Pivotal Auto-Encoder via Self-Normalizing ReLU

Nelson Goldenstein, Jeremias Sulam, and Yaniv Romano

***Abstract*—Sparse auto-encoders are useful for extracting low-dimensional representations from high-dimensional data. However, their performance degrades sharply when the input noise at test time differs from the noise employed during training. This limitation hinders the applicability of auto-encoders in real-world scenarios where the level of noise in the input is unpredictable. In this paper, we formalize single hidden layer sparse auto-encoders as a transform learning problem. Leveraging the transform modeling interpretation, we propose an optimization problem that leads to a predictive model invariant to the noise level at test time. In other words, the same pre-trained model is able to generalize to different noise levels. The proposed optimization algorithm, derived from the square root lasso, is translated into a new, computationally efficient auto-encoding architecture. After proving that our new method is invariant to the noise level, we evaluate our approach by training networks using the proposed architecture for denoising tasks. Our experimental results demonstrate that the trained models yield a significant improvement in stability against varying types of noise compared to commonly used architectures.**

***Index Terms*—Sparse coding, transform learning, sparse auto-encoders, square root lasso.**

I. Introduction

A sparse auto-encoder is a type of artificial neural network that learns efficient data encodings through unsupervised learning [1]. The purpose of an auto-encoder is to capture the most important elements of the input to learn a lower dimensional representation for higher dimensional data, such as images [2]. It is commonly used for dimensionality reduction or feature extraction. The sparse auto-encoder architecture consists of two modules: an encoder and a decoder. The encoder compresses the input data into an encoded representation in a different domain, which is forced to be sparse. It then processes and filters the encoded data representations so that only the most important information is allowed through, preventing the model from memorizing the inputs and overfitting. The final module of the network, the decoder, decompresses the extracted sparse representations and reconstructs the data from its encoded state back to its original domain.

Interestingly, the observation that natural data can often be accurately approximated by sparse signals has been a prominent framework over the last twenty years [3], [4], [5]. Specifically, the transform model [6]—a generalized analysis sparse representation model—assumes that a signal $x \in \mathbb{R}^n$ has a sparse representation $z^* \in \mathbb{R}^d$ over a particular transformation $W \in \mathbb{R}^{d \times n}$ to another domain, i.e.,

$$Wx = z^*, \text{ where } \|z^*\|_0 \ll d, \tag{1}$$

where $\|\cdot\|_0$ counts the number of nonzero elements of a vector. This representation is typically a higher dimensional signal, i.e., $d \geqslant n$, and this is the setting that we assume in this work. When $n = d$ and W is of full rank, the transformation forms a basis, whereas when $d > n$, the transform is considered to be overcomplete. In situations where we observe a noisy version y of the clean signal x, corrupted by additive noise, the equation becomes

$$Wy = z^* + e,$$

where e denotes the error or residual in the transform domain.

In this context, the task of finding the sparse representation of a signal given W is called sparse coding. The first module of a sparse auto-encoder—the encoder—can be formally written as a transform sparse coding problem:

$$\widehat{z} = \arg\min_z \frac{1}{2}\|z - Wy\|_2^2 + \lambda\|z\|_1, \tag{2}$$

where $\widehat{z}$ is the estimated latent space representation, W is a known transformation, and $\lambda \in \mathbb{R}_+$ is a hyperparameter. The optimization problem in (2) minimizes the transform residual e with a sparse prior for z to estimate z^*. The parameter λ governs the trade-off between sparsity and residual error. Observe that the optimal selection of λ is dependent on e since $z^* - Wy = e$; hence, the value of λ ought to be calibrated to increase with the magnitude of e.

The closed form solution to the encoding problem (2) is

$$\widehat{z} = \text{prox}_{\lambda\|z\|_1}(Wy) = S_\lambda(Wy),$$

where $\text{prox}_f(v) = \arg\min_x (f(x) + \frac{1}{2}\|x - v\|^2)$ is the proximal operator and S_λ is the soft-thresholding operator

Manuscript received 7 May 2023; revised 29 November 2023 and 1 March 2024; accepted 10 June 2024. Date of publication 26 June 2024; date of current version 23 July 2024. The work of Nelson Goldenstein and Yaniv Romano was supported by the Israel Science Foundation under Grant 729/21. The work of Jeremias Sulam was supported by the NSF under Grant CCF 2007649. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Mingyi Hong. *(Corresponding author: Nelson Goldenstein.)*

Nelson Goldenstein is with the Department of Electrical and Computer Engineering, the Technion, Haifa 3200003, Israel (e-mail: golden@campus.technion.ac.il).

Jeremias Sulam is with the Department of Biomedical Engineering and the Mathematical Institute for Data Science, Johns Hopkins University, Baltimore, MD 21218 USA.

Yaniv Romano is with the Department of Electrical and Computer Engineering and also with the Department of Computer Science, the Technion, Haifa 3200003, Israel.

Digital Object Identifier 10.1109/TSP.2024.3418971

$S_\lambda(x) = \text{sign}(x)\max(|x| - \lambda, 0)$. If we add the assumption of non-negativity of z^*, the solution can be rewritten as

$$\widehat{z} = \text{ReLU}(Wy - \lambda),$$

where $\text{ReLU}(x) = \max(x, 0)$. From this expression we understand the influence of λ on the solution and its role in filtering perturbations. Higher values of λ lead to lower and sparser solutions. Therefore, as mentioned earlier, the optimal value of λ is a function of the noise; the stronger the noise, the larger λ must be.

The decoder can also be written as an inverse operation following the same model. Formally, given W and $\widehat{z}$, we can obtain a least squares approximation to the true signal x by minimizing $\|Wx - \widehat{z}\|_2^2$ with respect to x. Thus, the recovered signal is

$$\widehat{x} = W^+\widehat{z},$$

where W^+ is the pseudoinverse of W. In particular, if W has full column rank, $W^+ = (W^\mathsf{T}W)^{-1}W^\mathsf{T}$.

The connection between the transform model (1) and sparse auto-encoders is clear. The linear transformation W represents the weights of any combination of linear layers, including convolutional operations, and λ is the bias parameter; and both are trainable parameters of the network. This connection has been successfully applied to numerous computer vision and image processing tasks [7]. When adapted to deep learning, it has demonstrated state-of-the-art performance in various machine learning applications, such as image classification [8], online video denoising [9], semantic segmentation of images [10], super-resolution [11], clustering [12], and others.

The main problem of the presented encoder algorithm (2) is the bias parameter λ. The optimal selection of λ is influenced by the noise, which means its ideal value varies with differing noise levels. Indeed, the optimality conditions for the correct estimation of the sparse representation z^* are not guaranteed at different noise intensities, even for a known W. In other words, the model's performance may deteriorate significantly if the noise level at test time is different from the one used during training [13]. Therefore, the network must be re-trained, and a different bias must be learned *for each noise level* [14]. Moreover, in an environment where the noise is unknown, as in most practical cases, finding the best bias becomes infeasible: it is impossible to choose the correct bias without estimating the noise level.

Our contribution. To overcome the dependence of the bias λ on the noise level, we draw inspiration from the square root lasso problem, introduced by [15] and detailed in Section II-A, and propose a modification of the transform sparse coding algorithm for the encoder module

$$\widehat{z} = \arg\min_z \|z - Wy\|_2 + \lambda\|z\|_1. \tag{3}$$

Notice that the residual term is no longer quadratic. The main advantage of (3) is that the hyperparameter λ is now pivotal to the noise energy. In other words, **the optimal choice of λ is independent of the noise level**, which is difficult to estimate reliably because it is an ill-posed problem [16]. We prove in Section III that this property holds in the presence of both bounded noise and additive Gaussian noise. This stands in sharp contrast to vanilla sparse auto-encoders, where one needs to know the true standard deviation of the noise to fit the bias of the original transform sparse coding problem (2).

Furthermore, we propose an efficient and differentiable algorithm to solve the new pivotal sparse coding problem based on proximal gradient descent, as described in Section IV. Our algorithm is differentiable in the sense that it is compatible with gradient-based optimization techniques, enabling the minimization of a cost function through methods such as automatic differentiation and backpropagation. This leads to the development of a novel non-linear function called Self Normalizing ReLU (NeLU), which easily integrates into common neural network architectures. In Section V, we conduct numerical experiments using both synthetic and real data to illustrate how our approach is significantly more resilient to various noise levels.

II. Related Work

A. Synthesis Sparse Modeling of Signals

We begin by introducing a framework parallel to the analysis sparse representation model, presented in Section I, named the synthesis model. This model serves as the premise for introducing the square root lasso algorithm, which we will extend to the transform model. The synthesis model assumes that a signal $x \in \mathbb{R}^n$ can be represented as a linear combination of a few columns, called atoms, from a matrix $D \in \mathbb{R}^{n\times d}$ named dictionary. In plain language, the signal corresponds to a multiplication of a dictionary by a sparse vector $z^* \in \mathbb{R}^d$, i.e.,

$$x = Dz^*.$$

Various algorithms have been proposed to implement the sparse coding task of estimating z^* given x [17]. These include the matching pursuit [18] and the basis pursuit [19], also called lasso in the statistics literature [20]

$$\min_z \frac{1}{2n}\|y - Dz\|_2^2 + \lambda_L\|z\|_1. \tag{4}$$

As with (2), the primary drawback of lasso is that the optimal value of the parameter λ_L is dependent on the noise level and, therefore, must be adjusted for each specific noise level. For example, in a Gaussian noise environment, λ_L proportional to $\sigma\sqrt{\log d/n}$ is minimax optimal for signal reconstruction, $\|\widehat{x} - x\|_2$, in high dimensions [21]. Therefore, to achieve a good estimation of z^* or prediction of x for an unknown noise level, one must estimate σ.

Square root lasso. A modified version of the lasso has been proposed to solve the dependence of λ on noise power, called square root lasso [15]

$$\min_z \frac{1}{\sqrt{n}}\|y - Dz\|_2 + \lambda_{\sqrt{L}}\|z\|_1, \tag{5}$$

which takes the square root of the error term of (4). Belloni et al. [15] have proven that the square root lasso achieves minimax optimality of estimation and signal reconstruction error for a hyperparameter $\lambda_{\sqrt{L}}$, which is pivotal to the noise

level. For instance, in the case of Gaussian noise, the minimax optimal $\lambda_{\sqrt{L}}$ is proportional to $\sqrt{\log d/n}$, which is independent of σ and leads to a constant parameter for all Gaussian distributions. Moreover, numerous algorithms have been developed to efficiently solve the problem based on its convex property [21].

Although the square root lasso is powerful and attractive, it has never been applied in the context of dictionary learning or adapted to form a novel neural network architecture. In this work, we draw an exciting connection between the square root lasso and transform modeling. The extension to synthesis model and dictionary learning is a direct consequence of the analysis of the transform learning since (3) can be seen as a particular case of (5), where $D = I$ and the input signal is Wy. Further details are provided in Appendix B.

B. Transform Learning

As elaborated in Section I, the transform model and sparse auto-encoders are tightly connected. The forward pass in sparse encoders essentially acts as a sparse representation pursuit within the transform model. In this framework, the transformation is characterized by the weights within a set of layers, which may include convolutional ones. Thus, the transformation is also learned during the model training process. This method of deriving the transformation directly from the data is known as transform learning.

At its core, transform learning [22] employs a data-driven feature extractor to transform input data into a suitable representation. This approach can improve upon the limited ability of analytical transformation methods, such as wavelets, to handle data. As a result, transform learning often yields superior restoration performance compared to the analytical approaches [6].

Transform learning is comparable to dictionary learning from an analysis perspective [23]. In dictionary learning, a basis D is trained to recover the data, $x = Dz^*$, from the representation z^* [24]. In contrast, transform learning aims to learn a transformation W to generate the representation z^*. Since the transformation W is data-dependent and the output representation z^* is unknown, jointly learning both is a challenging task.

In this work, we propose a new learning problem that combines the transform model with deep learning through the interpretable framework of transform learning. Our objective is to demonstrate that our method can exhibit the interesting properties of the classical problem established in Section III. We extend our algorithm to the transform learning framework and demonstrate its effectiveness in enhancing the robustness of deep learning through experiments in Section V.

C. Blind Denoising Networks

Blind denoising is the task of removing noise from an input signal when the noise magnitude is unknown at test time. This task is closely related to our objective. Several methods have been proposed to tackle this task, with the most common approach for blind denoising being to estimate the noise distribution (or simply the noise level) to identify and remove it from the signal [25], [26], [27]. However, this method lacks flexibility because it requires learning different weights for each noise level and solving the difficult problem of noise estimation.

In contrast to this approach, where all weights are learned from scratch for each noise level, some existing methods recognize that not all weights depend on the noise and only adjust the regularization parameters, such as the bias in the case of auto-encoders, for different noise levels [14]. This method reduces the overall number of network parameters to be learned, but they still need to be adjusted for each noise level.

Another standard technique, which emerged alongside the advancement of deep learning in computer vision tasks, is to train one model across a wide range of expected noise levels [28]. In this case, the denoising performance of such a model is generally inferior compared to a model trained for a specific noise distribution [29]. Moreover, it has been shown that a model trained using this method tends to focus on the average noise level of the training range, rather than learning generalizable weights for all noise levels. To address this issue, Gnansambandam et al. [29] proposed determining the optimal noise training sample distribution from a minimax risk optimization perspective. The approach proposed in [29] is orthogonal to ours, as it does not suggest modifying the network architecture but instead focuses solely on the training strategy.

In this work, we propose a new architecture for implementing (3) that is inherently noise level independent. Our theoretical study, presented in Section III, shows that the same model parameters achieve high-quality signal recovery across all noise levels when learned correctly. Indeed, our experimental results indicate that a single neural network can be trained and practically applied to handle all noise levels, without re-training or updating the bias term.

III. Theory

We begin by formally restating the transform model from Section I. Specifically, we consider a sparse linear model in high dimensions for a noisy signal

$$y = x + \xi,$$

where $x \in \mathbb{R}^n$ is the clean sparsifiable signal and $\xi \in \mathbb{R}^n$ denotes the random additive noise. Thus, applying a given transformation W to y yields

$$Wy = W(x + \xi) = Wx + W\xi = z^* + e.$$

Here, the goal is to recover the clean sparse representation $z^* = Wx$, while $e = W\xi$ is the error.

In the following subsections, we demonstrate that the desired properties of the square root lasso also hold for the sparse encoding problem (3), given a known transform W. Specifically, the parameter λ is pivotal to the noise level, and as a result, not only can the solution of (3) be computed efficiently, but all parameters of the problem are also independent of the noise level. We prove that the recovery of the correct support, i.e., the group of nonzero elements $\{i \in [d] : |z_i^*| > 0\}$, and the

bound on the estimation error, $\|\widehat{z}-z^*\|_2$, can be extended to our proposed new optimization problem under the presence of bounded noise.

A. Support Recovery

First, we prove the recovery of the correct support in the presence of bounded noise, a prevalent scenario in the robustness literature [30], [31]. Subsequently, we extend these results to Gaussian noise within a probabilistic setting. To prove these results, the following assumptions are necessary.

Assumption 1: The noise $\|\xi\|_2$ is bounded.

Consequently, $\|e\|_2$ is also bounded since

$$\|e\|_2 \leqslant s_{\max}\|\xi\|_2 \triangleq \epsilon,$$

where $s_{\max}$ is the largest singular value of W.

Assumption 2: There exists a minimal positive number η that satisfies the condition

$$\lambda\|z^*\|_1 \leqslant \eta\epsilon.$$

The constant η represents the ratio between the regularization term $\lambda\|z^*\|_1$ and the noise threshold ϵ, which is the value of the reconstruction error term when $\widehat{z}=z^*$. A smaller value of η implies that the regularization term is proportionally smaller, leading to reduced shrinkage and potentially more accurate signal recovery.

Theorem 1: Let Assumption 1 be satisfied, let η satisfy Assumption 2. Then, for

$$\lambda = \frac{\|e\|_\infty}{\|e\|_2},$$

and $\widehat{z}$ be the solution to (3), we get that

$$\|\widehat{z}-z^*\|_\infty \leqslant \lambda(2+\eta)\epsilon.$$

Moreover, if

$$\min_{j\in[d]} |z_j^*| > 2\lambda(2+\eta)\epsilon,$$

then the estimated support

$$\widehat{\mathcal{S}} = \{j\in[d] : |\widehat{z}_j| > \lambda(2+\eta)\epsilon\}$$

recovers the true sparsity pattern $\mathcal{S}=\{j\in[d]:|z_j|>0\}$ correctly, i.e.,

$$\widehat{\mathcal{S}} = \mathcal{S}.$$

Proof of Theorem 1: Our derivation is inspired by the one in [32]. For the solution $\widehat{z}$ of (3), we have:

$$\begin{aligned}\|\widehat{z}-z^*\|_\infty &= \|\widehat{e}-e\|_\infty \qquad (\widehat{e}=Wy-\widehat{z})\\ &\leqslant \|\widehat{e}\|_\infty + \|e\|_\infty\\ &\leqslant \|\widehat{e}\|_\infty + \lambda\epsilon. \end{aligned} \tag{6}$$

We bound $\|\widehat{e}\|_\infty$ using KKT sub-gradient optimality conditions,

$$\|\widehat{e}\|_\infty \leqslant \lambda\|\widehat{e}\|_2. \tag{7}$$

It now remains to bound $\|\widehat{e}\|_2$, which is done with Assumption 2. By minimality of the estimator,

$$\begin{aligned}\|\widehat{e}\|_2 + \lambda\|\widehat{z}\|_1 &\leqslant \|e\|_2 + \lambda\|z^*\|_1\\ \|\widehat{e}\|_2 &\leqslant \|e\|_2 + \lambda\|z^*\|_1\\ &\leqslant \epsilon + \eta\epsilon. \end{aligned} \tag{8}$$

Combining equations (6), (7) and (8), we get:

$$\|\widehat{z}-z^*\|_\infty \leqslant \|\widehat{e}\|_\infty + \lambda\epsilon \leqslant \lambda(\eta+1)\epsilon + \lambda\epsilon.$$

This proves the bound on $\|\widehat{z}-z^*\|_\infty$. Finally, the correct support recovery follows directly from Theorem 4.1 in [33]. □

Remark 1: It is essential to highlight that λ maintains the same value across different noise levels. For example, let ξ_1 be a realization from a standard normal distribution $\mathcal{N}(0,I)$ and define $\xi_2=\sigma\xi_1$, for any standard deviation $\sigma\in\mathbb{R}_+$. Consequently, we obtain

$$\lambda = \frac{\|e\|_\infty}{\|e\|_2} = \frac{\|W\xi_1\|_\infty}{\|W\xi_1\|_2} = \frac{\|W\xi_2\|_\infty}{\|W\xi_2\|_2}.$$

This observation underscores the importance of our choice of λ, as it is independent of σ and remains consistent across various noise levels.

On a different note, we can deduce that the correct support can be fully recovered if the signal-to-noise ratio is sufficiently high, as formally stated in Theorem 1. Furthermore, λ can be conveniently bounded in all cases by

$$\frac{1}{\sqrt{n}} \leqslant \lambda = \frac{\|e\|_\infty}{\|e\|_2} \leqslant 1.$$

A bound for λ can be used to guarantee that no false positive support error occurs. Improved bounds can be achieved with additional assumptions on the noise. In fact, we investigate this aspect for the Gaussian case in Section III-C.

B. Estimation Error

We now proceed to show that the estimation error, $\|\widehat{z}-z^*\|_2$, can also be bounded with the same choice of parameter λ, under the same Assumptions 1 and 2.

Theorem 2: Let Assumption 1 be satisfied and let η satisfy Assumption 2. Then, for $\widehat{z}$, the solution to (3), we get

$$\|\widehat{z}-z^*\|_2 \leqslant (2+\eta)\epsilon.$$

Proof of Theorem 2: From the optimality of the solution:

$$\|\widehat{z}-Wy\|_2 + \lambda\|\widehat{z}\|_1 \leqslant \|z^*-Wy\|_2 + \lambda\|z^*\|_1.$$

Using $Wy=z^*+e$, and applying the reverse triangle inequality, we get

$$\begin{aligned}\|\widehat{z}-z^*-e\|_2 &\leqslant \|e\|_2 + \lambda\|z^*\|_1 - \lambda\|\widehat{z}\|_1\\ \|\widehat{z}-z^*\|_2 - \|e\|_2 &\leqslant \|e\|_2 + \lambda\|z^*\|_1 - \lambda\|\widehat{z}\|_1\\ \|\widehat{z}-z^*\|_2 &\leqslant 2\|e\|_2 + \lambda\left(\|z^*\|_1 - \|\widehat{z}\|_1\right).\end{aligned}$$

Finally, using Assumptions 1 and 2, we have

$$\|\widehat{z}-z^*\| \leqslant 2\|e\| + \lambda\|z^*\|_1 \leqslant (2+\eta)\epsilon,$$

which concludes the proof. □

In Appendix F, we present an empirical validation of Theorem 2.

C. Gaussian Noise

We now extend the results of Section III-A to the Gaussian case. The key point of this analysis is that we can use practical values for λ, which can be computed independently of the noise level. We show that these λ values are suitable for any additive Gaussian noise in the input and are thus pivotal to its standard deviation σ.

Assumption 3: The entries of ξ are i.i.d. $\mathcal{N}(0,\sigma^2)$ random variables.

Assumption 4: The rows of W are normalized to unit ℓ_2 norm.

Theorem 3: Let Assumption 3 and 4 be satisfied, let η satisfy Assumption 2. Then, set

$$\lambda = a\sqrt{\frac{2\log d}{n}},$$

where $a \geqslant 2\sqrt{2}$ is a constant.

With probability at least $1 - 2d^{1-a^2/8} - (1+e^2)e^{-n/24}$, we have

$$\|\widehat{z} - z^*\|_\infty \leqslant \lambda\left(2 + 2\eta + \frac{1}{s_{\min}}\right)\sqrt{n}s_{\max}\sigma,$$

where $s_{\min}$ and $s_{\max}$ are the minimum and maximum singular values of W, respectively.

Proof of Theorem 3: Let $\mathcal{A}$ be the event

$$\mathcal{A} = \left\{\frac{\|e\|_\infty}{\|e\|_2} \leqslant \frac{\lambda}{2s_{\min}}\right\} \cap \left\{s_{\min}\frac{\sigma}{\sqrt{2}} < \frac{\|e\|_2}{\sqrt{n}} < 2s_{\max}\sigma\right\}.$$

From Lemma 1 presented in Appendix A, it is deduced that $\mathbb{P}(\mathcal{A}) \geqslant 1 - 2d^{1-a^2/8} - (1+e^2)e^{-n/24}$. For our model and under the event $\mathcal{A}$, we have

$$\begin{aligned}\|\widehat{z} - z^*\|_\infty &\leqslant \|\widehat{e}\|_\infty + \|e\|_\infty \\ &\leqslant \|\widehat{e}\|_\infty + \lambda\frac{s_{\max}}{s_{\min}}\sqrt{n}\sigma. \end{aligned} \tag{9}$$

By minimality of the estimator and Assumption 2,

$$\begin{aligned}\|\widehat{e}\|_2 &\leqslant \|e\|_2 + \lambda\|z^*\|_1 \\ &\leqslant 2\sqrt{n}s_{\max}\sigma + 2\eta\sqrt{n}s_{\max}\sigma. \end{aligned} \tag{10}$$

Combining equations (7), (9) and (10), we get:

$$\|\widehat{z} - z^*\|_\infty \leqslant \lambda\left(2\sqrt{n}s_{\max}\sigma + 2\eta\sqrt{n}s_{\max}\sigma\right) + \lambda\frac{s_{\max}}{s_{\min}}\sqrt{n}\sigma$$

$$\|\widehat{z} - z^*\|_\infty \leqslant \lambda\left(2 + 2\eta + \frac{1}{s_{\min}}\right)\sqrt{n}s_{\max}\sigma.$$

□

IV. Computational Algorithm

A. An Iterative Solver

In this section, we introduce an iterative optimization algorithm for minimizing (3) that can be efficiently implemented and formulated as a novel sparse auto-encoder architecture. It is worth noting that this objective function corresponds to a convex optimization problem. Therefore, it inherits not only all the theoretical properties of convex optimization problems, but also the algorithms that can be used to solve it, such as the interior point method [34] or the alternating direction method of multipliers [35].

Algorithm 1 Self Normalizing ReLU (NeLU)

Input:

$\bar{y} \leftarrow Wy$, where W is the transform and y its input signal.

Parameters:

λ – bias.

β – step size.

Output:

The estimated representation $\widehat{z}$.

Process:

$\widehat{z} \leftarrow 0$

While not converged:

$\widehat{z} \leftarrow S_{\beta\lambda}\left\{\widehat{z} - \beta\frac{\widehat{z}-\bar{y}}{\|\widehat{z}-\bar{y}\|_2}\right\}$

return $\widehat{z}$

In [21], the authors studied the geometric structure of the square root lasso problem and concluded that the ℓ_2 loss function is non-differentiable only in extreme cases of overfitting. In practice, this situation is rare when the data are corrupted by noise and a sufficiently large regularization parameter λ is used to produce a sparse solution. Consequently, the data fitting term in the objective function behaves as a strongly convex and smooth function.

Leveraging these attractive geometric properties, we can use proximal gradient descent to iteratively minimize (3). The theoretical analysis in [21] shows that such an optimization algorithm achieves fast local linear convergence. The same theoretical justification also applies to (3) since our optimization problem can be viewed as a simpler instance of the square root lasso, presented in Section II-A. Therefore, we propose adapting proximal gradient descent to the problem studied here, as described in Algorithm 1, which we have named Self Normalizing ReLU or NeLU for short.

The algorithm works by continuously refining the solution via iterative updates, progressing in the direction opposite to the gradient of the objective function. Each iterative update incorporates a proximal operator, which introduces a penalty term to the objective function, thereby promoting sparsity. In the specific problem at hand, the proximal operator takes the form of a soft-thresholding operator, as outlined in Section I. This operator performs a shrinkage operation on the variables, setting any values below a specified absolute threshold to zero. Importantly, the soft-thresholding operator can be replaced with ReLU to enforce nonnegative representations. The iterative process continues until the desired level of convergence is achieved.

B. Transform Learning

We propose to adapt Algorithm 1 for transform learning by unrolling the algorithm into a layered neural network architecture, following the approach presented in [36]. The idea is to unfold an iterative algorithm and construct it as a network architecture, mapping each iteration to a single operation and stacking a finite number of operations on top of each other.

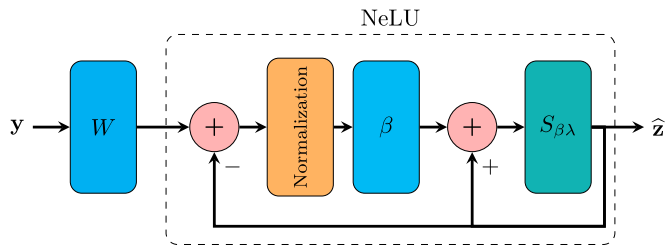


Fig. 1. The NeLU architecture: A recurrent sparse encoder model, unrolled for a predetermined number of iterations.

Algorithm 2 Accelerated NeLU

Input:

$\bar{y} \leftarrow Wy$, where W is the transform and y its input signal.

Learnable weights:

λ – bias.
β – step size.
α – momentum factor.

Hyperparameters:

N – number of iterations.

Output:

The estimated representation $\widehat{z}$.

Process:

$\widehat{z} \leftarrow 0$
$v \leftarrow 0$
For $i = 1 : N$
$v \leftarrow \alpha v - \beta \frac{\widehat{z}+\alpha v-\bar{y}}{\|\widehat{z}+\alpha v-\bar{y}\|_2}$
$\widehat{z} \leftarrow \text{ReLU}\left(\widehat{z} + v - \beta\lambda\right)$
return $\widehat{z}$

This approach enables the incorporation of a wide range of mathematical techniques into deep learning models [37], [38], [39]. Specifically, we achieve this unfolding by limiting the number of iterations in Algorithm 1 to N iterations. The resulting architecture is depicted in Fig. 1.

In addition, we propose to improve the performance of the algorithm by employing Nesterov acceleration [40]. Nesterov acceleration is a variant of momentum that speeds up the convergence of gradient descent algorithms, and has demonstrated efficacy in various contexts. By incorporating it into Algorithm 1, we aim to achieve superior performance in transform learning tasks, based on the understanding that accelerated gradient descent converges faster and operates effectively with shallower networks, which are easier to train.

Finally, we note that all parameters of the resulting algorithm, including W, λ, β, and α, can be trained end-to-end. This means that the network can be trained on a dataset to learn the optimal values for these parameters, allowing it to perform well on various transform learning tasks. The final accelerated algorithm is presented in Algorithm 2.

At each iteration, the algorithm computes the gradient of the objective function with respect to the model parameters at a point in the direction of the momentum and updates the momentum in the opposite direction of the gradient. The solution is then updated based to the momentum, taking into account the proximal operator. This operator introduces regularization to prevent overfitting and improve the generalization performance of the model.

V. Experiments

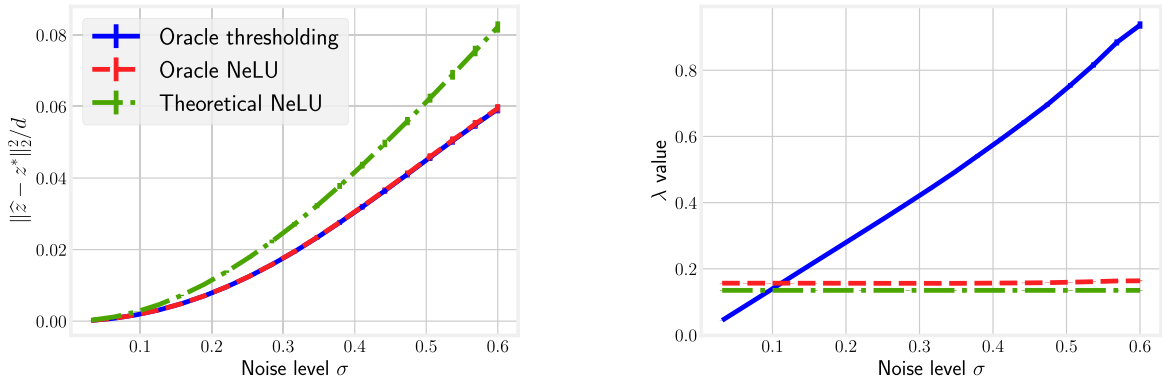
In this section, we present experimental results to evaluate the effectiveness of our proposed method under three different settings. First, in Section V-A, we use synthetic data to compare the performance of the soft-thresholding algorithm (2) with that of our proposed algorithm (3), which we minimize using the iterative approach detailed in Algorithm 1, *given a known transformation*. Next, in Section V-B, we use the same synthetic data to assess the trainable version of our method, where *the transformation matrix is also learned*, as summarized in Algorithm 2. Here, we compare our method with a baseline model based on a standard sparse auto-encoder. Lastly, in Section V-C, we evaluate the performance of our trainable Algorithm 2 against a baseline convolutional neural network in the task of image denoising.

A. Synthetic Data

First, we present experiments conducted on synthetic data to demonstrate the advantage of our proposed method over the traditional sparse encoder algorithm (2). We assume that the transformation W is known and construct a 100×100 random matrix, where each entry is sampled from the standard normal distribution, and then normalize the rows to have unit ℓ_2 norm to satisfy Assumption 4. Next, we generate the input signal by creating a vector z^* with fixed sparsity level, following the procedure described in [41], to obtain a signal consistent with the transform model, such that $\|Wx\|_0 = 5$. Finally, we contaminate the signal with i.i.d. Gaussian noise of level σ to produce the measurements $y = x + \xi$.

We evaluate the estimation error, $\|\widehat{z} - z^*\|_2$, which measures the distance between the estimated signal $\widehat{z}$ and the true signal z^*, as a function of the noise standard deviation σ. We compare the solutions obtained by minimizing (2) and (3) in two settings. In the first setting, we perform an oracle cross-validation that sweeps over a range of parameters λ to find the regularization parameter that minimizes the estimation error for each algorithm. Importantly, this setting is infeasible since it requires the ground truth data to calculate the estimation error. Nevertheless, it reveals the best performance that one can hope to achieve. In the second setting, we consider a more realistic scenario where we compare the performance of the proposed Algorithm 1 using the theoretical value of λ divided by 2, following Belloni's empirical improvement [15].

Fig. 2 shows that both algorithms achieve similar performance in the oracle setting. However, the optimal value of λ for the soft-thresholding algorithm (2) is linearly proportional to the noise level σ, while the optimal value for the proposed algorithm (3) is pivotal to it. This conclusion is consistent with our theoretical analysis presented in Section III. Additionally,



(a) (b)

Fig. 2. Experimental results for analytical transform with synthetic data. (a) Mean squared error (MSE) of ℓ_2 estimation error as a function of the noise level σ, evaluated in both settings. In the oracle setting, the regularization parameter is tuned to achieve the smallest estimation error. In the theoretical setting, we use $\lambda = \frac{1}{2}\frac{\|e\|_\infty}{\|e\|_2}$ in Algorithm 1. (b) λ values used for each algorithm in the previous graph. Note that the optimal λ values for Algorithm 1 are constant, while they are linear for the traditional algorithm. The standard errors are below 0.02 and thus barely visible.

we observe that the theoretical value of λ for the proposed algorithm, described in Theorem 1, is very close to the actual optimal value. This indicates that the proposed optimization problem may be a reasonable alternative to the current soft-thresholding algorithm, which forms classic encoder architectures, in practical situations where the noise level is unknown.

B. Trainable Transforms

Trainable transforms often outperform analytical transformations, such as total variation and wavelets, in most signal processing applications [6]. This is because the transformation used in signal processing is frequently unknown and must be inferred from the data. This motivates the use of neural networks to simultaneously learn the transformation and the sparse representation of the data.

The first learning task is supervised sparse coding, where the input consists of signals y determined by the model and their corresponding synthetic sparse vectors z^* generated by a sparsifying transform, as described in Section V-A. In this case, the goal of the neural network is to learn the transformation W stored in its weights and accurately identify the corresponding sparse output vector given a set of input-output pairs. Mathematically, this can be expressed as an end-to-end training scheme, minimizing the cost function

$$\min_{W,\lambda,\alpha,\beta} \|\widehat{z} - z^*\|_2^2,$$

where $\widehat{z}$ is the output of the network outlined in Fig. 1.

To compare the effectiveness of the proposed approach, two different neural network architectures are used: one based on Algorithm 2, and a baseline sparse encoder architecture. Both networks consist of a linear layer followed by a thresholding layer. In the baseline version, the thresholding layer is the soft-thresholding non-linearity. In contrast, our proposed architecture uses the Accelerated NeLU non-linearity as presented in Algorithm 2, with the soft-thresholding operator. Both networks are trained using the AdamW optimizer [42] and minimize the mean squared error (MSE) loss with a fixed noise standard deviation of $\sigma = 0.1$. After training, we evaluate the performance of the fitted networks on different noise levels, σ, than those used during training. The results, displayed in Fig. 3, suggest

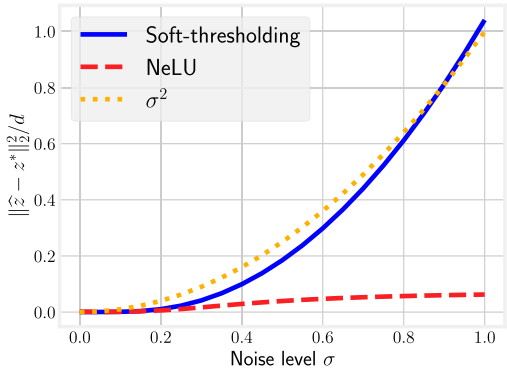


Fig. 3. Synthetic supervised sparse coding: a comparison of mean squared error (MSE) for estimation error between a two-layer sparse encoder architecture with NeLU and a similar architecture with soft-thresholding, trained on data with a fixed noise level of 0.1. The performance is evaluated at different noise levels, averaged over 2048 realizations of the data.

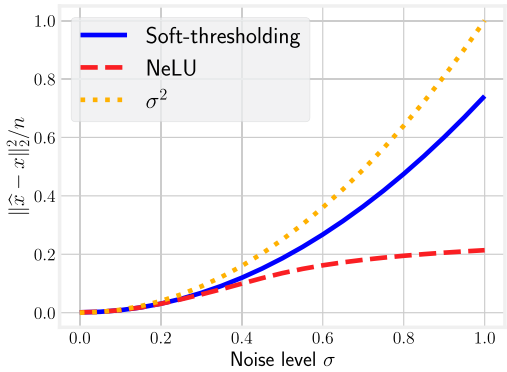


Fig. 4. Synthetic sparse signal denoising: a comparison of the mean squared error (MSE) for the reconstruction error, $\widehat{x} - x$. Other details are the same as in Fig. 3.

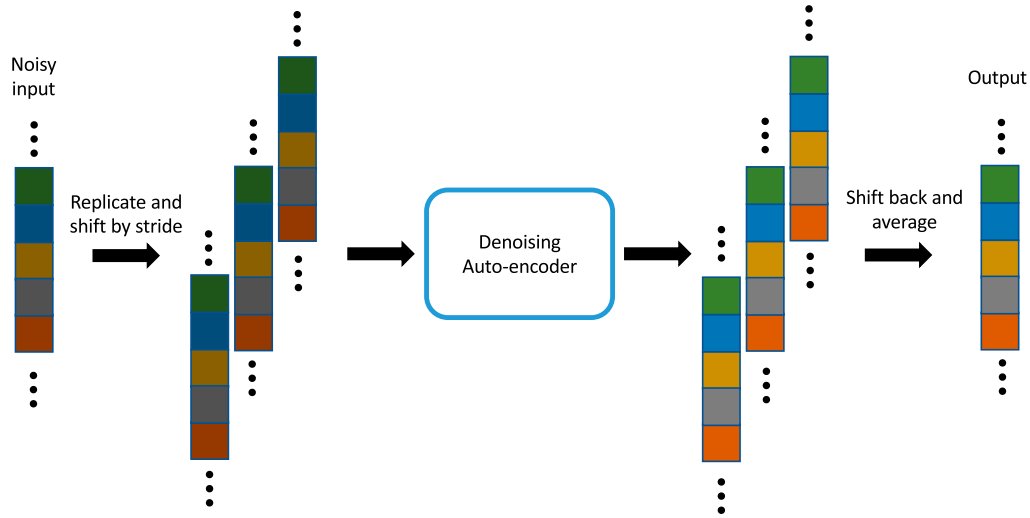


Fig. 5. 1D example of the process utilized in the experiment in Section V-C when *stride* = 2. Each signal is replicated *stride* times and subsequently translated, yielding slight shifts of the original for each replica. This process effectively transforms a single image into a collection of *stride* variations, each exhibiting a slight spatial offset. The final output is an average of all denoised shifts.

TABLE I
PSNR COMPARISON OF SPARSE AUTO-ENCODER MODELS WITH NELU AND RELU ACTIVATIONS FOR NATURAL IMAGE DENOISING. THE MODELS ARE TRAINED ON CLEAN-NOISY IMAGE PAIRS WITH A FIXED NOISE LEVEL OF $\sigma = 15$ AND EVALUATED ON THE TEST SET

σ	15	25	35	50	75	90	105	120
Noise	24.61	20.17	17.25	14.15	10.63	9.05	7.70	6.55
ReLU	28.47	25.89	**23.49**	20.58	17.07	15.48	14.13	12.99
NeLU	**28.65**	**25.93**	23.48	**20.69**	**17.62**	**16.37**	**15.36**	**14.57**

that NeLU is significantly more robust to unseen noise levels than soft-thresholding, indicating that the NeLU non-linearity results in a predictive model that generalizes to other noise levels without additional training.

We also demonstrate the effectiveness of the proposed model in signal denoising applications. In this task, the input is a noisy signal $y = x + \xi$ and the goal is to produce a cleaned version of the signal, $\widehat{x} = W^{+}\widehat{z}$, by removing the additive noise. To this end, a final linear layer is added to each network according to the sparse model. The first two layers perform sparse coding, i.e., they estimate the sparse representation, while the last layer projects the sparse estimation back into the input space. Sparse data is generated in the same manner as before and the MSE loss is minimized. The results, shown in Fig. 4, again reveal that the NeLU leads to more stable recovery than the soft-thresholding non-linearity.

C. Natural Images

In this experiment, the aforementioned networks are employed to perform natural image denoising using a patch averaging technique based on the Convolution Sparse Coding model [43]. To accomplish this, each input image is replicated *stride* times and translated across every dimension, producing slight shifts of the original for each replica. As a result, a single image transforms into a set of $stride^2$ slightly offset variations. These shifted versions of the input are then processed collectively by the network, resulting in intermediate (shifted) denoised versions of the same input image. Finally, these intermediate denoised output images are shifted back and averaged to yield the final reconstructed output image. This process is visualized in Fig. 5 for a one-dimensional (1D) signal. For more details, see [43].

The datasets and preprocessing procedures are adopted from [43]. Specifically, we use clean training images from the Waterloo Exploration dataset [44], and a validation set consisting of 432 images from BSD [45]. Noisy images are generated by adding white Gaussian noise with a constant standard deviation $\sigma = 15$. In each iteration, we randomly crop a patch of size 128^2 from an image and obtain a random realization of the noise.

In this setting, we replace the linear layers with convolution and deconvolution layers, respectively. Concurrently, the soft-thresholding operator is substituted by ReLU as the proximal operator. The models learn 175 filters of dimensions 11×11 and a stride of 8. We utilize the AdamW optimizer with a learning rate of $2 \cdot 10^{-2}$, which is reduced by a factor of 0.7 after every 50 epochs. Additionally, the optimizer's ϵ parameter is set to 10^{-3} to ensure stability. The models are trained for 300 epochs, minimizing the MSE loss.

To evaluate the performance of the models, we use the BSD68 dataset, which is distinct from the validation set. The experimental results, as shown in Table I, allow us to compare the performance of each model on the test dataset at varying

noise levels. We can see that the proposed Algorithm 2 layer outperforms the ReLU activation function for virtually all noise levels, and the performance gap widens as the noise level deviates further from the trained noise level.

VI. Conclusion

In this work, we proposed a novel sparse auto-encoder architecture as an alternative to traditional auto-encoder architectures. We offer a novel activation function, called Self Normalizing ReLU (NeLU), which is the solution of a square root lasso problem under a transform learning formulation. Importantly, as we showed in Section III, the bias parameter of our proposed NeLU layer is pivotal (i.e., invariant) to the noise level in the input signal. This feature leads to an activation function that is significantly more robust to varying noise levels in terms of signal recovery and denoising, both on synthetic data as well as in real imaging settings. Our research showcases how theoretical understanding of neural networks can give rise to improved algorithms, derived from theoretical insights and analysis.

Several open questions present opportunities for future directions. While our paper focuses on establishing foundational theory, future efforts might apply these insights on a larger scale to develop a state-of-the-art network. A potential direction would be to extrapolate the model to a multilayer architecture, building upon the work reported in [46]. The multilayer expansion strategy proposed by [47] also offers an attractive option. Incorporating additional layers into the model, and possibly broadening the analysis to convolutional neural network (CNN) architectures, could result in improved theoretical bounds and performance.

Appendix A
Lemmas

Lemma 1: Consider a Gaussian vector $\xi \sim \mathcal{N}(0, \sigma^2 I)$ and a deterministic matrix W with normalized rows, where $s_{\min}$ and $s_{\max}$ denote the minimum and maximum singular values of W, respectively. Then,

1) Let $\mathcal{C}_1 \triangleq \left\{ \frac{1}{\sqrt{n}} \|W\xi\|_\infty \leqslant \frac{\lambda}{2} \right\}$. Take $\lambda = a\sigma\sqrt{\frac{\log d}{n}}$ and $a > 2\sqrt{2}$, then:

$$\mathbb{P}(\mathcal{C}_1) \geqslant 1 - 2d^{1-a^2/8}.$$

2) Let $\mathcal{C}_2 \triangleq \left\{ \frac{1}{\sqrt{n}} \|W\xi\|_\infty \leqslant \frac{\lambda}{2}\frac{\sigma}{\sqrt{2}} \right\}$. Take $\lambda = a\sqrt{\frac{2\log d}{n}}$ and $a > 2\sqrt{2}$, then:

$$\mathbb{P}(\mathcal{C}_2) \geqslant 1 - 2d^{1-a^2/8}.$$

3) Let $\mathcal{C}_3 \triangleq \left\{ s_{\min}\frac{\sigma}{\sqrt{2}} < \frac{\|W\xi\|_2}{\sqrt{n}} < 2s_{\max}\sigma \right\}$, then:

$$\mathbb{P}(\mathcal{C}_3) \geqslant 1 - (1+e^2)e^{-n/24}.$$

4) Let

$$\mathcal{A} = \left\{ \frac{\|W\xi\|_\infty}{\|W\xi\|_2} \leqslant \frac{\lambda}{2s_{\min}} \right\} \cap \left\{ s_{\min}\frac{\sigma}{\sqrt{2}} < \frac{\|W\xi\|_2}{\sqrt{n}} < 2s_{\max}\sigma \right\},$$

then:

$$\mathbb{P}(\mathcal{A}) \geqslant 1 - \mathbb{P}(\mathcal{C}_2^c) - \mathbb{P}(\mathcal{C}_3^c).$$

Proof: Item 1: Since ξ is isotropic, the law of $d^\mathsf{T}\xi$ is the same for all vectors $d \in \mathbb{R}^n$ of the same norm. In particular, $W_i\xi$, where W_i is the ith row of W, and ξ_1 have the same law.

$$\begin{aligned}
\mathbb{P}\left(\mathcal{C}_1^c\right) &\leqslant \sum_i^d \mathbb{P}\left(|W_i\xi| \geqslant \sqrt{n}\lambda/2\right) \\
&\leqslant d\,\mathbb{P}\left(|\xi_1| \geqslant \sqrt{n}\lambda/2\right) && (\|W_i\|_2 = 1) \\
&\leqslant 2d\exp\left(-\frac{n\lambda^2}{8\sigma^2}\right) && \text{(Hoeffding's Inequality)} \\
&\leqslant 2d^{1-\frac{A^2}{8}} && \left(\lambda = A\sigma\sqrt{\frac{\log d}{n}}\right).
\end{aligned}$$

Item 2 is a direct consequence of Item 1.

Item 3: Giraud [48] controls the event

$$\left\{ \frac{\sigma}{\sqrt{2}} \leqslant \frac{\|\xi\|_2}{\sqrt{n}} \leqslant \left(2 - \frac{1}{\sqrt{2}}\right)\sigma \right\}.$$

with probability $(1+e^2)e^{-n/24}$. Therefore, we can control our event by

$$s_{\min}\frac{\|\xi\|_2}{\sqrt{n}} \leqslant \frac{\|W\xi\|_2}{\sqrt{n}} \leqslant s_{\max}\frac{\|\xi\|_2}{\sqrt{n}}.$$

Proof of Item 4 is done using items 2 and 3. Indeed we have

$$\mathcal{A} \supset \left\{ s_{\min}\frac{\sigma}{\sqrt{2}} < \frac{\|W\xi\|_2}{\sqrt{n}} < 2s_{\max}\sigma \right\} \cap \left\{ \frac{\|W\xi\|_\infty}{\sqrt{n}} \leqslant \frac{\lambda\sigma}{2\sqrt{2}} \right\} = \mathcal{C}_2 \cap \mathcal{C}_3.$$

Hence $\mathbb{P}(\mathcal{A}) \geqslant 1 - \mathbb{P}(\mathcal{C}_2^c) - \mathbb{P}(\mathcal{C}_3^c)$. □

Remark 2: The bounds and probabilities presented in this analysis can be further refined through a more rigorous examination. Nevertheless, these bounds are sufficient for our primary objective, which is to demonstrate the pivotalness of λ in the Gaussian scenario.

Appendix B
Extension to Synthesis Model

Note that if we define $D = W^\mathsf{T}$,

$$\bar{z} = D^\mathsf{T}Dz^*.$$

Then the problem (3) can be rewritten as

$$\min_z \|z - \bar{y}\|_2 + \lambda\|z\|_1,$$

where $\bar{y} = D^{\mathsf{T}} y = \bar{z} + e$. This is identical to the square root lasso when the dictionary is the identity matrix. Therefore, it can be solved for $\bar{z}$ using all the tools available for the square root lasso, as previously studied in [15], [32]. We extend the previous results of the paper to obtain bounds for the synthesis model.

Theorem 4: Following the assumptions defined in Theorem 1, we have

$$\|\widehat{z} - z^*\|_\infty \leqslant \lambda(2+\eta)\epsilon + \rho\,\mu(D)\|z^*\|_\infty,$$

where $\rho = \|z^*\|_0$ and $\mu(D) = \max_{i \neq j} |D_i^{\mathsf{T}} D_j|$. Moreover, if

$$\min_{j \in [d]} |z_j^*| > 2\lambda(2+\eta)\epsilon + 2\rho\,\mu(D)\|z^*\|_\infty, \tag{11}$$

then the estimated support

$$\widehat{\mathcal{S}} = \{j \in [d] : |\widehat{z}_j| > \lambda(2+\eta)\epsilon + \rho\,\mu(D)\|z^*\|_\infty\}$$

recovers the true sparsity pattern correctly, i.e., $\widehat{\mathcal{S}} = \mathcal{S}$.

Proof of Theorem 4: Using Theorem 1 and the sparsity of z^*, we have

$$\begin{aligned}
\|\widehat{z} - z^*\|_\infty &\leqslant \|\widehat{z} - \bar{z}\|_\infty + \|\bar{z} - z^*\|_\infty \\
&\leqslant \lambda(2+\eta)\epsilon + \|(I - D^{\mathsf{T}} D) z^*\|_\infty \\
&= \lambda(2+\eta)\epsilon + \max_i |(I - D^{\mathsf{T}} D)_{\mathcal{S}} z^*_{\mathcal{S}}|_i \\
&\leqslant \lambda(2+\eta)\epsilon + \max_i \|(I - D^{\mathsf{T}} D)_{\mathcal{S}i}\|_1 \|z^*\|_\infty \\
&\leqslant \lambda(2+\eta)\epsilon + \rho\,\mu(D)\|z^*\|_\infty.
\end{aligned}$$

This proves the bound on $\|\widehat{z} - z^*\|_\infty$. Then, the support recovery property easily follows as in Theorem 1. □

Corollary 1: From the condition (11), we can derive a bound on the sparsity of the signal for correct support recovery:

$$\begin{aligned}
|z^*_{\min}| &> 2\lambda(2+\eta)\epsilon + 2\rho\,\mu(D)|z^*_{\max}|, \\
\rho &< \frac{|z^*_{\min}|}{2\mu(D)|z^*_{\max}|} - \frac{\lambda(2+\eta)\epsilon}{\mu(D)|z^*_{\max}|},
\end{aligned}$$

where $|z^*_{\min}|$ and $|z^*_{\max}|$ are the minimum and maximum absolute values of the entries of z^*, respectively. This bound closely resembles the optimality condition of the thresholding lasso algorithm [46].

Theorem 5: Following the assumptions defined in Theorem 2, then, we get

$$\|\widehat{z} - z^*\|_2 \leqslant (2+\eta)\epsilon + \mu(D)\sqrt{d}\|z^*\|_1.$$

Proof of Theorem 5: Using Theorem 2 and the sparsity of z^*, we have:

$$\begin{aligned}
\|\widehat{z} - z^*\|_2 &\leqslant \|\widehat{z} - \bar{z}\|_2 + \|\bar{z} - z^*\|_2 \\
&\leqslant (2+\eta)\epsilon + \|(I - D^{\mathsf{T}} D) z^*\|_2 \\
&\leqslant (2+\eta)\epsilon + \sqrt{\sum_{i=1}^{d} \mu(D)^2 \|z^*\|_1^2} \\
&\leqslant (2+\eta)\epsilon + \mu(D)\sqrt{d}\|z^*\|_1.
\end{aligned}$$

□

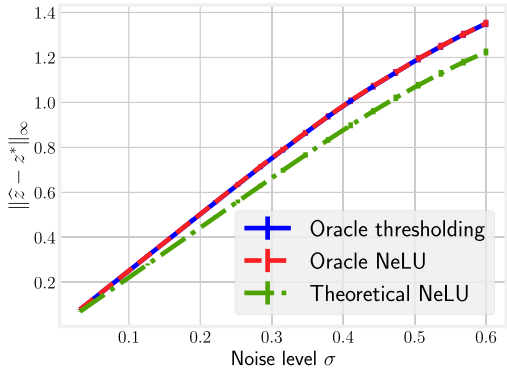


Fig. 6. The ℓ_∞ estimation error of each algorithm in the experiment presented in Fig. 2. The theoretical NeLU achieves a comparatively lower ℓ_∞ error because the oracle algorithms were optimized using the ℓ_2 error. The standard errors are less than 0.02 and are thus barely noticeable.

Appendix C
Illustration of Experiment in Section V-C

In the real-data experiment described in Section V-C, we follow the approach of Simon and Elad for deploying the Convolutional Sparse Coding (CSC) model [1]. The process of this experiment is illustrated in Fig. 5.

Appendix D
Details of the Analytical Transform Experiment - ℓ_∞ Error

In this section, we delve deeper into the performance of the algorithms from the main experiment, focusing on the ℓ_∞ estimation error. This analysis complements our earlier discussion centered around the ℓ_2 error, as shown in Fig. 2.

As observed in Fig. 6, the theoretical NeLU outperforms other methods, yielding a lower ℓ_∞ error. This performance can be attributed to the fact that the oracle algorithms, in the primary experiment, were fine-tuned using the ℓ_2 error criterion. A parallel behaviour is observed for the ℓ_2 error when optimizing with respect to the ℓ_∞ error.

Appendix E
Qualitative Analysis

This appendix provides a visual representation of the outcomes from the experiments detailed in Section V-C. The emphasis here is on a qualitative assessment of images processed using the networks specified in our study. Such an analysis aids comprehending the practical efficacy of the applied denoising methods.

Fig. 7 showcases a comprehensive comparison, juxtaposing the original images with their noisy counterparts and the subsequent denoised versions. This layout facilitates a direct visual evaluation of the noise reduction capabilities of the networks. For each image displayed, a specific patch has been chosen for detailed analysis. Specifically, for each image, the selection includes the original image marked with the patch location, the unaltered patch, the corresponding noisy patch (the original

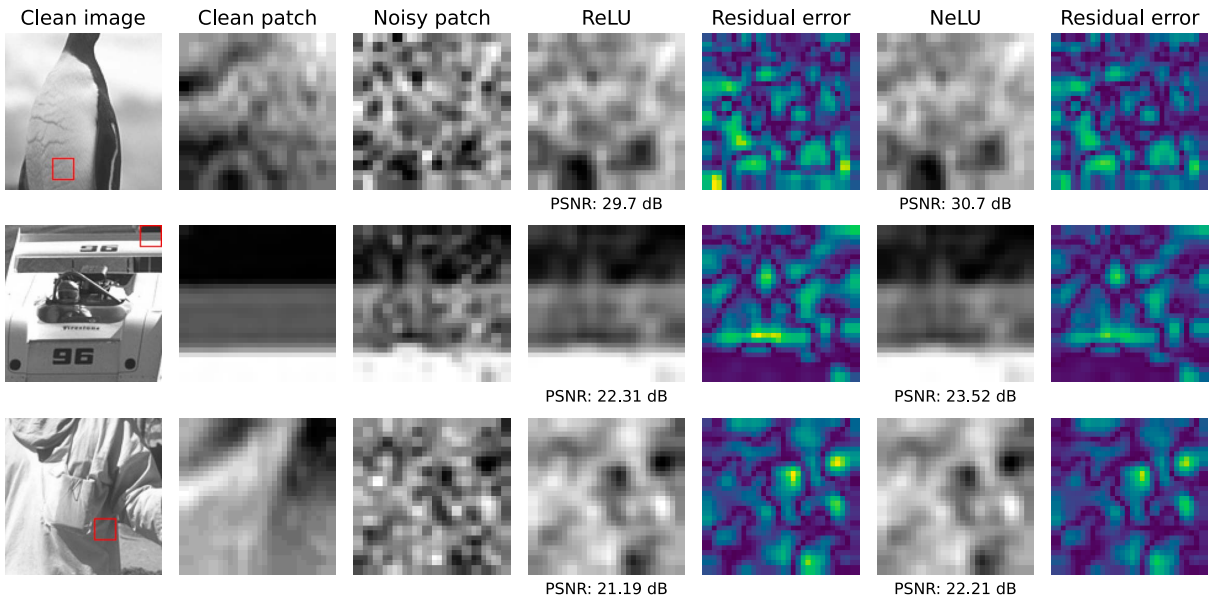


Fig. 7. Qualitative analysis of denoising networks from Section V-C. For each image, a selected patch is displayed in three states: the original, the version with Gaussian noise ($\sigma = 25$ for the first image, $\sigma = 50$ for the second and third), and the denoised output from each evaluated network. Additionally, heatmaps depict the residual errors, with warmer colors indicating larger absolute errors in those pixels. The Peak Signal-to-Noise Ratio (PSNR) values are also reported below each network's output to quantitatively assess the denoising performance.

with added Gaussian noise), and the denoised version processed by each network. Additionally, a heatmap of the residual error is presented, enabling a more precise and detailed comparison.

The residual error visualized in the figure highlights the enhancements our algorithm achieved, particularly in handling previously unseen noise levels. This visual representation serves not only as a validation of the algorithm's effectiveness but also offers insights into its potential limitations and areas for future improvement.

Appendix F
Empirical Validation of Theorem 2

In this appendix, we extend our investigation to empirically validate the theoretical estimation error bound introduced within our theoretical framework. To accomplish this, we replicate the experimental setup delineated in Section V-A. Herein, we evaluate the performance of our proposed algorithm (3) in juxtaposition with the theoretical threshold. Employing the iterative process specified in Algorithm 1, and considering a given transformation, we subject the algorithm to rigorous evaluation across a range of noise levels. For each noise level, the experiment encompasses 20 independent trials, incorporating diverse instances of signal and noise, with λ set to $\frac{\|e\|_\infty}{\|e\|_2}$.

The outcomes of these experiments are illustrated in Fig. 8. These experiments corroborate the assertion that the reconstruction error adheres to the bounds established in Theorem 2, thereby reinforcing the theoretical underpinnings of our methodology. Despite being conservative, the framework provides a reliable method for ensuring model robustness across different noise levels.

Further investigation into the exact optimal value of λ, considering factors such as the sparsity level of the signals, presents a promising avenue for future research. It is our hope that these insights will contribute to a deeper understanding of the interplay between theoretical analysis and empirical application in the domain of sparse autoencoders.

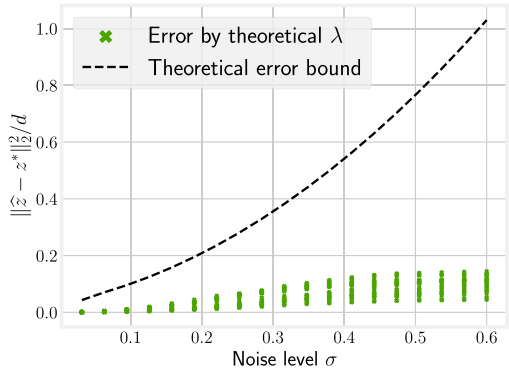


Fig. 8. Empirical validation of the theoretical estimation error bound. The figure showcases the MSE of the ℓ_2 estimation error as a function of the noise level σ. For each noise level, 20 independent trials were conducted with distinct combinations of signal and noise, with $\lambda = \frac{\|e\|_\infty}{\|e\|_2}$.

Acknowledgment

Yaniv Romano thanks the Career Advancement Fellowship, Technion, for providing research support.

References

[1] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[2] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou, "Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, no. 12, pp. 3371–3408, 2010.

[3] Y. Romano and M. Elad, "Boosting of image denoising algorithms," *SIAM J. Imag. Sci.*, vol. 8, no. 2, pp. 1187–1219, 2015.

[4] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Trans. Image Process.*, vol. 19, no. 11, pp. 2861–2873, Nov. 2010.

[5] V. Papyan, Y. Romano, J. Sulam, and M. Elad, "Theoretical foundations of deep learning via sparse representations: A multilayer sparse model and its connection to convolutional neural networks," *IEEE Signal Process. Mag.*, vol. 35, no. 4, pp. 72–89, Jul. 2018.

[6] S. Ravishankar and Y. Bresler, "Learning sparsifying transforms," *IEEE Trans. Signal Process.*, vol. 61, no. 5, pp. 1072–1086, Mar. 2013.

[7] S. Ravishankar and Y. Bresler, "Data-driven adaptation of a union of sparsifying transforms for blind compressed sensing MRI reconstruction," in *Wavelets and Sparsity*, vol. 9597, Bellingham, WA, USA: SPIE, 2015, pp. 247–256.

[8] J. Maggu, E. Chouzenoux, G. Chierchia, and A. Majumdar, "Convolutional transform learning," in *Proc. Neural Inf. Process.,* New York, NY, USA: Springer, 2018, pp. 162–174.

[9] B. Wen, S. Ravishankar, and Y. Bresler, "VIDOSAT: High-Dimensional Sparsifying Transform Learning for Online Video Denoising," *IEEE Trans. Image Process.*, vol. 28, no. 4, pp. 1691–1704, Apr. 2019.

[10] B. Wen, S. Ravishankar, and Y. Bresler, "FRIST—Flipping and rotation invariant sparsifying transform learning and applications," *Inverse Problems*, vol. 33, no. 7, 2017, Art. no. 074007.

[11] A. Gigie, A. A. Kumar, A. Majumdar, K. Kumar, and M. G. Chandra, "Joint coupled transform learning framework for multimodal image super-resolution," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2021, pp. 1640–1644.

[12] J. Maggu, A. Majumdar, E. Chouzenoux, and G. Chierchia, "Deep convolutional transform learning," in *Proc. Neural Inf. Process.,* New York, NY, USA: Springer, 2020, pp. 300–307.

[13] S. Mohan, Z. Kadkhodaie, E. P. Simoncelli, and C. Fernandez-Granda, "Robust and interpretable blind image denoising via bias-free convolutional neural networks," in *Proc. Int. Conf. Learn. Representations*, 2020.

[14] B. Lecouat, J. Ponce, and J. Mairal, "Fully trainable and interpretable non-local sparse models for image restoration," in *Proc. Eur. Conf. Comput. Vis.,* New York, NY, USA: Springer, 2020, pp. 238–254.

[15] A. Belloni, V. Chernozhukov, and L. Wang, "Square-root lasso: Pivotal recovery of sparse signals via conic programming," *Biometrika*, vol. 98, no. 4, pp. 791–806, 2011.

[16] A. Nakamura and M. Kobayashi, "Noise-level estimation from single color image using correlations between textures in RGB channels," 2019, *arXiv:1904.02566*.

[17] H. Lee, A. Battle, R. Raina, and A. Ng, "Efficient sparse coding algorithms," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 19, 2006, pp. 801–808.

[18] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993.

[19] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. Sci. Comput.*, vol. 43, no. 1, pp. 129–159, 2001.

[20] R. Tibshirani, "Regression shrinkage and selection via the Lasso," *J. Roy. Statist. Soc. Ser. B: Statist. Methodol.*, vol. 58, no. 1, pp. 267–288, 1996.

[21] X. Li et al., "On fast convergence of proximal algorithms for SQRT-Lasso optimization: Don't worry about its nonsmooth loss function," in *Proc. Uncertainty Artif. Intell.,* PMLR, 2020, pp. 49–59.

[22] S. Ravishankar, B. Wen, and Y. Bresler, "Online sparsifying transform learning—Part I: Algorithms," *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 4, pp. 625–636, Jun. 2015.

[23] A. Aberdam, J. Sulam, and M. Elad, "Multi-layer sparse coding: The holistic way," *SIAM J. Math. Data Sci.*, vol. 1, no. 1, pp. 46–77, 2019.

[24] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T.-W. Lee, and T. J. Sejnowski, "Dictionary learning algorithms for sparse representation," *Neural Comput.*, vol. 15, no. 2, pp. 349–396, 2003.

[25] K. Zhang, W. Zuo, and L. Zhang, "FFDNet: Toward a fast and flexible solution for CNN-based image denoising," *IEEE Trans. Image Process.*, vol. 27, no. 9, pp. 4608–4622, Sep. 2018.

[26] M. Zhussip, S. Soltanayev, and S. Y. Chun, "Training deep learning based image denoisers from undersampled measurements without ground truth and without image prior," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 10255–10264.

[27] M. Chen, Y. Chang, S. Cao, and L. Yan, "Learning blind denoising network for noisy image deblurring," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2020, pp. 2533–2537.

[28] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, 2017.

[29] A. Gnanasambandam and S. Chan, "One size fits all: Can we train one denoiser for all noise levels?" in *Proc. Int. Conf. Mach. Learn.,* PMLR, 2020, pp. 3576–3586.

[30] R. Tempo, "Robust estimation and filtering in the presence of bounded noise," *IEEE Trans. Autom. Control*, vol. 33, no. 9, pp. 864–867, Sep. 1988.

[31] Y. Romano, A. Aberdam, J. Sulam, and M. Elad, "Adversarial noise attacks of deep learning architectures: Stability analysis via sparse-modeled signals," *J. Math. Imag. Vis.*, vol. 62, pp. 313–327, 2020.

[32] M. Massias, Q. Bertrand, A. Gramfort, and J. Salmon, "Support recovery and sup-norm convergence rates for sparse pivotal estimation," in *Proc. Int. Conf. Artif. Intell. Statist.,* PMLR, 2020, pp. 2655–2665.

[33] K. Lounici, M. Pontil, A. Tsybakov, and S. Van De Geer, "Taking advantage of sparsity in multi-task learning," in *Proc. 22nd Conf. Learn. Theory*, 2009.

[34] F. A. Potra and S. J. Wright, "Interior-point methods," *J. Comput. Appl. Math.*, vol. 124, nos. 1–2, pp. 281–302, 2000.

[35] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.

[36] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proc. Int. Conf. Mach. Learn.*, 2010, pp. 399–406.

[37] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Process. Mag.*, vol. 38, no. 2, pp. 18–44, Mar. 2021.

[38] K. Zhang, L. V. Gool, and R. Timofte, "Deep unfolding network for image super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 3217–3226.

[39] J. Liu, Y. Sun, W. Gan, X. Xu, B. Wohlberg, and U. S. Kamilov, "SGD-Net: Efficient model-based deep learning with theoretical guarantees," *IEEE Trans. Comput. Imag.*, vol. 7, pp. 598–610, 2021.

[40] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proc. Int. Conf. Mach. Learn.,* PMLR, 2013, pp. 1139–1147.

[41] J. Sulam, A. Aberdam, A. Beck, and M. Elad, "On multi-layer basis pursuit, efficient algorithms and convolutional neural networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 8, pp. 1968–1980, Aug. 2020.

[42] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. Int. Conf. Learn. Representations*, 2018.

[43] D. Simon and M. Elad, "Rethinking the CSC model for natural images," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 2274–2284.

[44] K. Ma et al., "Waterloo exploration database: New challenges for image quality assessment models," *IEEE Trans. Image Process.*, vol. 26, no. 2, pp. 1004–1016, Feb. 2017.

[45] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, May 2011.

[46] V. Papyan, Y. Romano, and M. Elad, "Convolutional neural networks analyzed via convolutional sparse coding," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 2887–2938, 2017.

[47] J. Sulam, V. Papyan, Y. Romano, and M. Elad, "Multilayer convolutional sparse modeling: Pursuit and dictionary learning," *IEEE Trans. Signal Process.*, vol. 66, no. 15, pp. 4090–4104, Aug. 2018.

[48] C. Giraud, *Introduction to High-Dimensional Statistics*. Boca Raton, FL, USA: CRC Press, 2021.