individually for each layer through backpropagation. The process iteratively refines the estimate $x^{(k+1)}$ until the final output $x^{(L)}$ is obtained by minimizing the loss function $\mathcal{L}$.



$x^{(k)}$
A
$Ax^{(k)}$
$x^{(k)} - A^{\dagger}(Ax^{(k)} - y)$
λ
$x^{(k+1)}$
$\mathcal{T}_{\lambda}$
y
$Ax^{(k)} - y$
$\mathbf{A}^{\dagger}$
$A^{\dagger}(Ax^{(k)} - y)$
$\mathbf{A}^{\dagger} = \mathbf{A}^{T}(\mathbf{A}\mathbf{A}^{T})^{-1}$

Fig. 3: This figure illustrates the computational workflow of TF-ISTA. The process updates the initial guess $x^{(k)}$ using matrix A and its pseudoinverse $A^{\dagger}$, followed by thresholding controlled by λ, leading to the next iteration $x^{(k+1)}$.

refer to the data fidelity as TF loss, which is defined by:

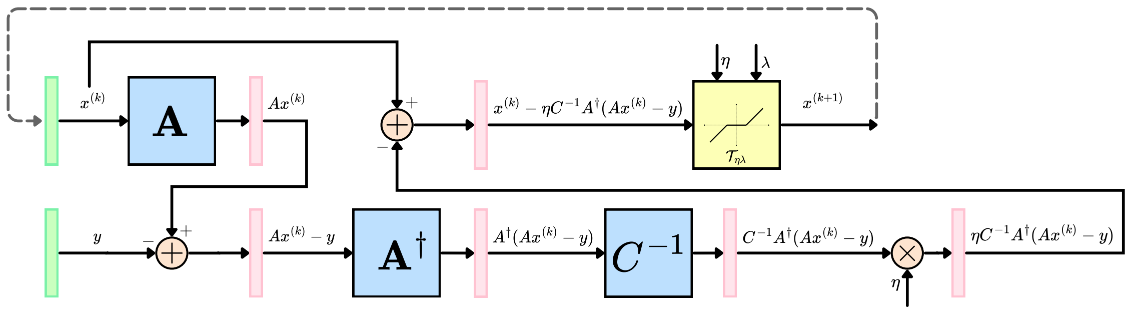
$$f(\boldsymbol{x}) = \frac{1}{2}\left\|\left(\mathbf{A}\mathbf{A}^{\mathrm{T}}\right)^{-\frac{1}{2}}(\mathbf{A}\boldsymbol{x} - \boldsymbol{y})\right\|_2^2. \tag{8}$$

It is assumed that $\mathbf{A}$ has full row rank. When combined with the ℓ_1 penalty, this leads to a new optimization formulation:

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}}\ \frac{1}{2}\left\|\left(\mathbf{A}\mathbf{A}^{\mathrm{T}}\right)^{-\frac{1}{2}}(\mathbf{A}\boldsymbol{x} - \boldsymbol{y})\right\|_2^2 + \lambda\|\boldsymbol{x}\|_1, \tag{9}$$

where $\lambda > 0$ is the regularization parameter. The *effective sensing matrix* in (9) becomes $\mathbf{A}_e = \left(\mathbf{A}\mathbf{A}^{\mathrm{T}}\right)^{-\frac{1}{2}}\mathbf{A}$, which satisfies the property: $\mathbf{A}_e\mathbf{A}_e^{\mathrm{T}} = \mathbf{I}$, i.e., it is a Parseval tight frame [7], [8]. The data-fidelity loss $f(\cdot)$ is convex as it is the composition of an affine transform and a norm, and also differentiable. The $\|\cdot\|_1$ penalty, on the other hand, is convex, but not differentiable. The optimization can be performed using the proximal gradient method (PGM) [9]. PGM requires a convex majorizer, which is possible in this case as the data-fidelity loss has a Lipschitz continuous gradient [3].

Using ℓ_1-norm, ℓ_2-norm, and DCGAN [10] priors, Tirer and Giryes [3], [4] examined the performance of least-squares and back-projection costs across different compression ratios. They demonstrated that, at all compression ratios, back-projection (BP) routinely outperforms the least-squares (LS) goal. Theoretical results from [4] show that



$x^{(k)}$
$\mathbf{A}$
$Ax^{(k)}$
$+$
$-$
$x^{(k)} - \eta C^{-1}A^{\dagger}(Ax^{(k)} - y)$
η
λ
$\mathcal{T}_{\eta\lambda}$
$x^{(k+1)}$
y
$-$
$+$
$Ax^{(k)} - y$
$\mathbf{A}^{\dagger}$
$A^{\dagger}(Ax^{(k)} - y)$
C^{-1}
$C^{-1}A^{\dagger}(Ax^{(k)} - y)$
η
$\eta C^{-1}A^{\dagger}(Ax^{(k)} - y)$

Fig. 4: This figure illustrates the computational workflow of RTF-ISTA. The process enhances the update step of $x^{(k)}$ by incorporating a regularization matrix C^{-1} in addition to the components of TF-ISTA.

even in cases of poor conditioning of $\mathbf{A}\mathbf{A}^{\mathrm{T}}$, BP cost is still more robust than LS. Particularly, BP iterates converge more successfully than LS if $\mathbf{A}\mathbf{A}^{\mathrm{T}}$ has a large condition number, assuming a convex prior as per Condition 4.1 in [4].

Using BM3D [11] and CNN [12] priors, the effect of noise on image restoration performance using the BP objective was studied in [2]. Tirer and Giryes proved in a theoretical analysis [3] that, given at least one singular value of $\mathbf{A}$ is less than one in a noiseless context, BP achieves a lower MSE than LS with an ℓ_2-norm prior. Empirical results from [3] verify that the recovery accuracy with BP is superior to that with LS even in noisy situations. At low noise levels, BP outperforms LS in image restoration tasks involving total-variation [13], DCGAN, and BM3D priors more noticeably. BP was shown to converge faster than LS for CS recovery using the ℓ_1-norm prior at all noise levels, with a larger difference at lower noise levels [4].

D. *Updates for TF-ISTA*

The update step for solving (9) using the proximal gradient method is as follows:

$$\boldsymbol{x}^{(k+1)} = \mathcal{T}_{\eta\lambda}\left(\boldsymbol{x}^{(k)} - \eta\mathbf{A}^{\mathrm{T}}(\mathbf{A}\mathbf{A}^{\mathrm{T}})^{-1}(\mathbf{A}\boldsymbol{x}^{(k)} - \boldsymbol{y})\right), \tag{10}$$

Given that the Lipschitz constant of $\mathbf{A}^{\mathrm{T}}(\mathbf{A}\mathbf{A}^{\mathrm{T}})^{-1}(\mathbf{A}\boldsymbol{x}^{(k)} - \boldsymbol{y})$ equals 1 [3], [5], the step-size parameter η can be set to 1, yielding:

$$\boldsymbol{x}^{(k+1)} = \mathcal{T}_{\lambda}\left(\boldsymbol{x}^{(k)} - \mathbf{A}^{\mathrm{T}}(\mathbf{A}\mathbf{A}^{\mathrm{T}})^{-1}(\mathbf{A}\boldsymbol{x}^{(k)} - \boldsymbol{y})\right). \tag{11}$$

The schematic of TF-ISTA is given in Figure 3.

E. *Derivation of RTF-ISTA*

Consider modifying the gradient of the data-fidelity loss using a diagonal matrix $\mathbf{C} \in \mathbb{R}^{n\times n}$ to update:

$$\boldsymbol{x}^{(k+1)} = \mathcal{T}_{\eta\lambda}\left(\boldsymbol{x}^{(k)} - \eta\mathbf{C}^{-1}\mathbf{A}^{\mathrm{T}}(\mathbf{A}\mathbf{A}^{\mathrm{T}})^{-1}(\mathbf{A}\boldsymbol{x}^{(k)} - \boldsymbol{y})\right). \tag{12}$$

Consider the specific case where $\mathbf{C} = \mathrm{diag}\left(\mathbf{A}^{\mathrm{T}}\left(\mathbf{A}\mathbf{A}^{\mathrm{T}}\right)^{-1}\mathbf{A}\right)$, where $\mathrm{diag}(\mathbf{A})$ equals $\mathbf{I} \odot \mathbf{A}$ and $\odot$ denotes the Hadamard product. This adjustment ensures that the corresponding sensing matrix has unit ℓ_2-norm columns. This method is termed *RTF-ISTA* in [5] and its schematic is given in Figure 4.

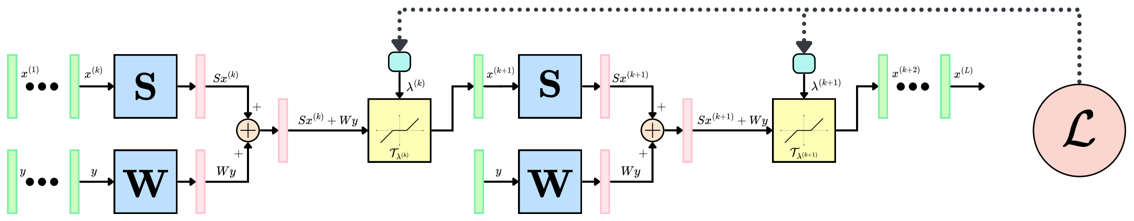


Fig. 5: This diagram showcases the TF-LISTA architecture, where the matrices S and W are fixed, contrasting with LISTA's learnable matrices. The only learnable components in TF-LISTA are the shrinkage-thresholding parameters $\{\lambda^{(k)}\}_{k=1}^{L}$.

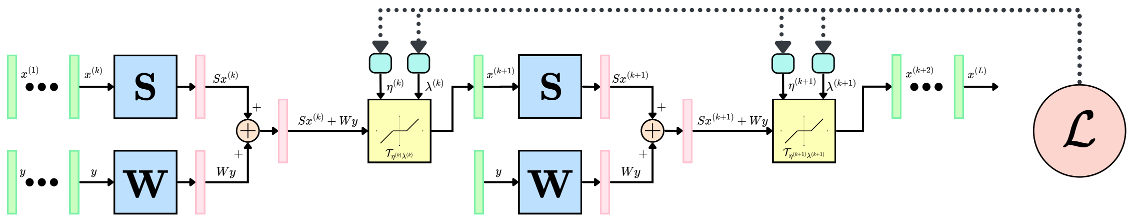


Fig. 6: This diagram depicts the ALISTA architecture, which builds upon the TF-LISTA framework by integrating additional learnable step-size parameters $\{\eta^{(k)}\}_{k=1}^{L}$ along with the shrinkage-thresholding parameters $\{\lambda^{(k)}\}_{k=1}^{L}$.

F. Derivation of TF-LISTA

The process of deriving TF-LISTA from TF-ISTA is somewhat similar to the transition from ISTA to LISTA. The key difference lies in the nature of the linear transformations; in TF-LISTA, the matrices $\mathbf{S}$ and $\mathbf{W}$ are fixed, unlike in LISTA where these matrices are data-driven and learnable. The schematic of TF-LISTA is given in Figure 5. In TF-LISTA, the shrinkage-thresholding parameters $\{\lambda^{(k)}\}_{k=1}^{L}$ are the only learnable components. The iterative or layer-wise sparse-signal reconstruction updates for TF-LISTA are given by:

$$\boldsymbol{x}^{(k+1)} = \mathcal{T}_{\lambda^{(k)}}\left(\boldsymbol{x}^{(k)} - \mathbf{A}^{\mathrm{T}}(\mathbf{A}\mathbf{A}^{\mathrm{T}})^{-1}(\mathbf{A}\boldsymbol{x}^{(k)} - \boldsymbol{y})\right), \tag{13}$$

where $\{\lambda^{(k)}\}_{k=1}^{L}$ are the only learnable parameters in the TF-LISTA network.

G. Differences between LISTA and TF-LISTA

The primary distinction between LISTA and TF-LISTA lies in the configuration of the network weights $\mathbf{S}$ and $\mathbf{W}$. In LISTA, these weights are learnable and thus require substantial data for effective training. Conversely, TF-LISTA utilizes fixed linear transformations, and only the shrinkage-threshold parameters $\{\lambda^{(k)}\}_{k=1}^{L}$ are learnable. As a result, while LISTA involves training several thousand parameters, TF-LISTA significantly reduces the number of trainable parameters.

H. Derivation of ALISTA

Liu *et al.* [14] demonstrated that the optimal weight matrices in a LISTA network can be obtained by solving a data-free optimization problem, significantly reducing the number of parameters to be learned during training.The schematic of ALISTA is given in Figure 6. The transformation for the kth unfolding of ALISTA is expressed as:

$$\boldsymbol{x}^{(k+1)} = \mathcal{T}_{\eta^{(k)}\lambda^{(k)}}\left(\boldsymbol{x}^{(k)} - \eta^{(k)}\mathbf{W}_*^{\mathrm{T}}\left(\mathbf{A}\boldsymbol{x}^{(k)} - \boldsymbol{y}\right)\right), \tag{14}$$

where $\mathbf{W}_*$ is determined by solving the following optimization problem:

$$\begin{aligned} \mathbf{W}_* = \arg\min_{\mathbf{W}} \|\mathbf{W}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2, \\ \text{subject to } \boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{a}_i = 1,\ i \in [\![1, n]\!], \\ \text{where } \mathbf{W} = [\boldsymbol{w}_1 \cdots \boldsymbol{w}_n] \in \mathbb{R}^{m\times n} \end{aligned} \tag{15}$$

This optimization aims to find the matrix $\mathbf{W}_*$ with the least coherence with $\mathbf{A}$ [15]. Liu *et al.* proposed an iterative method to solve Equation (15). However, they also found that an iterative algorithm is not necessary, as $\mathbf{W}_*$ can be determined in closed form:

$$\mathbf{W}_* = \left(\mathbf{A}\mathbf{A}^{\mathrm{T}}\right)^{-1}\mathbf{A}\,\mathbf{C}^{-1}, \tag{16}$$

where $\mathbf{C} = \left(\mathbf{A}^{\mathrm{T}}\left(\mathbf{A}\mathbf{A}^{\mathrm{T}}\right)^{-1}\mathbf{A}\right) \odot \mathbf{I}$, serving to normalize the columns of $\mathbf{W}_*^{\mathrm{T}}\mathbf{A}$. The sparse signal reconstruction updates in ALISTA are thus:

$$\boldsymbol{x}^{(k+1)} = \mathcal{T}_{\eta^{(k)}\lambda^{(k)}}\left(\boldsymbol{x}^{(k)} - \eta^{(k)}\mathbf{C}^{-1}\mathbf{A}^{\mathrm{T}}\left(\mathbf{A}\mathbf{A}^{\mathrm{T}}\right)^{-1}\left(\mathbf{A}\boldsymbol{x}^{(k)} - \boldsymbol{y}\right)\right). \tag{17}$$

This update process in ALISTA closely mirrors the updates in TF-LISTA (13), with additional step-size parameters and a normalizing matrix $\mathbf{C}^{-1}$ differentiating ALISTA. The RTF-ISTA updates from (12) match the layer updates in ALISTA, thus positioning ALISTA as a deep-unfolded variant of RTF-ISTA.

II. Additional Experimental Results

A. Reverse Diagonal Effect in LISTA

The $\mathbf{S}$ matrix in LISTA shows an interesting pattern that differs from other methods. While ISTA and TF-LISTA exhibit diagonal dominance, LISTA reverses this trend. This section explores this unique characteristic.

Figure 7 shows the original and sign-inverted $\mathbf{S}$ matrices in LISTA. The sign inversion reveals a pattern similar to ISTA and TF-LISTA, with diagonal dominance. This suggests that LISTA's unique structure might offer similar benefits in terms of regularization and numerical stability, despite its apparent difference.

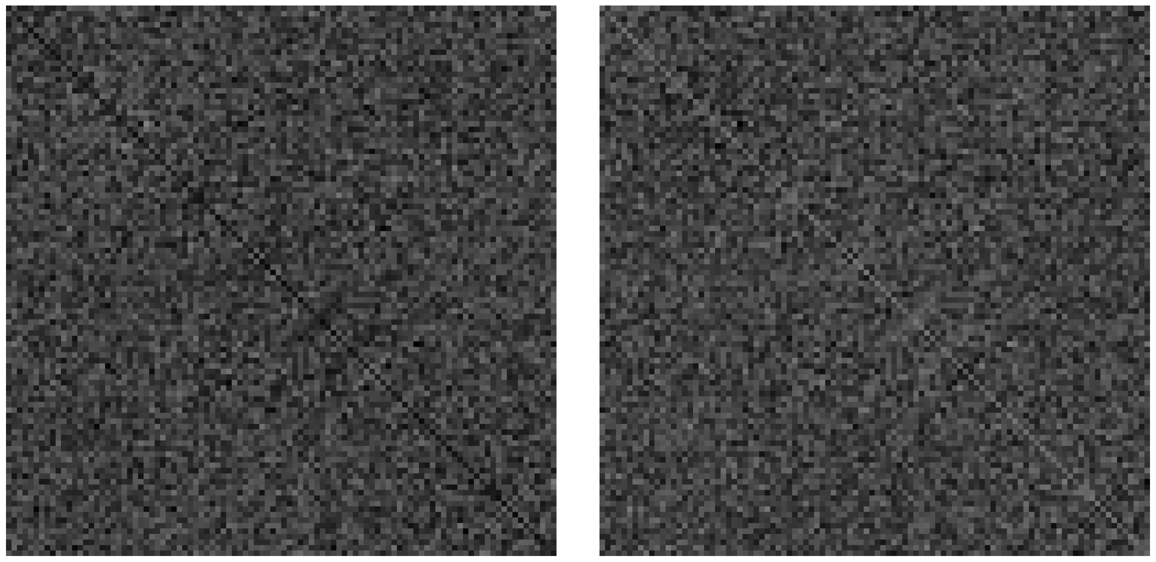
Diagonal dominance in matrices often leads to better conditioning, enhancing regularization and numerical stability. While LISTA's original $\mathbf{S}$ matrix doesn't show this property directly, the emergence of a similar pattern after sign inversion indicates that LISTA might achieve these benefits through a different mechanism.

B. Learned Threshold Values in LISTA

Figure 8 illustrates the learned threshold values in LISTA for networks with 15, 100, and 500 layers. As the network depth increases, we observe a decreasing trend in threshold values, indicating a saturation effect in deeper architectures. This behavior suggests that LISTA adapts its thresholding strategy based on network depth, potentially optimizing the trade-off between signal amplification and stability.

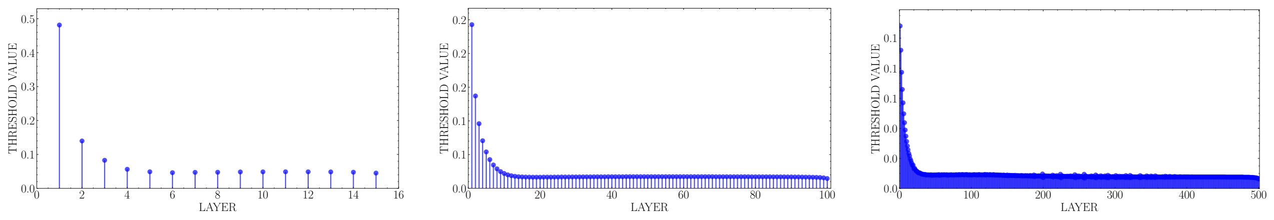
C. Impact of the Soft Threshold Function on Stability

These figures show how the soft threshold function affects LISTA's stability. Figure 9 shows that with the activation, the squared ℓ^2 norm $\|\hat{x}\|_2^2$ stays nearly constant across different layer counts. This indicates stability.



(a) (b)

Fig. 7: Comparison of learned $\mathbf{S}$ matrices from LISTA: (a) original and (b) sign-inverted. The inversion reveals a diagonal dominance trend similar to ISTA and TF-LISTA, suggesting improved regularization and stability.



15 Layers 100 Layers 500 Layers

Fig. 8: Threshold values in LISTA across different layer counts. The threshold decreases as the number of layers increases, showing saturation in deeper networks.

Figure 10 shows what happens without the activation. The norm increases a lot, especially in networks with more layers. This suggests the network becomes unstable without the soft threshold function.

The soft threshold function is key for controlling error amplification. It has a non-expansive property that stops $\|\hat{x}\|_2^2$ from growing too much. Without it, the norm increases fast, especially in deeper networks, leading to instability.

D. Evolution of $\mathbf{S}$ *and* $\mathbf{W}$ *Matrices During Training*

These figures show how the $\mathbf{S}$ and $\mathbf{W}$ matrices change during LISTA training. Figure 11 shows the $\mathbf{S}$ matrix. At first (Epoch 0), it has a narrow distribution centered at 0, with some values near 0.8. This is because it starts with ISTA initialization. As training goes on, the distribution gets wider and becomes more Gaussian-shaped.

Figure 12 shows the $\mathbf{W}$ matrix. It keeps a Gaussian shape throughout training, but its variance increases over time.

Both matrices end up with nearly Gaussian distributions. The $\mathbf{S}$ matrix has a narrower spread (standard deviation about 0.09) than the $\mathbf{W}$ matrix (standard deviation about 0.27). This analysis helps us understand how LISTA learns and adapts its weights during training.

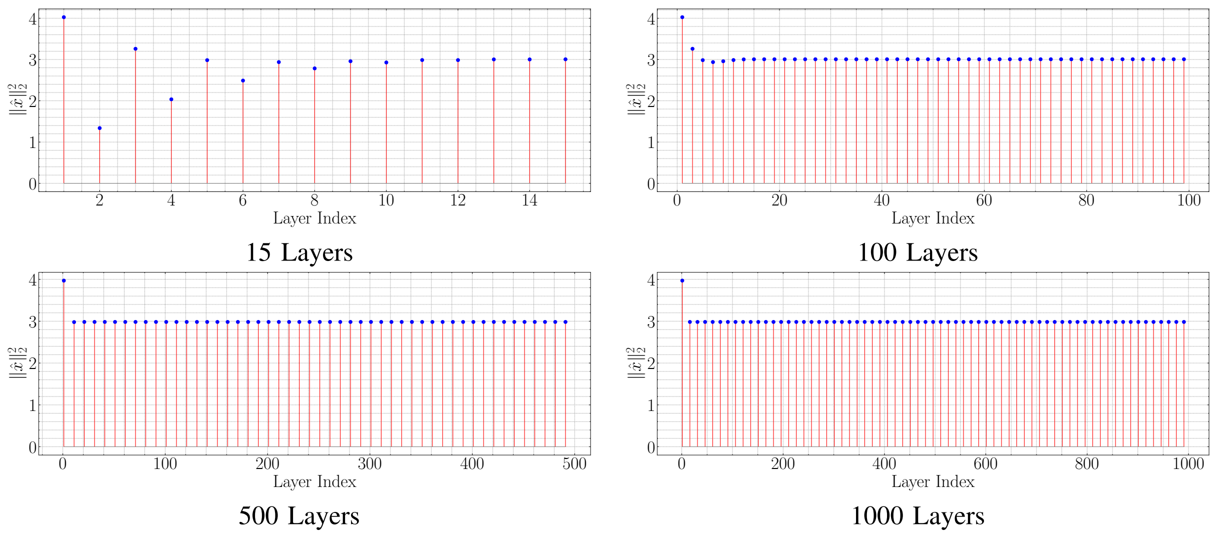


Fig. 9: Stability of LISTA with activation over different layers. It shows the squared ℓ^2 norm $\|\hat{x}\|_2^2$ for 15, 100, 500, and 1000 layers, indicating stability, and reflecting robustness with soft threshold activation.

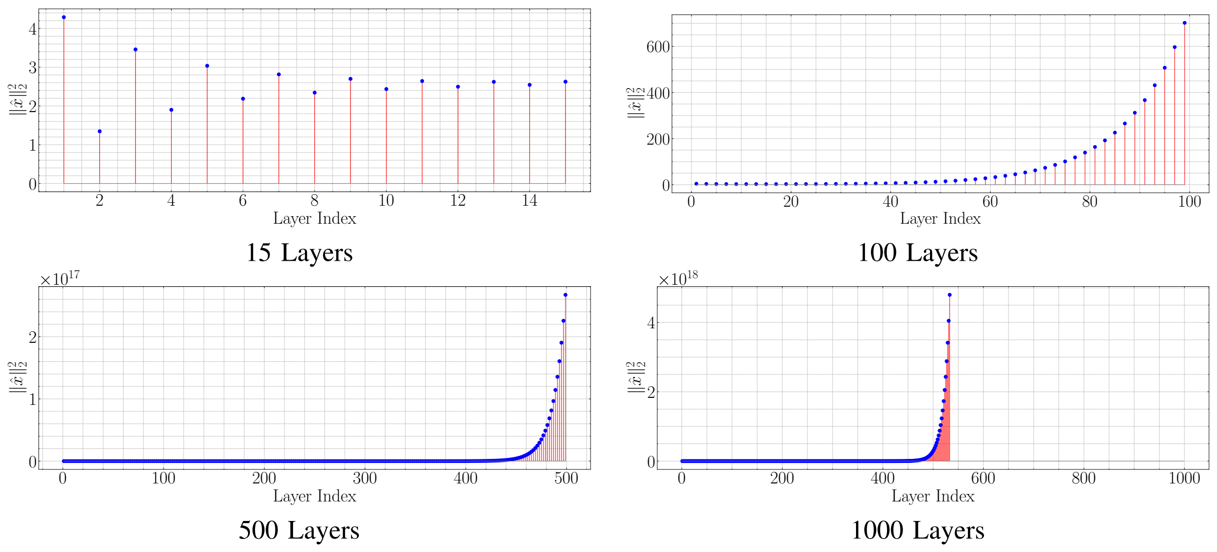


Fig. 10: Impact of removing the soft threshold activation in LISTA. The plots track the squared ℓ^2 norm $\|\hat{x}\|_2^2$ without activation across 15, 100, 500, and 1000 layers, demonstrating significant increases in the norm, especially at higher layers, which suggests instability.

III. Limitations and Future Directions

While our experiments provide valuable insights into the behavior of LISTA and related algorithms, it's important to acknowledge some limitations:

A. *Experimental Limitations*

- Our experiments focused on specific sparsity levels and noise conditions. Future work could explore a broader range of these parameters.

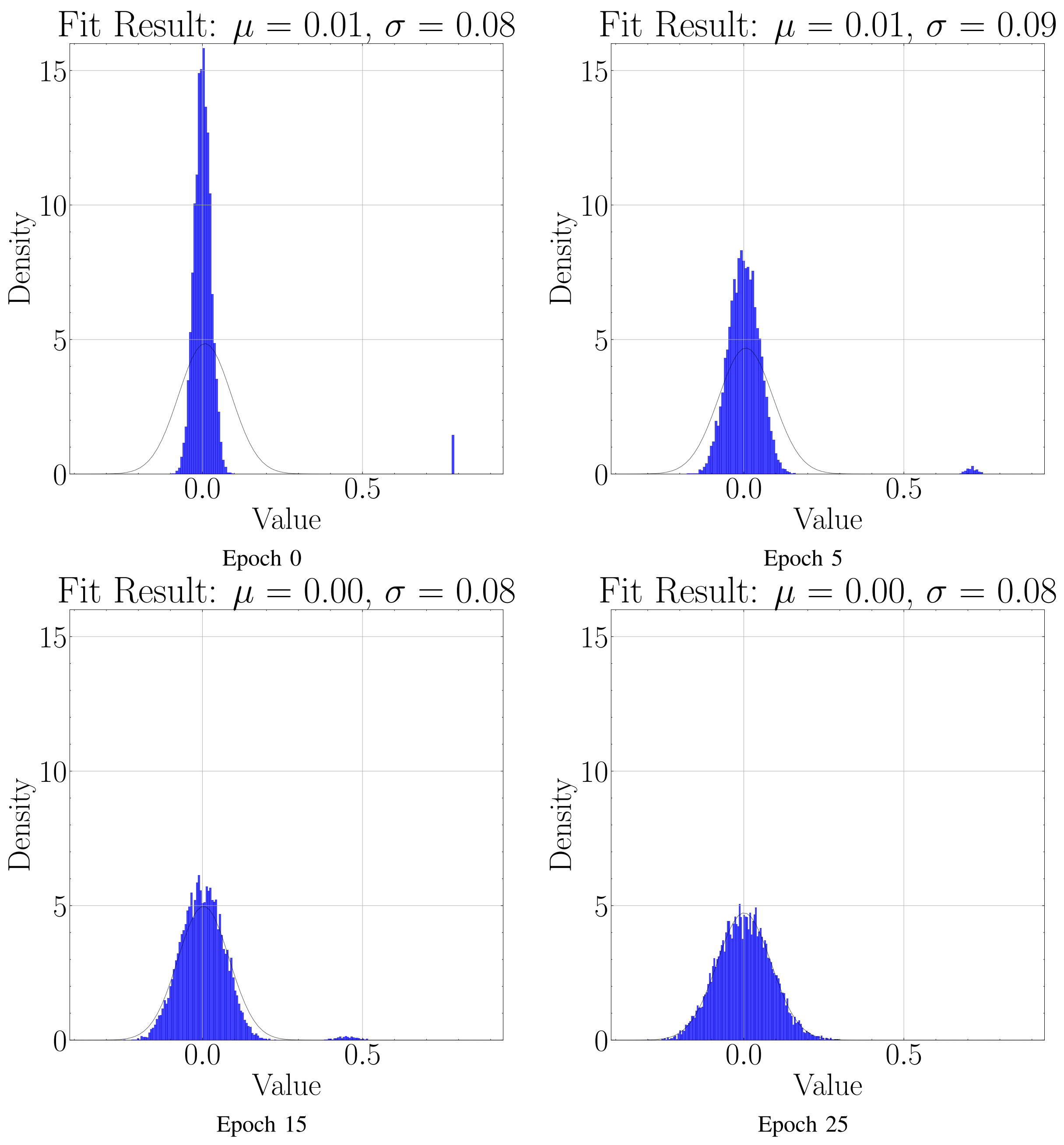


Fig. 11: Evolution of the entries of the $\mathbf{S}$ matrix over training epochs. The weights are nearly Gaussian as shown by the fit obtained using sample mean and sample variance.

- The performance of these algorithms may vary with different types of sensing matrices or signal distributions, which were not extensively tested in this study.
- The computational complexity and training time of these algorithms were not thoroughly compared, which could be important for practical applications.

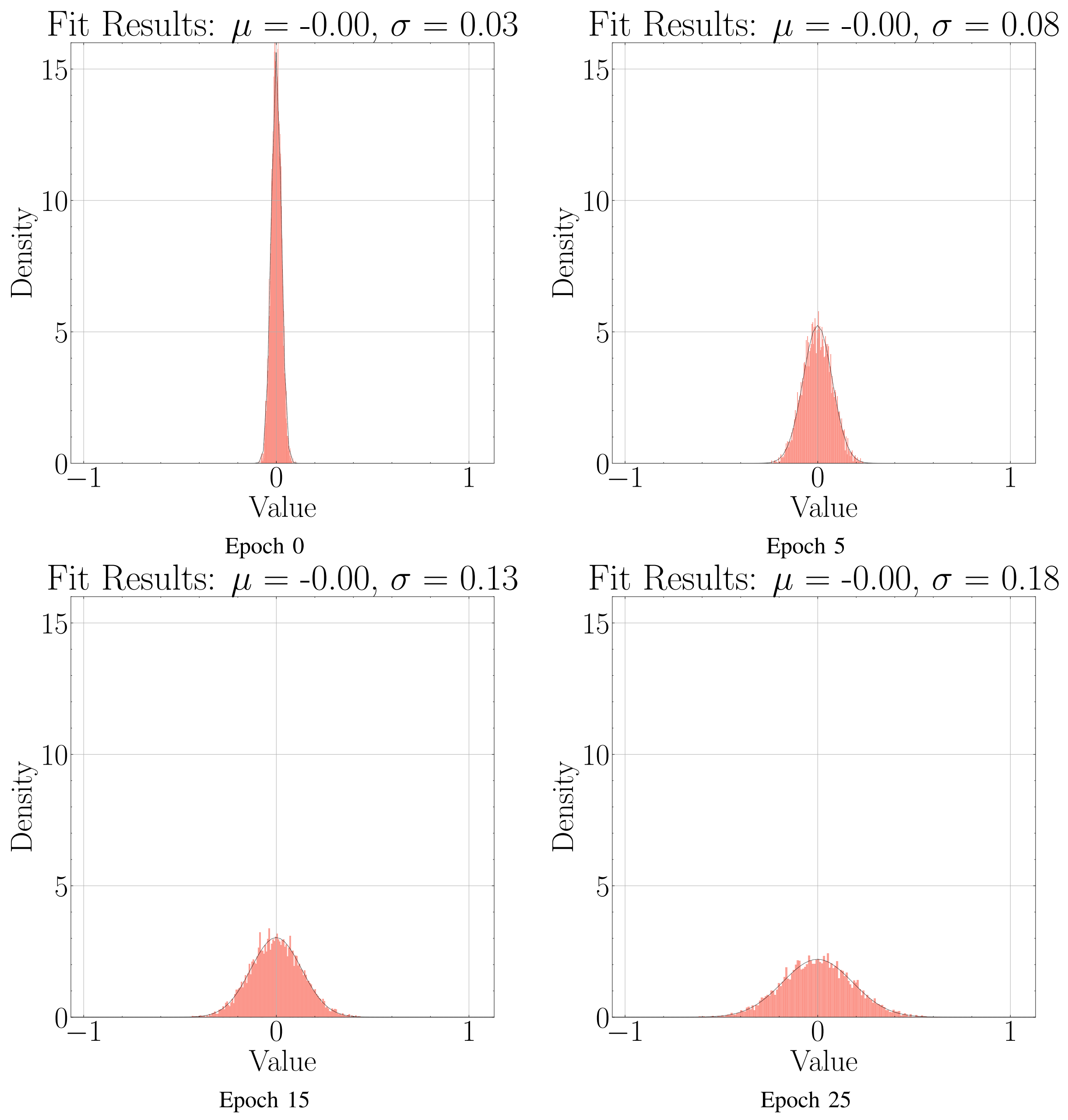


Fig. 12: Evolution of the distribution of the entries of $\mathbf{W}$ over training epochs. The Gaussian distribution is an accurate fit to the distribution of the learnt weights.

B. Unbiasing Approach: Strengths and Limitations

We introduced a novel unbiasing technique to address the bias introduced by the shrinkage-thresholding function. This approach showed promising results:

- It consistently improved reconstruction quality across all tested methods and noise levels.
- Unbiased techniques demonstrated high noise immunity, especially at lower noise levels.
- Simpler models with unbiasing (like U-ISTA and U-LISTA) outperformed more complex models in many cases.

However, the unbiasing approach also has limitations:

- Its effectiveness depends on the accuracy of the initial support estimation. When the reconstructed vector is dense or the initial reconstruction has incorrect support, the unbiasing technique may not improve performance.
- The approach assumes that the true signal is sparse, which may not always be the case in real-world scenarios.
- The computational cost of the unbiasing step, especially for large-scale problems, was not thoroughly evaluated in this study.

References

[1] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proc. Int. Conf. Mach. Learn. (ICML)*, p. 399–406, 2010.

[2] T. Tirer and R. Giryes, "Image restoration by iterative denoising and backward projections," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1220–1234, 2018.

[3] T. Tirer and R. Giryes, "Back-projection based fidelity term for ill-posed linear inverse problems," *IEEE Trans. on Image Process.*, vol. 29, pp. 6164–6179, 2020.

[4] T. Tirer and R. Giryes, "On the convergence rate of projected gradient descent for a back-projection based objective," *SIAM J. Imaging Sciences*, vol. 14, no. 4, pp. 1504–1531, 2021.

[5] K. K. R. Nareddy, A. J. Kamath, and C. S. Seelamantula, "Tight-frame-like analysis-sparse recovery using non-tight sensing matrices," *arXiv preprint arXiv:2307.10862*, 2023.

[6] K. K. R. Nareddy, A. J. Kamath, and C. S. Seelamantula, "Image restoration with a generalized ℓ_2 loss and convergent plug-and-play prior," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2024. (to appear).

[7] J. Kovačević and A. Chebira, "Life beyond bases: The advent of frames (Part I)," *IEEE Signal Process. Mag.*, vol. 24, no. 4, pp. 86–104, 2007.

[8] J. Kovačević and A. Chebira, "Life beyond bases: The advent of frames (Part II)," *IEEE Signal Process. Mag.*, vol. 24, no. 5, pp. 115–125, 2007.

[9] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[10] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

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

[12] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep cnn denoiser prior for image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 3929–3938, 2017.

[13] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1, pp. 259–268, 1992.

[14] J. Liu and X. Chen, "ALISTA: Analytic weights are as good as learned weights in LISTA," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019.

[15] S. Foucart and H. Rauhut, *A Mathematical Introduction to Compressive Sensing*. Applied and Numerical Harmonic Analysis, Springer New York, 2013.