sensors

MDPI

Article

Deep Unfolding of Iteratively Reweighted ADMM for Wireless RF Sensing

Udaya S. K. P. Miriya Thanthrige [1,*], Peter Jung [2,3] and Aydin Sezgin [1]

1 Institute of Digital Communication Systems, Ruhr University Bochum, 44801 Bochum, Germany; aydin.sezgin@rub.de
2 Institute of Communications and Information Theory, Technical University Berlin, 10587 Berlin, Germany; peter.jung@tu-berlin.de
3 Data Science in Earth Observation, Technical University of Munich, 82024 Munich, Germany
* Correspondence: udaya.miriyathanthrige@rub.de

Abstract: We address the detection of material defects, which are inside a layered material structure using compressive sensing-based multiple-input and multiple-output (MIMO) wireless radar. Here, strong clutter due to the reflection of the layered structure's surface often makes the detection of the defects challenging. Thus, sophisticated signal separation methods are required for improved defect detection. In many scenarios, the number of defects that we are interested in is limited, and the signaling response of the layered structure can be modeled as a low-rank structure. Therefore, we propose joint rank and sparsity minimization for defect detection. In particular, we propose a non-convex approach based on the iteratively reweighted nuclear and ℓ_1-norm (a double-reweighted approach) to obtain a higher accuracy compared to the conventional nuclear norm and ℓ_1-norm minimization. To this end, an iterative algorithm is designed to estimate the low-rank and sparse contributions. Further, we propose deep learning-based parameter tuning of the algorithm (i.e., algorithm unfolding) to improve the accuracy and the speed of convergence of the algorithm. Our numerical results show that the proposed approach outperforms the conventional approaches in terms of mean squared errors of the recovered low-rank and sparse components and the speed of convergence.

Keywords: algorithm unfolding; clutter suppression; defects detection; compressive sensing; reweighted norm

check for updates

Citation: Miriya Thanthrige, U.S.K.P.; Jung, P.; Sezgin, A. Deep Unfolding of Iteratively Reweighted ADMM for Wireless RF Sensing. *Sensors* **2022**, *22*, 3065. https://doi.org/10.3390/s22083065

Academic Editors: Thomas P Karnowski and Giacomo Oliveri

Received: 14 February 2022
Accepted: 13 April 2022
Published: 15 April 2022

1. Introduction

The electromagnetic (EM) waves-based remote sensing has many potential applications such as behind the wall object identification [1], multi-layer target detection [2], material characterization [3], defect detection [4–7], and many more. In EM and radio frequency (RF) waves-based detection of objects/defects which are behind or inside a layered structure, the EM waves that reflect from the object/defect are analyzed. Here, one major challenge is the presence of strong unwanted reflections, i.e., clutter [1,8]. In this context, the main source of the clutter is the reflection from the surface of the layered material structure.

The state-of-the-art clutter suppression methods such as background subtraction (BS), time-gating, and subspace projection (SP) [9] are not able to suppress the clutter in the context of object/defect detection. This is due to the fact that in BS, it requires the reference data of the scene, and this reference data is not available most of the time. Moreover, in the SP, prior knowledge is required to determine the perfect threshold for clutter removal. On the other hand, in time-gating, the time window in which clutter resides needs to be determined for successful clutter removal. However, this time window cannot be determined exactly. Clutter suppression becomes even more challenging if objects and clutter are closely located. This occurs regularly in the detection of defects which are inside a layered structure. Then, due to the small delay spread, the signaling responses of defects and clutter superimpose each other. In order to overcome these challenges, advanced signal processing methods are required for clutter suppression [1,8,10].

In many scenarios, the responses of the material defects are weak and, thus, difficult to detect. Even if there is no clutter, due to very low signal amplitude, it may be difficult to detect material defects in the presence of noise. In this context, the weak signal detection in the presence of noise has drawn attention in the defect detection research field. Therefore, we briefly discuss the weak signal detection in the following. Stochastic resonance has been widely used in weak signal detection [11–13]. In [11], to improve upon the weak signal detection by stochastic resonance, the relationship between the current and the previous value of the state variable of the system has been utilized. It is worth noticing that weak signal detection plays an important role in other applications such as health monitoring. Similar to defect detection, health monitoring aims to detect weak signals in the presence of strong noise. In [14], a comparative study of well-known adaptive mode decomposition approaches that are used for the aforementioned task is reviewed. Here, the advantages, limitations, and the performance comparison of adaptive mode decomposition approaches, namely empirical mode decomposition, Hilbert vibration decomposition, and variational mode decomposition, have been given. Other than signal detection, the extraction of features of the detected signal is important in many applications as these features are used for classification and clustering. In this context, it is important to select the most important features as the accuracy and speed of the classification depend on the features that are used. The impact of the feature selection for electromyographic signal decomposition is studied in [15]. Moreover, in this study, various feature extraction methods are compared, and a guide to select the most important features that improve the signal decomposition is provided [15]. As we discussed above, weak signal detection in the presence of disturbances like noise or clutter is challenging, therefore, advanced signal procession methods are required. Next, we discuss clutter suppression in more detail.

In many scenarios, the number of defects is limited. Therefore, the signaling response of the defect is sparse in nature. By exploring this, compressive sensing (CS) [16] based approaches have shown promising results in object/defect detection with clutter [1,8]. In addition, the CS-based approaches do not require a full measurement data set, which results in fast data acquisition and less sensitivity to sensor failure, wireless interference, and jamming. In CS-based approaches, it is considered that the clutter resides in a low-rank subspace and the response of the objects is sparse [1,8].

To this end, we present a general data acquisition model where the received data vector $\boldsymbol{y} \in \mathbb{C}^K$ is modeled as a combination of a low-rank matrix $\boldsymbol{L} \in \mathbb{C}^{M \times N}$ and a sparse matrix $\boldsymbol{S} \in \mathbb{C}^{M \times N}$ with $M \leq N$:

$$\boldsymbol{y} = \boldsymbol{A}_l \mathrm{vec}(\boldsymbol{L}) + \boldsymbol{A}_s \mathrm{vec}(\boldsymbol{S}) + \boldsymbol{n}, \tag{1}$$

in which $\boldsymbol{A}_l,\ \boldsymbol{A}_s, \in \mathbb{C}^{K \times MN}$ with $K \ll MN, \boldsymbol{n}\ \in \mathbb{C}^K$ are compression operators/measurement matrices and measurement noise, respectively. Here, the compression ratio is defined as K/MN. Further, $\mathrm{vec}(\cdot)$ denotes the vectorization operator, which converts a matrix to a vector by stacking the columns of the matrix. Given the received data vector $\boldsymbol{y}$, our aim is to estimate the signals of interest, $\boldsymbol{L}$ and $\boldsymbol{S}$, using a small number of linear measurements by minimizing the rank and sparsity as

$$\begin{aligned} \{\hat{\boldsymbol{L}}, \hat{\boldsymbol{S}}\} &= \arg\min_{\boldsymbol{L},\, \boldsymbol{S}} \lambda_l \ \mathrm{rank}(\boldsymbol{L}) + \lambda_s \|\boldsymbol{S}\|_0, \\ &\text{s.t. } \|\boldsymbol{y} - \boldsymbol{A}_s \mathrm{vec}(\boldsymbol{S}) - \boldsymbol{A}_l \mathrm{vec}(\boldsymbol{L})\|_2^2 \ \leq \ \epsilon, \end{aligned} \tag{2}$$

where λ_l, λ_s are regularization parameters and ϵ is a small positive constant (noise bound). Here, $\|\cdot\|_0$ is the ℓ_0-norm, i.e., sparsity (the number of non-zero components). Note that the problem given in (2) is also known as robust principal component analysis (RPCA) [17]. The RPCA problem has different types as follows: (a) standard/classical RPCA in which both $\boldsymbol{A}_l$ and $\boldsymbol{A}_s$ in (1) are identity matrices [17], (b) the matrices $\boldsymbol{A}_l = \boldsymbol{A}_s = \boldsymbol{A}$ and $\boldsymbol{A}$ is a selection operator which select a random subset of size K from MN entries [18], (c) both $\boldsymbol{A}_l$ and $\boldsymbol{A}_s$ are $K \times MN$ matrices which map the vector space $\mathbb{C}^{MN}$ to the vector space $\mathbb{C}^K$ [18].

The problem given in (2) is an NP-hard problem and, thus, difficult to solve. To this end, convex relaxations of sparsity and rank in terms of ℓ_1-norm of a matrix (absolute sum of elements) and nuclear norm of a matrix (sum of singular values) are utilized, respectively [19–21]. However, enjoying a rigorous analysis, the convex relaxations of sparsity and rank cause disadvantages in many applications. In addition to that, in many applications, the important properties of the signal are preserved by the large coefficients/singular values of the signal [22]. However, the ℓ_1-norm/nuclear norm minimization algorithms shrink all the coefficients/singular values with the same threshold. Thus, to avoid this weakness, we should shrink less the larger coefficients/singular values. To address aforementioned drawbacks, non-convex approaches such as reweighted nuclear norm and reweighted ℓ_1-norm minimization have been considered [22–25]. These non-convex approaches have shown better performance over the convex relaxations by providing tighter characterizations of rank and sparsity, yet their behavior and convergence have not been fully studied [26].

Generally, RPCA problems are numerically solved by means of iterative algorithms based on the alternating direction method of multipliers (ADMM) [17,27,28] or accelerated proximal gradient (APG) [29,30]. In iterative algorithms, the accuracy of the recovered signal component and the convergence rate depends on the proper selection of parameters (e.g., regularization/thresholding/denoising parameters). Generally, parameters are chosen by handcrafting, and it is a time-consuming task. In this context, machine learning-based parameter tuning using training data has shown promising results in many applications such as sparse vector recovery [31–33] and image processing [34]. For instance, as shown in [31], the unfolded iterative soft-thresholding algorithm (LISTA) converges twenty times faster than the conventional iterative soft-thresholding algorithm (ISTA). This approach is known as algorithm unrolling/unfolding, and an overview can be found in [35].

In this work, we formulate the detection of material defects as a RPCA problem. This RPCA problem is solved based on the reweighted nuclear norm and reweighted ℓ_1-norm minimization. However, most of the time, RPCA problems are solved by using the convex relaxation or with the single reweighting, i.e., either reweighted ℓ_1-norm or reweighted nuclear norm [22,30,36,37]. Next, our objective is to jointly estimate the low-rank matrix and the sparse matrix from few compressive measurements. It is worth noticing that most of the work in the literature focuses on the standard RPCA problem, where $\boldsymbol{A}_l$ and $\boldsymbol{A}_s$ are identity matrices [22,36]. To the best of our knowledge, the full doubly reweighted (joint reweighted nuclear norm and reweighted ℓ_1-norm) approach has not yet been studied comprehensively in the literature for the compressive case. Then, we propose an iterative algorithm for (locally) minimizing the objective, i.e., reweighted nuclear norm and reweighted ℓ_1-norm, which is based on the alternating direction method of multipliers (ADMM) [38,39]. Further, we propose deep learning-based parameter tuning to improve the accuracies of the recovered low-rank and sparse components and the convergence rate of the ADMM-based iterative algorithm.

In addition to the EM-based defect detection, there are many applications where the data generated by the application can be modeled as a combination of low-rank plus sparse contributions. For instance, in video surveillance, the static background results in a low-rank contribution, and moving objects result in a sparse contribution [40]. Further, in human face recognition from a corrupted face image, the human face can be approximated as a low-rank structure while self-shadowing and specularities are modeled as sparse contributions [40,41]. Therefore, RPCA can be applied to the aforementioned applications and other applications as long as the data/measurements are combinations of low-rank and sparse contributions. It is worth noticing that our proposed full doubly reweighted (joint reweighted nuclear norm and reweighted ℓ_1-norm) approach with deep learning-based parameter tuning for RPCA is not limited to EM-based defect detection and can be applied to other applications that are solved using RPCA.

In the context of the algorithm unfolding for the RPCA, the convolutional robust principal component analysis (CORONA) [30,37] are the closest studies to our work. There are fundamental methodological differences between our work and [30,37]: (a) Both [30,37] considered the standard convex relaxation ($\ell_{1,2}$-norm and nuclear norm) to solve the RPCA problem, while we propose the reweighted ℓ_1-norm and reweighted nuclear norm. (b) In this work, the RPCA problem is solved by an iterative algorithm based on ADMM, while the iterative algorithm in [30,37] is based on fast ISTA (FISTA). The motivation to propose ADMM over ISTA/FISTA for RPCA is as follows. As shown in [17,27] for RPCA, the ADMM-based approach is able to achieve the desired solution with a good recovery error with few iterations for a wide range of applications compared to APG-based approaches like ISTA/FISTA. Further, the performances of the APG-based approaches are heavily dependent on the good continuation schemes [17]. This condition may not be satisfied for a wide range of applications. (c) Different from [30,37], our focus is on defect detection based on the stepped-frequency continuous wave (SFCW) radar, while [30,37] focus on ultrasound imaging application. Moreover, experimental measurement data of [30,37] have considered that $\boldsymbol{A}_l = \boldsymbol{A}_s = \boldsymbol{A}$ in (1) is an identity matrix, while we consider both scenarios where $\boldsymbol{A}$ is an identity matrix and it is a compression operator. Further, for the SFCW radar application, we consider that $\boldsymbol{A}_l \neq \boldsymbol{A}_s$. Further, we have studied the performance of our approach with a generic real-valued Gaussian model for different compression ratios.

The CORONA focuses on ultrasound imaging applications where sparse matrix has row-sparse structure. Thus, there is a strong relationship between measurement to measurement, and there is a common sparsity structure. Therefore, $\ell_{1,2}$-norm minimization is more suitable than ℓ_1-norm minimization to estimate sparse matrix $\boldsymbol{S}$. Further, the CORONA is based on a convolutional deep neural network to learn spatial invariance features of data, which is more suitable for ultrasound imaging applications than a dense deep neural network (DNN). However, we assume that there is no strong relationship of a data element to its neighboring elements, nor is there a specific sparsity structure. Thus, we consider a dense DNN in this work. It is straightforward to modify our ADMM approach with convolutional DNN and the $\ell_{1,2}$-norm minimization. In CORONA [30], customized complex-valued convolution layers and singular value decomposition operations are utilized. In our work, we have implemented a dense DNN which supports complex-valued data and singular value decomposition (SVD) operation. The contributions of this work are summarized as follows:

1.1. Contribution

- We propose a generic approach based on the non-convex fully double-reweighted approach, i.e., both reweighted ℓ_1-norm and reweighted nuclear norm simultaneously to solve the RPCA problem. To this end, we propose an iterative algorithm based on ADMM to estimate the low-rank and sparse components jointly.
- In contrast to standard/classical RPCA, we consider the compressive sensing data acquisition model, which reflects more on the practical problem at hand. Next, to improve the accuracy and convergence speed of the ADMM-based iterative algorithm, we propose a deep neural network (DNN) to tune the parameters of the iterative algorithm (i.e., algorithm unfolding/unrolling) from training data.
- We intensively evaluate our proposed approach for a generic Gaussian data acquisition model with $\boldsymbol{A}_l = \boldsymbol{A}_s = \boldsymbol{A}$. In addition to that, the defect detection by SFCW radar from compressive measurements with $\boldsymbol{A}_l \neq \boldsymbol{A}_s$ is considered. To compare our approach, we consider the standard convex approach (i.e., nuclear norm and ℓ_1-norm minimization) and the untrained ADMM-based iterative algorithm for different compression ratios. In both the generic Gaussian data acquisition model and SFCW-based defect detection, our numerical results show that the proposed approach outperforms the conventional approaches in terms of mean squared errors of the recovered low-rank and sparse components and the speed of convergence.

- In the context of algorithm unrolling for RPCA, we compare our approach with the approach given in [30] (CORONA). It turns out that our proposed approach shows similar performance as CORONA for experimental ultrasound imaging data used in [30], and our approach outperforms CORONA for generic Gaussian data. It is worth noticing that there is a row-sparse nature of the experimental ultrasound data. That is the reason CORONA uses $\ell_{1,2}$-norm minimization to estimate sparse matrix $\boldsymbol{S}$. Our approach is generic, yet our approach is able to achieve similar results as CORONA by learning. This shows the applicability of our approach to different types of use cases and data (defect detection, ultrasound imaging, generic Gaussian data).
- We numerically analyze the robustness of our proposed approach for the generic Gaussian data acquisition model. Here, we consider the deviation in the measurement matrices ($\boldsymbol{A}_l$, $\boldsymbol{A}_s$) and testing signal-to-noise ratio (SNR) uncertainty. It was observed that the proposed approach is robust for a small deviation in the measurement matrices. Further, it was observed that training with the SNR like 5 dB is favorable when SNR of the testing data is unknown.

The remainder of the paper is organized as follows. We introduce the SFCW radar-based defect detection and the low-rank plus sparse recovery with reweighting in Section 2. In Section 3, we discuss the DNN-based low-rank plus sparse recovery algorithm unfolding. In Section 4, we provide an evaluation of the proposed DNN-based low-rank plus sparse recovery algorithm unfolding approaches and provide interesting insights. Section 5 concludes the paper.

1.2. Notation

In this paper, the following notation is used. A vector is denoted in boldface lower-case letter, while the matrices are denoted in boldface upper-case. The ℓ_0-norm (the number of nonzero components), ℓ_1-norm (absolute sum of elements) of a matrix/vector, and nuclear norm of a matrix (sum of singular values) are denoted by $\|\cdot\|_0$, $\|\cdot\|_1$, and $\|\cdot\|_*$, respectively. Further, the Frobenius norm of a matrix and ℓ_2-norm is given by $\|\cdot\|_F$ and $\|\cdot\|_2$, respectively. The Hermitian and transpose of the matrix $\boldsymbol{A}$ are represented by $\boldsymbol{A}^H$ and $\boldsymbol{A}^T$, respectively. In addition, the Moore–Penrose pseudo inverse is denoted by $(\cdot)^\dagger$. A matrix of size $M \times N$ with all elements equal to zero and one are denoted by $\mathbf{0}_{M,N}$ and $\mathbf{1}_{M,N}$, respectively. Moreover, a vector of size M with all elements equal to zero and one are denoted by $\mathbf{0}_M$ and $\mathbf{1}_M$, respectively. In addition, identity matrix is denoted by $\boldsymbol{I}$. The main variable list and abbreviations that are used in this manuscript are listed at the end of the manuscript.

2. System Model

First, we briefly present the system model of the mono-static SFCW radar-based defect detection. Next, we discuss the ADMM -based iterative algorithm for the low-rank plus sparse recovery.

2.1. SFCW Radar Based Defect Detection

We consider an SFCW radar with M transceivers which are placed in parallel to the single-layered material structure while maintaining an equal distance between transceivers, as shown in Figure 1. In SFCW radar, each transceiver transmits a stepped-frequency signal containing N frequencies which are equally spaced over the bandwidth of B Hz. To this end, the received signal corresponding to all M transceivers and N frequencies $\boldsymbol{Y} \in \mathbb{C}^{M \times N}$ are given by

$$\boldsymbol{Y} = \boldsymbol{Y}^l + \boldsymbol{Y}^d + \boldsymbol{Z}. \tag{3}$$

Note that $\boldsymbol{Y}$ consists of two main components, the reflection of the layered material structure ($\boldsymbol{Y}^l$) and the reflection of the defects ($\boldsymbol{Y}^d$). Here, $\boldsymbol{Z}$ is the additive Gaussian noise matrix. Next, we discuss in detail the modeling of the received signal of the defects by using the propagation time delay. To this end, the scene shown in Figure 1 is virtually partitioned into a rectangular grid of size Q. Suppose that the round-travel time of the signal from the m-th antenna location to the p-th defect and back is given by $\tau_{m,p}$. Then, the received signal of the defects $y^d_{m,n}$ in m-th transceiver corresponding to n-th frequency band f_n is given by [1]

$$y^d_{m,n} = \sum_{p=1}^{P} \alpha_p \exp(-j2\pi f_n \tau_{m,p}). \tag{4}$$

Here, $j = \sqrt{-1}$, $\alpha_p \in \mathbb{C}$ is the complex reflectivity coefficient of the p-th defect, and P is the total number of defects. To this end, $\text{vec}(\boldsymbol{Y}^d) \in \mathbb{C}^{MN\times 1}$ is given by

$$\text{vec}(\boldsymbol{Y}^d) = \boldsymbol{D}\boldsymbol{s}, \tag{5}$$

where $\boldsymbol{s} \in \mathbb{C}^{Q\times 1}$ contains all the α_p values of the defects. Since there are P defects, the vector $\boldsymbol{s}$ only contains P non-zero entries. The matrix $\boldsymbol{D}$ is given by $[(\boldsymbol{D}_1)^T, \dots, (\boldsymbol{D}_m)^T, \dots, (\boldsymbol{D}_M)^T]^T \in \mathbb{C}^{MN\times Q}$. Note that the (n,q)-th element of the matrix $\boldsymbol{D}_m \in \mathbb{C}^{N\times Q}$ is given by $\exp(-j2\pi f_n \tau_{m,q})$, where $\tau_{m,q}$ is the propagation time delay between the m-th antenna to the q-th grid location. We assume that the propagation time delays $\tau_{m,p}$ of the defects are exactly matched with the propagation time delays of the grid locations. If this condition does not satisfy, it is known as grid mismatch. The grid mismatch degrades the performance of the sparse signal estimation [42]. There are several approaches proposed to rectify this problem, e.g., Bayesian learning-based approach [43], iterative dictionary updates [3], and many more. Similar to the received signal of the defects $y^d_{m,n}$, the received signal of the layered material structure $y^l_{m,n}$ in m-th transceiver corresponding to n-th frequency band f_n is given by [1]:

$$y^l_{m,n} = \sum_{\bar{p}=1}^{\bar{P}+1} \alpha_l a_{\bar{p}} \exp(-j2\pi f_n \tau_{m,\bar{p}}). \tag{6}$$

Here, $\alpha_l \in \mathbb{C}$ is the complex reflectivity of the layered material structure. $a_{\bar{p}}$ and $\tau_{m,\bar{p}}$ are the propagation loss and the propagation delay of the $\bar{p}$-th return of the layered material structure. The number of internal reflections within the layered material is given by $\bar{P}$.

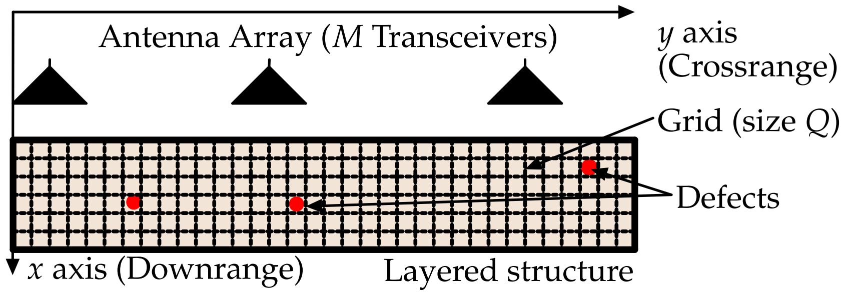


Figure 1. Getting the measurements of a single-layered material structure using an SFCW radar with M transceivers. The received signal consists of two main components, the reflection of the layered material structure ($\boldsymbol{Y}^l$) and the reflection of the defects ($\boldsymbol{Y}^d$), where $\boldsymbol{Y}^l$ is the main clutter source. Here, defects are shown as red circles.

2.2. Compressed Sensing (CS) Approach

In the compressed sensing (CS) setup, only a subset of antennas/frequencies are available or selected. Now, the reduced data vector $\boldsymbol{y}_{cs} \in \mathbb{C}^{K\times 1}$ of size K ($\ll MN$) is given by

$$\boldsymbol{y}_{cs} = \boldsymbol{\Phi}(\text{vec}(\boldsymbol{Y})) = \boldsymbol{\Phi}\text{vec}(\boldsymbol{Y}^l) + \boldsymbol{\Phi}\boldsymbol{D}\boldsymbol{s} + \boldsymbol{\Phi}\text{vec}(\boldsymbol{Z}), \tag{7}$$

where $\boldsymbol{\Phi} \in \mathbb{R}^{K\times MN}$ is the selection matrix. The matrix $\boldsymbol{\Phi}$ has a single non-zero element of value one in each row to indicate the selected frequency of a particular antenna if that antenna is selected. Here, our main objective is to recover $\boldsymbol{Y}^l$ and $\boldsymbol{s}$ from the reduced data vector $\boldsymbol{y}_{cs}$ using the low-rank plus sparse recovery approach as detailed below.

2.3. Low-Rank Plus Sparse Recovery Algorithm

From now on we consider the general data acquisition model given in (1) in Section 1, i.e., $\boldsymbol{y} = \boldsymbol{A}_l \text{vec}(\boldsymbol{L}) + \boldsymbol{A}_s \text{vec}(\boldsymbol{S}) + \boldsymbol{n}$. Note that the SFCW radar model given in (7) is mapped to the generic measurement model by considering $\boldsymbol{A}_s = \boldsymbol{\Phi}\boldsymbol{D}$, $\boldsymbol{A}_l = \boldsymbol{\Phi}$, $\boldsymbol{Y}^l = \boldsymbol{L}$, $\boldsymbol{s} = \text{vec}(\boldsymbol{S})$, and $\boldsymbol{y}_{cs} = \boldsymbol{y}$, respectively. Our objective is to recover the low-rank matrix $\boldsymbol{L}$ and the sparse matrix $\boldsymbol{S}$ from the compressive measurements $\boldsymbol{y}$. Thus, the estimation of $\boldsymbol{L}$ and $\boldsymbol{S}$ from $\boldsymbol{y}$ is done by minimizing rank and the sparsity (ℓ_0-norm). Note that rank and ℓ_0-norm minimization problems are usually NP-hard. Thus, one may use instead convex relaxations based on the nuclear norm of a matrix and ℓ_1-norm of a matrix as follows:

$$\begin{aligned} \{\hat{\boldsymbol{L}}, \hat{\boldsymbol{S}}\} = \underset{\boldsymbol{L},\, \boldsymbol{S}}{\arg\min}\, \lambda_l \|\boldsymbol{L}\|_* + \lambda_s \|\boldsymbol{S}\|_1, \\ \text{s.t. } \|\boldsymbol{y} - \boldsymbol{A}_s \text{vec}(\boldsymbol{S}) - \boldsymbol{A}_l \text{vec}(\boldsymbol{L})\|_2^2 \leq \epsilon. \end{aligned} \tag{8}$$

The resulting convex problems, i.e., ℓ_1-norm and nuclear norm minimization, are well studied in the literature, and there are several non-convex approaches to improve over the standard convex relaxation. One well-known approach is iterative reweighting of the ℓ_1-norm [23,32,44] and nuclear norm [22,45–47]. Alternating direction method of multipliers (ADMM) is used to solve the problem given in (8). First, we formulate the problem given in (8) based on ADMM approach, and then we introduce the non-convex double-reweighted approach, i.e., both reweighted ℓ_1-norm and reweighted nuclear norm simultaneously. Let the signal component value of $\boldsymbol{S}$ and $\boldsymbol{L}$ at the t-th iteration be denoted as $(\cdot)^t$. Now, based on the ADMM, $\boldsymbol{S}$ and $\boldsymbol{L}$ are estimated by

$$\boldsymbol{L}^{t+1} = \underset{\boldsymbol{L}}{\arg\min}\ \lambda_l \|\boldsymbol{L}\|_\star + \frac{\rho}{2} \left\| \boldsymbol{A}_s \text{vec}(\boldsymbol{S}^t) + \boldsymbol{A}_l \text{vec}(\boldsymbol{L}) - \boldsymbol{y} + \frac{1}{\rho} \boldsymbol{u}^t \right\|_2^2, \tag{9}$$

$$\boldsymbol{S}^{t+1} = \underset{\boldsymbol{S}}{\arg\min}\ \lambda_s \|\boldsymbol{S}\|_1 + \frac{\rho}{2} \left\| \boldsymbol{A}_s \text{vec}(\boldsymbol{S}) + \boldsymbol{A}_l \text{vec}\left(\boldsymbol{L}^{t+1}\right) - \boldsymbol{y} + \frac{1}{\rho} \boldsymbol{u}^t \right\|_2^2, \tag{10}$$

$$\boldsymbol{u}^{t+1} = \boldsymbol{u}^t + \rho\left(\boldsymbol{A}_s \text{vec}\left(\boldsymbol{S}^{t+1}\right) + \boldsymbol{A}_l \text{vec}\left(\boldsymbol{L}^{t+1}\right) - \boldsymbol{y}\right). \tag{11}$$

Here, $\boldsymbol{u}$, $\rho > 0$ are auxiliary variables and a penalty factor. Let $\boldsymbol{\sigma}(\boldsymbol{L}) = [\sigma_1, \ldots \sigma_m, \ldots, \sigma_M] \in \mathbb{R}^M$ be the singular values of $\boldsymbol{L}$. The nuclear norm of $\boldsymbol{L}$ is given by $\|\boldsymbol{L}\|_\star = \|\boldsymbol{\sigma}(\boldsymbol{L})\|_1$. Now, we are going to introduce the weighted ℓ_1-norm and weighted nuclear norm to the sub-problems given in (9) and (10) as follows:

$$\boldsymbol{L}^{t+1} = \underset{\boldsymbol{L}}{\arg\min}\ \lambda_l \left\|\boldsymbol{w}_l^t \odot \boldsymbol{\sigma}(\boldsymbol{L})\right\|_1 + \frac{\rho}{2} \left\| \boldsymbol{A}_s \text{vec}(\boldsymbol{S}^t) + \boldsymbol{A}_l \text{vec}(\boldsymbol{L}) - \boldsymbol{y} + \frac{1}{\rho} \boldsymbol{u}^t \right\|_2^2, \tag{12}$$

$$\boldsymbol{S}^{t+1} = \underset{\boldsymbol{S}}{\arg\min}\ \lambda_s \left\|\boldsymbol{w}_s^t \odot \boldsymbol{S}\right\|_1 + \frac{\rho}{2} \left\| \boldsymbol{A}_s \text{vec}(\boldsymbol{S}) + \boldsymbol{A}_l \text{vec}\left(\boldsymbol{L}^{t+1}\right) - \boldsymbol{y} + \frac{1}{\rho} \boldsymbol{u}^t \right\|_2^2. \tag{13}$$

The operator $\odot$ denotes element-wise multiplication. Here, $\boldsymbol{w}_l^t \in \mathbb{R}^M$ and $\boldsymbol{w}_s^t \in \mathbb{R}^{MN}$ are non-negative weight vectors in $t+1$-th iteration. To this end, $\boldsymbol{w}_l^t$ and $\boldsymbol{w}_s^t$ are calculated based on the previous estimation of the $\boldsymbol{L}$ and $\boldsymbol{S}$, i.e., $\boldsymbol{L}^t$ and $\boldsymbol{S}^t$.

$$\boldsymbol{w}_l^t = g_l(\boldsymbol{\sigma}(\boldsymbol{L}^t)) \text{ and } \boldsymbol{w}_s^t = g_s(|\boldsymbol{S}^t|). \tag{14}$$

Here, $g_l(\cdot)$ and $g_s(\cdot)$ are decay functions, applied component-wise, which are used to calculate the weights. There are several decay functions proposed in the literature, and an overview of the nuclear norm is given in [47]. In this work, motivated by [32], we consider element-wise (adaptive) soft-thresholding as the proximal operator of the weighted ℓ_1-norm. In addition, inspired by [48], element-wise (adaptive) singular value soft-thresholding (i.e., element-wise soft-thresholding on the singular values of a matrix) is used as a proximal operator of the weighted nuclear norm. Now, $\boldsymbol{L}^{t+1}$ and $\boldsymbol{S}^{t+1}$ are given by

$$\boldsymbol{L}^{t+1} = \mathrm{SVT}_{\lambda_{LT}^t}\left(\boldsymbol{A}_l^*\left(\boldsymbol{y} - \boldsymbol{A}_s \mathrm{vec}\left(\boldsymbol{S}^t\right) + \frac{\boldsymbol{u}^t}{\rho}\right)\right), \tag{15}$$

$$\boldsymbol{S}^{t+1} = \mathrm{ST}_{\lambda_{ST}^t}\left(\boldsymbol{A}_s^*\left(\boldsymbol{y} - \boldsymbol{A}_l \mathrm{vec}\left(\boldsymbol{L}^{t+1}\right) + \frac{\boldsymbol{u}^t}{\rho}\right)\right), \tag{16}$$

where $\mathrm{SVT}(\cdot)$ and $\mathrm{ST}(\cdot)$ are the element-wise singular value soft-thresholding and element-wise soft-thresholding operators [32,48], respectively. Note that $(\cdot)^*$ is a linear operator which back projects the vector into the target matrix subspace. There are two options for $(\cdot)^*$: (a) Hermitian transpose $(\cdot)^H$, as done in [32], or (b) Moore–Penrose pseudo inverse $(\cdot)^\dagger$, as done in [1]. Next, we are going to discuss the element-wise (adaptive) soft-thresholding and the element-wise (adaptive) singular value soft-thresholding.

2.4. Element-Wise Soft-Thresholding and Singular Value Soft-Thresholding

In (16), $\boldsymbol{\lambda}_{ST}^t = [\lambda_{ST}^{1,1}, \ldots, \lambda_{ST}^{m,n}, \ldots, \lambda_{ST}^{M,N}]$ contains the element-wise thresholds for $\boldsymbol{S}$ for the $t+1$-th iteration. These thresholds are derived based on the previous estimate $\boldsymbol{S}$, i.e., $\boldsymbol{S}^t$,

$$\lambda_{ST}^{m,n} = \lambda_S \, g_s(|s_{m,n}^t|). \tag{17}$$

Here, λ_S is a positive constant (soft-thresholding parameter), and $s_{m,n}^t$ is the m-th row and n-th column element of the t-th estimation of $\boldsymbol{S}$, i.e., $\boldsymbol{S}^t$. The same concept is also applied to the singular value soft-thresholding which is used in (15), as discussed next. In this work, we consider the same decay function for both sparsity and rank, i.e., $g_s(\cdot) = g_l(\cdot)$. In (15), $\boldsymbol{\lambda}_{LT}^t = [\lambda_{LT}^1, \ldots, \lambda_{LT}^m, \ldots \lambda_{LT}^M]$ contains the different thresholds calculated from the singular values of the previous estimate of $\boldsymbol{L}$ as given below:

$$\lambda_{LT}^m = \lambda_L \, g_l(\sigma_m^t). \tag{18}$$

Here, σ_m^t is the m-th singular value of $\boldsymbol{L}^t$, and λ_L is a positive constant (singular-value-soft-thresholding parameter). For completeness, definitions of the element-wise soft-thresholding and singular value soft-thresholding are given in Appendix A.1. Our objective is to tune the parameters λ_S in (17) and λ_L in (18) by using a deep neural network, as discussed next.

3. Unfolding ADMM-Based Low-Rank Plus Sparse Recovery Algorithm

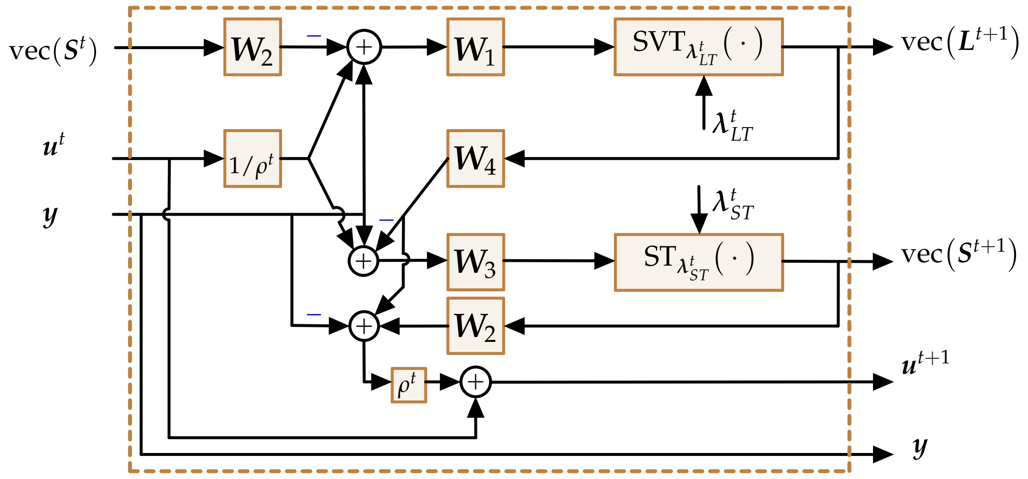
In this section, we are going to discuss the ADMM algorithm unfolding using a dense DNN. The iterative algorithm given in Algorithm 1 utilizes the ADMM steps given in (15), (16), and (11), and previous estimates are used in the next iteration. Thus, this kind of iterative algorithm can be considered as a recurrent neural network. The t-th iteration of the iterative Algorithm 1 is modeled as the t-th layer of the deep neural network as shown in Figure 2. Each matrix multiplication given in the ADMM steps (15), (16), and (11) are implemented as linear layers without biases. Here, our main objective is to learn the per iteration weights of the network and thresholding parameters λ_S and λ_L given in (17) and (18) from training data. To this end, the t-th layer of the neural network is represented by the following equations:

$$\boldsymbol{L}^{t+1} = \mathrm{SVT}_{\lambda_{LT}^t}\left(\boldsymbol{W}_1^t\left(\boldsymbol{y} - \boldsymbol{W}_2^t \mathrm{vec}\left(\boldsymbol{S}^t\right) + \frac{\boldsymbol{u}^t}{\rho^t}\right)\right), \tag{19}$$

$$\boldsymbol{S}^{t+1} = \mathrm{ST}_{\lambda_{ST}^t}\left(\boldsymbol{W}_3^t\left(\boldsymbol{y} - \boldsymbol{W}_4^t \mathrm{vec}\left(\boldsymbol{L}^{t+1}\right) + \frac{\boldsymbol{u}^t}{\rho^t}\right)\right), \tag{20}$$

$$\boldsymbol{u}^{t+1} = \boldsymbol{u}^t + \rho^t\left(\boldsymbol{W}_2^t \text{vec}\left(\boldsymbol{S}^{t+1}\right) + \boldsymbol{W}_4^t \text{vec}\left(\boldsymbol{L}^{t+1}\right) - \boldsymbol{y}\right). \tag{21}$$

Here, $\boldsymbol{W}_1^t$, $\boldsymbol{W}_2^t$, $\boldsymbol{W}_3^t$, and $\boldsymbol{W}_4^t$ are the weights of the t-th layer as shown in Figure 2. Their initial values are $\boldsymbol{W}_1^t = \boldsymbol{A}_l^*$, $\boldsymbol{W}_2^t = \boldsymbol{A}_s$, $\boldsymbol{W}_3^t = \boldsymbol{A}_s^*$, and $\boldsymbol{W}_4^t = \boldsymbol{A}_l$ to mimic the ADMM Algorithm 1. Further, $\boldsymbol{\lambda}_{LT}^t$ and $\boldsymbol{\lambda}_{ST}^t$ are the thresholding vectors of the t-th layer as given in (15) and (16). Note that $\boldsymbol{\lambda}_{LT}^t$, and $\boldsymbol{\lambda}_{ST}^t$ depend on the previous estimates of the $\boldsymbol{L}^t$, $\boldsymbol{S}^t$ and two parameters (λ_S and λ_L). Here, we consider the weights $\boldsymbol{W}_1^t$, $\boldsymbol{W}_2^t$, $\boldsymbol{W}_3^t$, and $\boldsymbol{W}_4^t$ are tied over all the layers, i.e., sharing weights. However, we do not consider thresholding parameters (λ_S and λ_L) γ and ρ to be tied over all layers, i.e., each layer has its own thresholding parameters. To this end, $\Theta = \left\{\lambda_S^t, \lambda_L^t, \gamma^t, \rho^t, \boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3, \boldsymbol{W}_4\right\}$ represents the set of learning parameters. Here, λ_S^t and λ_L^t are the thresholding parameters of the t-th layer.



$\text{vec}(\boldsymbol{S}^t)$
$\boldsymbol{u}^t$
$\boldsymbol{y}$
$\boldsymbol{W}_2$
$1/\rho^t$
$\boldsymbol{W}_1$
$\text{SVT}_{\boldsymbol{\lambda}_{LT}^t}(\cdot)$
$\boldsymbol{\lambda}_{LT}^t$
$\boldsymbol{W}_4$
$\boldsymbol{\lambda}_{ST}^t$
$\boldsymbol{W}_3$
$\text{ST}_{\boldsymbol{\lambda}_{ST}^t}(\cdot)$
$\boldsymbol{W}_2$
ρ^t
$-$
$\text{vec}(\boldsymbol{L}^{t+1})$
$\text{vec}(\boldsymbol{S}^{t+1})$
$\boldsymbol{u}^{t+1}$
$\boldsymbol{y}$

Figure 2. Block diagram of the t-th layer of the DNN which mimics the low-rank plus sparse recovery Algorithm 1. Weights of the linear layers ($\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3, \boldsymbol{W}_4$) and other parameters ($\lambda_T^t$, λ_S^t, γ^t, ρ^t) are learned from training data.

Algorithm 1: Low-rank plus sparse recovery algorithm.

```
Input: $\boldsymbol{y}$, $\epsilon = 10^{-6}$, max iterations ($J$), $\boldsymbol{A}_l$, $\boldsymbol{A}_s$.
Initialization: $\rho = 10/\sqrt{\max(M,N)}$, $\rho_o = 1.001$, $t = 0$, $\boldsymbol{L}^0 = \boldsymbol{S}^0 = \boldsymbol{0}_{M,N}$, $\boldsymbol{u}^0 = \boldsymbol{0}_K$.
while $\|\boldsymbol{A}_l\text{vec}(\boldsymbol{L}) + \boldsymbol{A}_s\text{vec}(\boldsymbol{S}) - \boldsymbol{y}\|_2^2 > \epsilon$ or $t < J$ do
    Get all the elements of $\boldsymbol{\lambda}_{LT}^t$ by (18), then estimate $\boldsymbol{L}$ by,
    $\boldsymbol{L}^{t+1} = \text{SVT}_{\lambda_{LT}^t}\left(\boldsymbol{A}_l^*\left(\boldsymbol{y} - \boldsymbol{A}_s\text{vec}(\boldsymbol{S}^t) + \frac{1}{\rho}\boldsymbol{u}^t\right)\right)$.
    Get all the elements of $\boldsymbol{\lambda}_{ST}^t$ by (17), then estimate $\boldsymbol{S}$ by,
    $\boldsymbol{S}^{t+1} = \text{ST}_{\lambda_{ST}^t}\left(\boldsymbol{A}_s^*\left(\boldsymbol{y} - \boldsymbol{A}_l\text{vec}(\boldsymbol{L}^{t+1}) + \frac{1}{\rho}\boldsymbol{u}^t\right)\right)$.
    $\boldsymbol{u}^{t+1} = \boldsymbol{u}^t + \rho\left(\boldsymbol{A}_s\text{vec}\left(\boldsymbol{S}^{t+1}\right) + \boldsymbol{A}_l\text{vec}\left(\boldsymbol{L}^{t+1}\right)\right) - \boldsymbol{y}\right)$.
    $\rho = \min(\rho_o \times \rho, \rho_m)$, and $t = t + 1$.
    $\hat{\boldsymbol{L}} \leftarrow \boldsymbol{L}^{t+1}$ and $\hat{\boldsymbol{S}} \leftarrow \boldsymbol{S}^{t+1}$.
Output: $\hat{\boldsymbol{L}}$, $\hat{\boldsymbol{S}}$.
```

3.1. Training Phase

In the training phase, the DNN is trained in a supervised manner. Here, the DNN learns the parameters given in $\Theta = \left\{\lambda_S^t, \lambda_L^t, \gamma^t, \rho^t, \boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3, \boldsymbol{W}_4\right\}$. Suppose that the DNN has T layers, then the outputs of the DNN in the training phase for the i-th sample are given by $\hat{\boldsymbol{L}}_i$ and $\hat{\boldsymbol{S}}_i$, respectively. Note that, in the training phase, the DNN minimizes the normalized mean squared error

$$\text{NMSE} = \frac{1}{T_s} \sum_{i=1}^{T_s} \left(\frac{1}{2} \frac{\|\hat{\boldsymbol{L}}_i - \boldsymbol{L}_i\|_F^2}{\|\boldsymbol{L}_i\|_F^2} + \frac{1}{2} \frac{\|\hat{\boldsymbol{S}}_i - \boldsymbol{S}_i\|_F^2}{\|\boldsymbol{S}_i\|_F^2} \right), \tag{22}$$

where $\boldsymbol{S}_i$ and $\boldsymbol{L}_i$ are i-th ground-truth low-rank and sparse matrices, and T_s is the number of training samples.

In this work, in the context of DNN-based parameter tuning, we consider three versions of the ADMM-based iterative algorithm to solve the RPCA problem as follows: (a) Parameter tuning with non-adaptive thresholding (i.e., $g_s(x) = g_l(x) = 1$). This approach is named as ADMM-based trained RPCA with thresholding (TRPCA-T). For the parameter tuning with adaptive thresholding, we consider two versions based on two decay functions as described in Section 2.4. These two approaches are named as follows: (b) ADMM-based trained RPCA with adaptive thresholding based on logarithm heuristic (TRPCA-AT(log)). (c) ADMM-based trained RPCA with adaptive thresholding based on exponential heuristic (TRPCA-AT(exp). Among above versions, in this work, we propose parameter tuning with adaptive thresholding approaches TRPCA-AT(log) and TRPCA-AT(exp) to solve the RPCA problem with the compressive sensing data acquisition model.

To have a comparison with our proposed approaches, we consider two approaches. In the first approach, we consider the untrained ADMM approach to solve the convex low-rank plus sparse recovery as given in Algorithm 1 with non-adaptive thresholding (i.e., $g_s(x) = g_l(x) = 1$). This method is named as ADMM-based untrained RPCA with thresholding (URPCA-T). As a second option, we consider the low-rank plus sparse recovery problem given in (8). This method is named as low-rank plus sparse recovery with convex relaxation (LRPSRC).

3.2. Computation Complexity

In this subsection, the computational complexity of the proposed DNN is briefly discussed. Detail breakdown is given in Appendix A.2. The training complexity of the DNN is the addition of the feed-forward and the back propagation complexities. For T_s, number of training samples, E_p, number of epochs, and for $M = N$, the training computational complexity for the DNN with T layers is given by $\mathcal{O}(TT_sE_p(N^2K + N^3))$. The testing computational complexity is the feed-forward propagation complexity of data through the DNN. It is given by $\mathcal{O}(TN_s(N^2K + N^3))$; here N_s is the number of testing samples. The $\mathcal{O}(\cdot)$ is the Big O notation for asymptotic computational complexity analysis [49].

4. Results and Discussion

In this section numerical results are presented. First, the performance of deep learning-based trained ADMM adaptive thresholding is evaluated with a generic real-valued Gaussian model, and next, a complex-valued SFCW radar model given in Section 2.1 is used.

4.1. Generic Gaussian Model

In this subsection, our proposed approach is evaluated using the generic Gaussian data. The order of this subsection is summarized as follows. First, the performance of the proposed approach is compared with state-of-the-art approaches for 50% and 25% compression ratios. Second, the Cramér–Rao bound (CRB) of unbiased estimation of low-rank and sparse matrices is used to evaluate the proposed approach. Third, to investigate the robustness of the proposed approach, two scenarios are used: (a) testing SNR uncertainty and (b) deviation in measurement matrices $\boldsymbol{A}_l$ and $\boldsymbol{A}_s$ between training and testing. Fourth, the performance comparison between ADMM- and FISTA-based approaches for RPCA is evaluated. Here, the approach given in [30] (CORONA) is used as unfolded FISTA-based approach for RPCA.

In the generic Gaussian model, the elements of $\boldsymbol{A}_l = \boldsymbol{A}_s = \boldsymbol{A} \in \mathbb{R}^{K \times MN}$ are generated once from an i.i.d. Gaussian with zero mean and unit variance. In this work, training and testing data are synthetically generated based on the system model given in (1). Therefore, ground-truth low-rank and sparse matrices are available in the training phase. In case only the received data vector $\boldsymbol{y}$ in (1) is available, in general, Algorithm 1 or LRPSRC given in (8) can be used to generate low-rank and sparse matrices in the training phase. Let the received signal, noise vector, and low-rank and sparse matrices for the i-th data sample as given in (1) be denoted by $\boldsymbol{y}_i \in \mathbb{R}^K$, $\boldsymbol{n}_i \in \mathbb{R}^K$, $\boldsymbol{L}_i \in \mathbb{R}^{M \times N}$, and $\boldsymbol{S}_i \in \mathbb{R}^{M \times N}$, respectively. We generate a low-rank matrix $\boldsymbol{L}_i$ with rank r as $\boldsymbol{L}_i = \boldsymbol{G}_i \boldsymbol{H}_i^T$ with $\boldsymbol{G}_i \in \mathbb{R}^{M \times r}$ and $\boldsymbol{H}_i \in \mathbb{R}^{N \times r}$. Here, elements of $\boldsymbol{G}_i$, $\boldsymbol{H}_i$ and non-zero entries of $\boldsymbol{S}_i$ are generated independently from an i.i.d. Gaussian with zero mean and unit variance. The fixed number of non-zero locations of each $\boldsymbol{S}_i$ are selected uniformly. We normalized $\boldsymbol{S}_i$ and $\boldsymbol{L}_i$ to have a unit Frobenius norm (i.e., $\|\boldsymbol{L}_i\|_F^2 = \|\boldsymbol{S}_i\|_F^2 = 1$). For better readability, we introduce a parameter set as $P = \{M, N, T_s, N_s, L_p, L_w, E_p, r, \|\boldsymbol{S}_i\|_0, \mathrm{SNR}_{\mathrm{tr}}, \mathrm{SNR}_{\mathrm{t}}\}$. Here, T_s, N_s, E_p, r, $\|\boldsymbol{S}_i\|_0$, $\mathrm{SNR}_{\mathrm{tr}}$, and $\mathrm{SNR}_{\mathrm{t}}$ are the number of training samples, number of testing samples, number of epochs, rank of the low-rank matrix, number of non-zero elements of the sparse matrix, SNR of the training data, and SNR of the testing data, respectively. The signal-to-noise ratio (SNR) of the i-th data sample for given $\boldsymbol{A}$ is defined as $\mathrm{SNR} := \|\boldsymbol{A}\mathrm{vec}(\boldsymbol{L}_i + \boldsymbol{S}_i)\|_2^2 / \|\boldsymbol{n}_i\|_2^2$. First, we generate a Gaussian noise vector $\boldsymbol{n}_i$, then re-normalize $\boldsymbol{n}_i$ to reach a given target SNR, and we set the same SNR for all samples.

In the training stage, we set different learning rates denoted by L_w and L_p for the weights of the linear layers ($\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3, \boldsymbol{W}_4$) and other parameters ($\lambda_S^t$, λ_L^t, γ^t, ρ^t) given in Θ. The main objective for setting different learning rates is to reduce over-fitting to training data. Generally, many training samples are required to train a deep neural network. However, due to the specific architecture of the iterative algorithm, we are able to train the DNN with a small data set with the number of training samples $T_s = 500$. In the training phase, the adaptive moment estimation (Adam) optimizer [50] is used to train the DNN. Here, we initialize $\boldsymbol{W}_1^t = \boldsymbol{A}_l^\dagger$, $\boldsymbol{W}_2^t = \boldsymbol{A}_s$, $\boldsymbol{W}_3^t = \boldsymbol{A}_s^\dagger$, and $\boldsymbol{W}_4^t = \boldsymbol{A}_l$ to mimic the ADMM Algorithm 1 and $\gamma = 1$. In the inference phase, to evaluate the performance of the DNN, the normalized average root mean squared error is used. For the low-rank and sparse matrix, it is given by

$$\mathrm{NRMSE_L} = \frac{1}{N_s} \sum_{i=1}^{N_s} \left(\frac{\|\hat{\boldsymbol{L}}_i - \boldsymbol{L}_i\|_F}{\|\boldsymbol{L}_i\|_F} \right), \tag{23}$$

$$\mathrm{NRMSE_S} = \frac{1}{N_s} \sum_{i=1}^{N_s} \left(\frac{\|\hat{\boldsymbol{S}}_i - \boldsymbol{S}_i\|_F}{\|\boldsymbol{S}_i\|_F} \right). \tag{24}$$

The outputs of a DNN with T layers for the i-th testing sample are given by $\hat{\boldsymbol{S}}_i$ and $\hat{\boldsymbol{L}}_i$, respectively. The CRB given in (A4) is based on the combined recovery error of both low-rank and sparse matrices. The combined average mean squared error and the combined average normalized root mean squared error for both low-rank and sparse matrices are given by

$$\mathrm{MSE_{LS}} = \frac{1}{N_s} \sum_{i=1}^{N_s} \left(\|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i\|_2^2 \right), \tag{25}$$

$$\mathrm{NRMSE_{LS}} = \frac{1}{N_s} \sum_{i=1}^{N_s} \left(\frac{\|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i\|_2}{\|\boldsymbol{x}_i\|_2} \right), \tag{26}$$

in which $\boldsymbol{x}_i = [\mathrm{vec}(\boldsymbol{L}_i)^T\ \mathrm{vec}(\boldsymbol{S}_i)^T]^T$ and $\hat{\boldsymbol{x}}_i = [\mathrm{vec}(\hat{\boldsymbol{L}}_i)^T\ \mathrm{vec}(\hat{\boldsymbol{S}}_i)^T]^T$.

Both Algorithm 1 and LRPSRC given in (8) are implemented using Matlab [51], and LRPSRC is solved using the CVX package [52]. Notice that, in the LRPSRC, λ_l and λ_s are set to 1 and $1/\sqrt{\max(M, N)}$, respectively, as suggested by [17]. Note that for Algorithm 1, there is no specific rule to select the λ_l and λ_s and ρ, thus they are manually tuned based on data. When $\boldsymbol{A}$ is identity matrix, there is a specific rule to select λ_s as $1/\sqrt{\max(M,N)}$ [17]. Note that, as a rule of thumb, thresholding parameters λ_S and λ_L given in (17) and (18) are initialized as $\lambda_S = \lambda_s/\rho$ and $\lambda_L = \lambda_l/\rho$, respectively [17]. The ADMM penalty factor ρ has an important impact on the convergence of the Algorithm 1. Usually, as ρ increases, the algorithm converges faster. However, ρ cannot be arbitrarily large, as it may overshoot the algorithm. Furthermore, ρ should not be too big or too small. However, finding an optimal value for ρ is an open problem, and it depends on the application/data. As a rule of thumb, ρ can be set as $0.25MN/\|\boldsymbol{y}\|_1$ [27]. In this work, we set $\rho = 10/\sqrt{\max(M,N)}$, as we observed that the value suggested in [27] is not optimal for our data. Note that, unless otherwise stated, for all the simulation with Gaussian data, aforementioned parameter settings are used throughout this paper. The Pytorch package was used to implement the DNN [53].

First, we analyzed the performance of the proposed approach for different compression ratios (K/MN) with respect to the number of layers of the DNN. For this simulation, the parameter set P is given by $P = \{30, 30, 500, 1200, 0.1, 0, 500, 2, 5, 20\text{ dB}, 20\text{ dB}\}$. Here, DNN only learns λ_S^t, λ_L^t, and ρ^t instead of all the parameters given in Θ, i.e., $L_w = 0$. This is due to the fact that the performance gain improvement by learning all the parameters given in Θ is very small compared to learning only λ_S^t and λ_L^t. The average normalized RMSEs for the different number of layers of the DNN are, for compression ratio (K/MN) 50% and 25%, shown in Figures 3 and 4, respectively. Figures 3 and 4 show that the proposed DNN-based thresholding (TRPCA-AT(log) and TRPCA-AT(exp)) outperforms the URPCA-T and the LRPSRC. Further, it is observed that as the number of layers increases, the average NRMSE decreases. For 50% compression ratio, the average NRMSE does not show a large variance after ten layers. However, for 25%, this is not the case. This is due to the fact that, as the compression increases, recovering of the low-rank and the sparse matrices becomes more challenging.

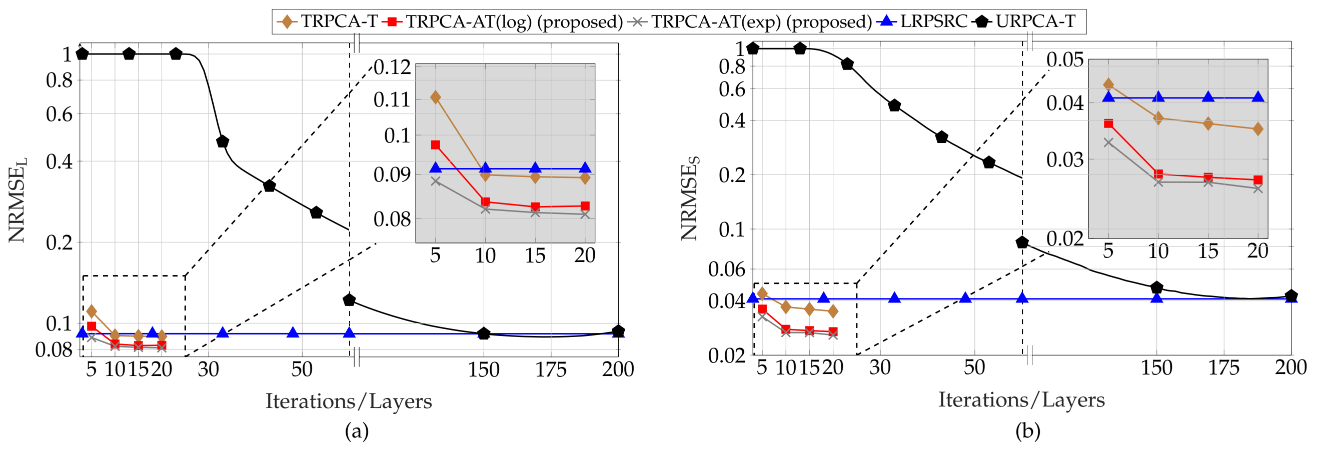


Figure 3. Average recovery error of low-rank (**a**) and sparsity (**b**) contributions for compression ratio $K/MN = 50\%$ for ADMM-based trained RPCA with thresholding (TRPCA-T), proposed ADMM-based trained RPCA with adaptive thresholding based on logarithm heuristic (TRPCA-AT(log)), proposed ADMM-based trained RPCA with adaptive thresholding based on exponential heuristic (TRPCA-AT(exp)), low-rank plus sparse recovery with convex relaxation (LRPSRC), and ADMM-based untrained RPCA with thresholding (URPCA-T).

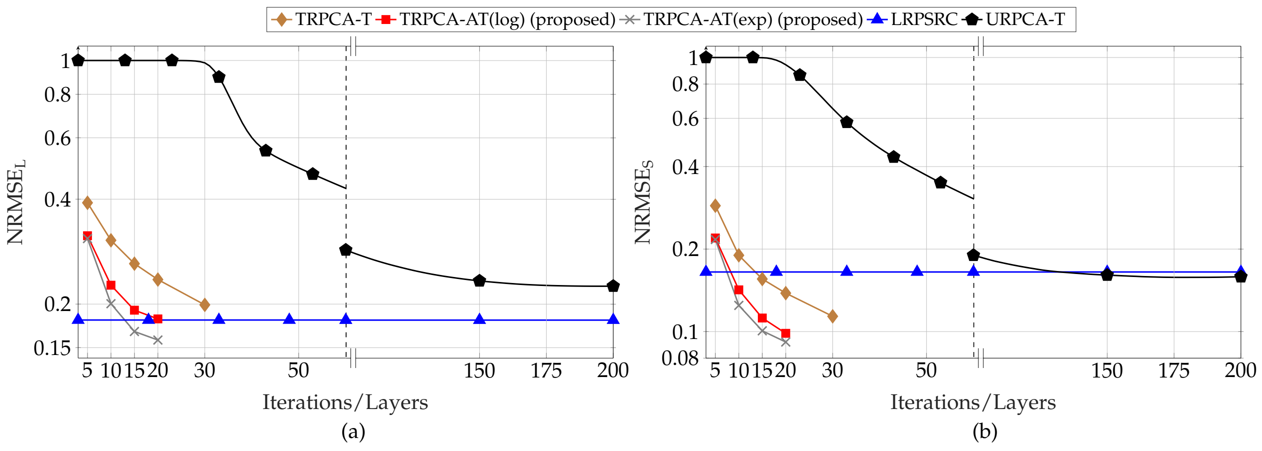


Figure 4. Average recovery error of low-rank (**a**) and sparsity (**b**) contributions for compression ratio $K/MN = 25\%$ for ADMM-based trained RPCA with thresholding (TRPCA-T), proposed ADMM-based trained RPCA with adaptive thresholding based on logarithm heuristic (TRPCA-AT(log)), proposed ADMM-based trained RPCA with adaptive thresholding based on exponential heuristic (TRPCA-AT(exp)), low-rank plus sparse recovery with convex relaxation (LRPSRC), and ADMM-based untrained RPCA with thresholding (URPCA-T).

Further, the TRPCA-AT outperforms the TRPCA-T. This performance improvement is mainly due to the iterative reweighting of ℓ_1-norm and nuclear norm minimization. In addition, the improvement over unweighted to iterative reweighting is more visible as the compression increases (i.e., as the problem gets more challenging). As an example, for 25% compression ratio, the average NRMSE improvement between the TRPCA-T with twenty layers and TRPCA-AT(exp) with twenty layers for the low-rank and sparse components are 32.93% and 50.77%, respectively. However, for 50% compression ratio, this improvement for the low-rank and sparse components are 9.31% and 26.21%, respectively. Further, we observe slight performance gains as the decay function is changed from log-determinant to exponential.

Next, we analyzed the convergence speed of the proposed TRPCA-AT(exp) and TRPCA-AT(log) with URPCA-T. For 50% of compression ratio, TRPCA-AT with ten layers outperforms URPCA-T with 150 iterations. Therefore, in the testing phase (inference phase), our proposed approaches (TRPCA-AT(exp) and TRPCA-AT(log)) are fifteen times faster than the conventional untrained approach URPCA-T. Moreover, for 25% of compression ratio, TRPCA-AT with twenty layers outperforms URPCA-T with 150 iterations. Thus, our approach is 7.5 times faster than the untrained approach URPCA-T. It is worth noticing that one layer of the DNN of our proposed approach is equivalent to one iteration of the conventional untrained approach URPCA-T. Therefore, the proposed approaches (TRPCA-AT(exp) and TRPCA-AT(log)) achieve lower NRMSE than the untrained approach URPCA-T with a much lower number of iterations. In Table 1, NRMSEs of recovered low-rank and sparse matrices with the corresponding number of iterations are listed for comparison.

To further demonstrate the advantage of non-convex iterative reweighting of ℓ_1-norm and nuclear norm minimization, histograms of the non-zero singular values of the low-rank matrix and non-zero element of the sparse matrix are shown in Figure 5 for the DNN with 20 layers. Here, these histograms correspond to the simulation given in Figure 3, i.e., 1200 testing samples and compression ratio $K/MN = 50\%$. Based on Figure 5, for the sparse matrix $\boldsymbol{S}$, the proposed non-convex iterative reweighted approaches (TRPCA-AT(exp) and TRPCA-AT(log)) closely follow the histogram of the true sparse matrix. Moreover, for a given value range, the number of occurrences of the recovered sparse matrix by the unweighted approach TRPCA-T is less than the true number of occurrences of that value range as shown in Figure 5a. However, this is not the case for the non-convex iterative reweighted approaches (TRPCA-AT(exp) and TRPCA-AT(log)).

These results validate that the important features preserved by the large coefficients are well recovered by the iterative reweighted approaches. This is the reason for the performance improvement of the iterative reweighted approaches compared to the unweighted approach TRPCA-T. In addition, recovered sparse matrices by the unweighted approach TRPCA-T have many small values compared to the iterative reweighted approaches. This indicates that the iterative reweighted approaches achieve more sparse solution than the unweighted approach.

As seen in Figure 5, histograms of the non-zero singular values of the low-rank matrix by the proposed non-convex iterative reweighted approaches are less spread out compared to the histogram of the unweighted approach TRPCA-T. This also validates the aforementioned argument that important features preserved by the large coefficients are well recovered by the iterative reweighted approaches TRPCA-AT(exp) and TRPCA-AT(log). Note that in the histograms, the number of occurrences of zero value is not shown. This is due to the fact that the number of occurrences of zero value is much larger than occurrences of other values. In Figure 5, histograms corresponding to the compression ratio $K/MN = 50\%$ are shown, and for the compression ratio $K/MN = 25\%$, similar results were observed.

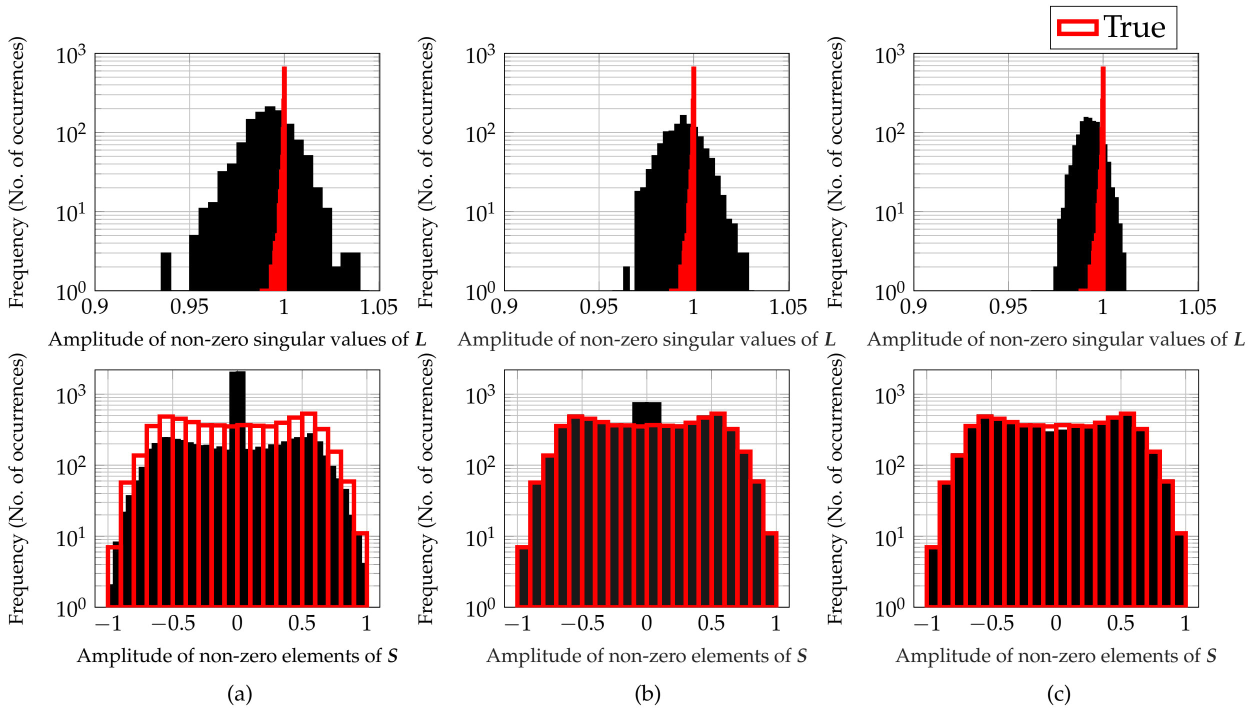


Figure 5. Histograms of the non-zero singular values of L (**top**) and non-zero elements of S (**bottom**) for $K/MN = 50\%$. (**a**) TRPCA-T, (**b**) TRPCA-AT(log) (proposed) and (**c**) TRPCA-AT(exp) (proposed). Note that, In the figure, true histograms are shown in red color and the recovered histograms are shown in black color. It is noticeable that the proposed non-convex iterative reweighted approaches (TRPCA-AT(exp) and TRPCA-AT(log)) closely follow the histograms of the true non-zero elements of S and non-zero singular values of L compared to the unweighted approach TRPCA-T. In addition, the recovered S by the unweighted approach TRPCA-T has many small values compared to the iterative reweighted approaches, i.e., the iterative reweighted approach achieves a more sparse solution than the unweighted approach.

Table 1. Comparison of convergence speeds for ADMM-based trained RPCA with thresholding (TRPCA-T), proposed ADMM-based trained RPCA with adaptive thresholding based on logarithm heuristic (TRPCA-AT(log)), proposed ADMM-based trained RPCA with adaptive thresholding based on exponential heuristic (TRPCA-AT(exp)), and ADMM-based untrained RPCA with thresholding (URPCA-T). The proposed approaches TRPCA-AT(log) and TRPCA-AT(exp) are 15 and 7.5 times faster than URPCA-T for compression ratios 50% and 25%, respectively.

Method	**Compression Ratio (K/MN) %**	**Number of Iterations**	$\mathbf{NRMSE} = \frac{1}{N_s}\sum_{i=1}^{N_s}\left(\|X_i - \hat{X}_i\|_F \big/ \|X_i\|_F\right)$	
			Low-Rank Matrix L	**Sparse Matrix S**
TRPCA-AT(log)	50%	10	8.36×10^{-2}	2.78×10^{-2}
TRPCA-AT(exp)	50%	10	8.20×10^{-2}	2.66×10^{-2}
TRPCA-T	50%	10	8.99×10^{-2}	3.69×10^{-2}
URPCA-T	50%	150	9.14×10^{-2}	4.72×10^{-2}
TRPCA-AT(log)	25%	20	1.81×10^{-1}	9.85×10^{-2}
TRPCA-AT(exp)	25%	20	1.57×10^{-1}	9.16×10^{-2}
TRPCA-T	25%	20	2.35×10^{-1}	1.38×10^{-1}
URPCA-T	25%	150	2.33×10^{-1}	1.61×10^{-1}

4.1.1. Cramér–Rao Bound (CRB) Analysis

To further evaluate the performance of the proposed approach, the Cramér-Rao bound (CRB) of unbiased estimation of low-rank and sparse matrices given in [18] (Equation (A4)) is used. For completeness, the CRB and recovery guarantees of the RPCA are given in Appendix A.3. Note that in [18], the measurement matrices $\boldsymbol{A}_l$ and $\boldsymbol{A}_s$ are assumed to be a selection operator. Therefore, to have a fair comparison, first we consider that both $\boldsymbol{A}_l$ and $\boldsymbol{A}_s$ are identity matrices, i.e., standard RPCA problems. Now, the data acquisition model (Equation (1)) is simplified as

$$\boldsymbol{Y} = \boldsymbol{L} + \boldsymbol{S} + \boldsymbol{N}, \tag{27}$$

where $\boldsymbol{Y}$ and $\boldsymbol{N}$ are received signal matrix and noise matrix of size $M \times N$, respectively. For this simulation, parameter set P is given by $\{30, 30, 500, 1200, 1 \times 10^{-2}, 5 \times 10^{-4}, 10, 2, 5, [-5 : 5 : 20]\text{ dB}, [-5 : 5 : 20]\text{ dB}\}$. In this simulation, we set the number of layers of the DNN as 10.

Figure 6 shows the CRB and average MSE of the combined low-rank and sparse matrices for 1200 testing samples for different SNR levels ranging from -5 dB to 20 dB in steps of 5 dB. Here, we consider same SNR in both training and testing. As an example, if the testing SNR (SNR_t) is 20 dB, then training SNR (SNR_tr) is 20 dB. As per Figure 6, it can be seen that the non-convex approach TRPCA-AT has the best performance compared to other approaches in higher SNR regime. Note that, here, the performance gap between the non-convex approach TRPCA-AT and the non-convex approach TRPCA-T is small. This is due to the fact that, as observed in Figures 3 and 4, when compression decreases, the gain achieve by the non-convex approaches decreases.

Next, we compare the results shown in Figures 3 and 4 with the CRB given in [18] (Equation (A4)). In [18], the measurement matrix $\boldsymbol{A}$ is assumed to be a selection operator which selects a random subset of size K from MN entries. Since this is the closest matching CRB to our model given in (1), we have considered this formulation as a benchmark. Further, we consider that $\boldsymbol{A}$ is fixed over all testing samples. Figure 7 shows the CRB of the combined low-rank and sparse matrices for compression ratios 50% and 25%. It can be seen that the non-convex approach TRPCA-AT has the closest performance to the CRB. As the compression increases, the estimation of low-rank and sparse matrices from compressive measurements becomes more challenging. This can be seen by the increase of CRB as the compression ratio changes from 50% to 25%.

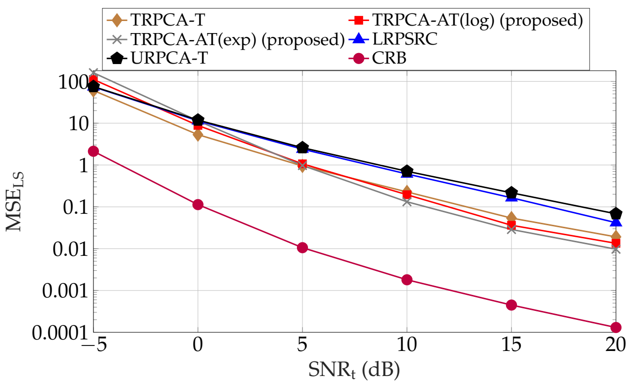


Figure 6. Average combined recovery error of low-rank and sparse matrices as given in (25) and Cramér-Rao bounds as given in (A4) for compression ratio $K/MN = 100\%$ for ADMM-based trained RPCA with thresholding (TRPCA-T), proposed ADMM-based trained RPCA with adaptive thresholding based on logarithm heuristic (TRPCA-AT(log)), proposed ADMM-based trained RPCA with adaptive thresholding based on exponential heuristic (TRPCA-AT(exp)), low-rank plus sparse recovery with convex relaxation (LRPSRC), and ADMM-based untrained RPCA with thresholding (URPCA-T).

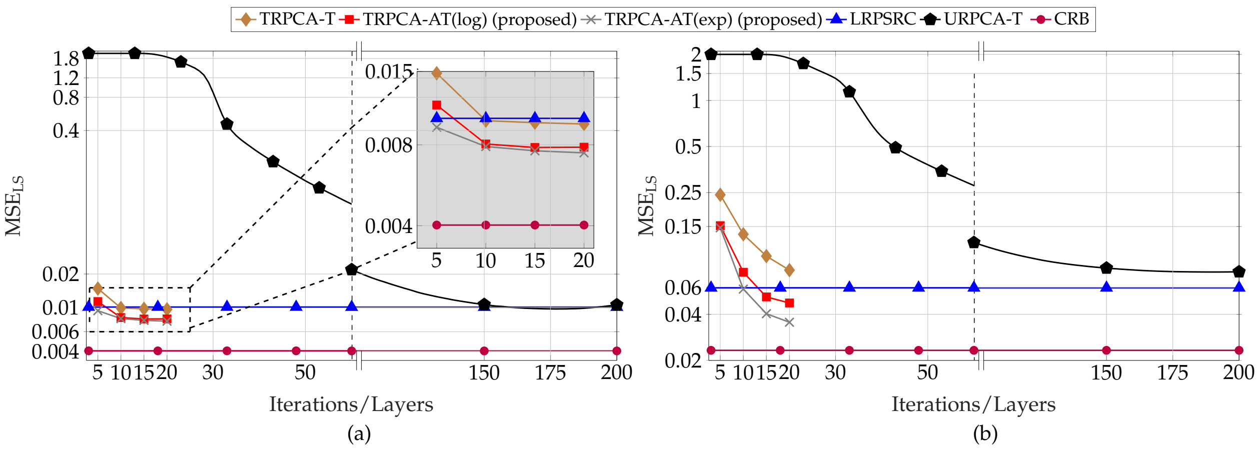


Figure 7. Average combined recovery error of low-rank and sparse matrices as given in (25) and Cramér-Rao bounds as given in (A4) for compression ratio $K/MN = 50\%$ (**a**) and $K/MN = 25\%$ (**b**) for ADMM-based trained RPCA with thresholding (TRPCA-T), proposed ADMM-based trained RPCA with adaptive thresholding based on logarithm heuristic (TRPCA-AT(log)), proposed ADMM-based trained RPCA with adaptive thresholding based on exponential heuristic (TRPCA-AT(exp)), low-rank plus sparse recovery with convex relaxation (LRPSRC), and ADMM-based untrained RPCA with thresholding (URPCA-T).

4.1.2. Robustness of the Proposed Approach

We considered two scenarios to analyze the robustness of the proposed trained ADMM adaptive thresholding approaches TRPCA-AT and TRPCA-T. First, motivated by [54], we analyzed the performance with respect to the test SNR uncertainty, i.e., the SNRs of training phase and testing phase are different. Second, we analyzed the effect of deviations in the measurement matrices $\boldsymbol{A}_l$ and $\boldsymbol{A}_s$ in (1) between training and testing.

To this end, to evaluate the effect of testing SNR uncertainty, training SNR (SNR_{tr}) is changed from -10 dB to 20 dB with a step size of 5 dB. In addition, testing SNR (SNR_{t}) is changed from -5 dB to 20 dB with a step size of 5 dB. For this simulation, P is given by $\{30, 30, 500, 1200, 1 \times 10^{-1}, 5 \times 10^{-4}, 20, 2, 5, [-10 : 5 : 20]\ \text{dB}, [-5 : 5 : 20]\ \text{dB}\}$. The cumulative average NRMSE_{LS} (where NRMSE_{LS} is defined in (26)) over all testing SNRs for each training SNR is shown in Figure 8. Here, we set $K/MN = 50\%$ and number of layers of the DNN as 10. We trained the DNN for 20 epochs using the Adam. Based on the results shown in Figure 8, we observed that, for all three approaches (TRPCA-AT(log), TRPCA-AT(exp), TRPCA-T), the cumulative average NRMSE_{LS} decreases as training SNR increases to some extent, and then, again, the cumulative average increases as training SNR further increases. Hence, these results show the importance of knowing the testing SNR, and as a simple rule, training SNR should be same as testing SNR to achieve the best performance. On the other hand, training with an SNR $\approx$ 5 dB is favorable in the presence of uncertainty about testing SNR.

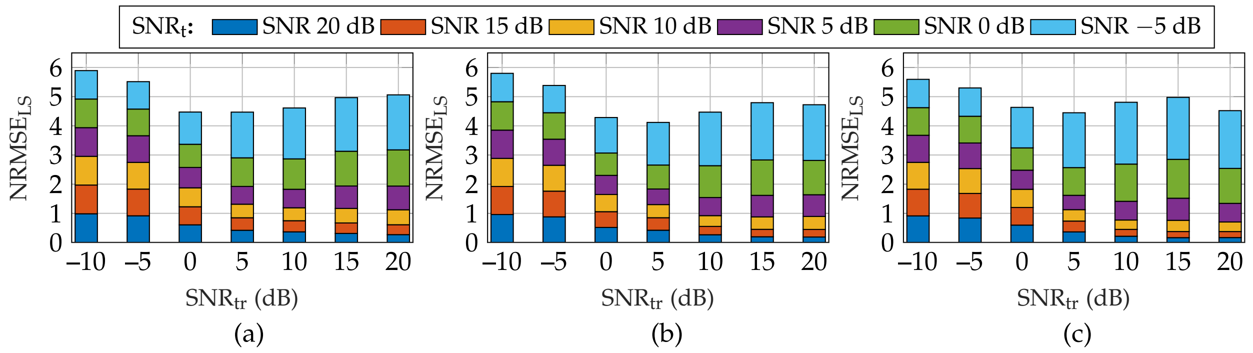


Figure 8. Average combined recovery error of low-rank and sparse matrices for compression ratio $K/MN = 50\%$ for training at a single SNR and testing with different SNRs for (**a**) ADMM-based trained RPCA with thresholding (TRPCA-T), (**b**) the proposed ADMM-based trained RPCA with adaptive thresholding based on logarithm heuristic (TRPCA-AT(log)), and (**c**) the proposed ADMM-based trained RPCA with adaptive thresholding based on exponential heuristic (TRPCA-AT(exp)). In the presence of uncertainty about testing SNR, then training with an SNR $\approx$ 5 dB is favorable.

Next, we evaluate the performance of the proposed approaches for different measurement matrices $\boldsymbol{A}_l$ and $\boldsymbol{A}_s$ in (1) during training and testing. For simplicity, we assume that $\boldsymbol{A}_l = \boldsymbol{A}_s = \boldsymbol{A} \in \mathbb{R}^{K\times MN}$. In the training phase, $\boldsymbol{y} = \boldsymbol{A}\text{vec}(\boldsymbol{L}+\boldsymbol{S}) + \boldsymbol{n}$ while in the testing phase $\boldsymbol{y} = \bar{\boldsymbol{A}}\text{vec}(\boldsymbol{L}+\boldsymbol{S}) + \boldsymbol{n}$. Here, $\bar{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{E} \in \mathbb{R}^{K\times MN}$ is the measurement matrix with error and $\boldsymbol{E} \in \mathbb{R}^{K\times MN}$. To quantify the effect of $\boldsymbol{E}$, $\text{SNR}_{\text{A}} := \|\text{vec}(\boldsymbol{A})\|_2^2 / \|\text{vec}(\boldsymbol{E})\|_2^2$ is used as a metric. We evaluate the performance of the proposed approaches while changing SNR_{A} from 0 dB to 20 dB in steps of 5 dB. For this simulation, parameter set P is given by $\{30, 30, 500, 1200, 1\times10^{-1}, 1\times10^{-6}, 50, 2, 5, 20\text{ dB}, \text{SNR}_{\text{t}}\}$. Note that testing SNR varies as SNR_{A} changes, therefore, it is shown as SNR_{t} in parameter set P. Figure 9 shows the average NRMSEs of the combined low-rank and sparse matrices $K/MN = 50\%$ and 25%. In Figure 9, solid lines represent the NRMSEs for the model with measurement matrix error ($\bar{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{E}$); we also include a prefix "-E" in the legend of the figure to indicate it. In Figure 9, the dotted line shows the NRMSEs without error in the measurement matrix. Based on the results shown in Figure 9, the proposed approaches are robust for smaller deviations like $\text{SNR}_{\text{A}} = 20$ and 15 dB. However, for higher deviations $\text{SNR}_{\text{A}} \leq 10$ dB, the proposed approaches are not robust enough and additional measures are required to rectify the matrix deviation. As a countermeasure, we assume that the model error distribution is available in training as well. Here, both i-th sample of training and testing data are generated by $\boldsymbol{y}_i = \bar{\boldsymbol{A}}\text{vec}(\boldsymbol{L}_i + \boldsymbol{S}_i) + \boldsymbol{n}_i$. For training, $\bar{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{E}_{tr,i}$, and for testing, $\bar{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{E}_{t,i}$. Here, for each training and testing sample, $\boldsymbol{E}_{tr,i}$ and $\boldsymbol{E}_{t,i}$ are generated independently from an i.i.d. Gaussian with zero mean and unit variance. For comparison, we include results with training without error distribution, i.e., training data are generated as $\boldsymbol{y}_i = \boldsymbol{A}\text{vec}(\boldsymbol{L}_i + \boldsymbol{S}_i) + \boldsymbol{n}_i$ while keeping testing data the same, i.e., $\boldsymbol{y}_i = \bar{\boldsymbol{A}}\text{vec}(\boldsymbol{L}_i + \boldsymbol{S}_i) + \boldsymbol{n}_i$. Note that this result is shown as solid lines in Figure 10. As shown in Figure 10, when model error distribution is included in training (dotted line in Figure 10), the NRMSEs show improvement over training without distribution of $\boldsymbol{E}$ when SNR_{A} is in the range of 0 dB to 15 dB. Moreover, as SNR_{A} increases, i.e., deviation decreases, training without distribution of error $\boldsymbol{E}$ provides similar results as training with distribution of $\boldsymbol{E}$. As a conclusion, when there is high deviation in the measurement matrix, a robust training approach, i.e., training with distribution of $\boldsymbol{E}$, provides an advantage.

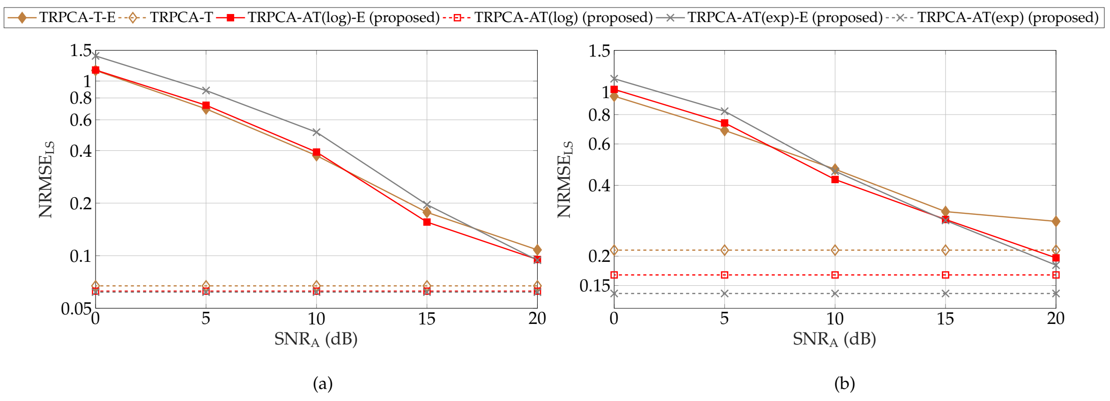


Figure 9. Average combined recovery error of low-rank and sparse matrices for compression ratio $K/MN = 50\%$ (**a**) and $K/MN = 25\%$ (**b**) with different model error levels for ADMM-based trained RPCA with thresholding (TRPCA-T), proposed ADMM-based trained RPCA with adaptive thresholding based on logarithm heuristic (TRPCA-AT(log)), and proposed ADMM-based trained RPCA with adaptive thresholding based on exponential heuristic (TRPCA-AT(exp)). Here, model error means that the train and testing samples are generated with different measurement matrices. For training, $\boldsymbol{y} = \boldsymbol{A}\text{vec}(\boldsymbol{L} + \boldsymbol{S}) + \boldsymbol{n}$; for testing, $\boldsymbol{y} = \bar{\boldsymbol{A}}\text{vec}(\boldsymbol{L} + \boldsymbol{S}) + \boldsymbol{n}$ with $\bar{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{E}$. The results with model error are represented by solid lines, whereas dotted lines indicates the results without model error ($\boldsymbol{E} = \boldsymbol{0}_{K,MN}$).

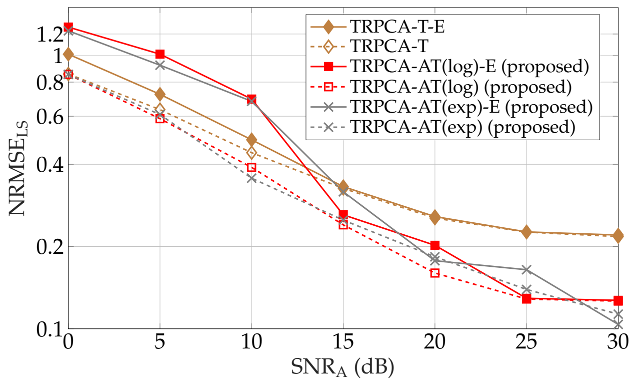


Figure 10. Average combined recovery error of low-rank and sparse matrices for compression ratio $K/MN = 25\%$ with different model error levels for ADMM-based trained RPCA with thresholding (TRPCA-T), proposed ADMM-based trained RPCA with adaptive thresholding based on logarithm heuristic (TRPCA-AT(log)), and proposed ADMM-based trained RPCA with adaptive thresholding based on exponential heuristic (TRPCA-AT(exp)). Here, model error means that the training and testing samples are generated with different measurement matrices. For training, $\boldsymbol{y} = \boldsymbol{A}\text{vec}(\boldsymbol{L} + \boldsymbol{S}) + \boldsymbol{n}$; for testing, $\boldsymbol{y} = \bar{\boldsymbol{A}}\text{vec}(\boldsymbol{L} + \boldsymbol{S}) + \boldsymbol{n}$ with $\bar{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{E}$. The results with model error are represented by solid lines, whereas dotted lines indicates the results when model error distribution is included in training.

4.1.3. ADMM or FISTA to Solve RPCA Problem

In this work, we consider an iterative algorithm based on the ADMM [39] to solve the RPCA problem. Alternatively, other methods such as ISTA and FISTA can be used [30,37]. In the following, we first compare the performance of the untrained ADMM-based Algorithm 1 with $g_s(x) = g_l(x) = 1$ (URPCA-T) and the untrained algorithm based on FISTA as given in [30,37]. Further, we consider three different combinations for the rank of $\boldsymbol{L}$ and sparsity of $\boldsymbol{S}$ with $\|\boldsymbol{S}\|_0 = MNp_s$. The aforementioned three combinations are given by $\text{rank}(\boldsymbol{L}) = \{1, 2, 2\}$ and $p_s = \{0.1, 0.1, 0.2\}$. Here, we consider 250 test samples in each combination. It turns out that for all three combinations, the ADMM-based approach achieves lower NRMSEs with fewer numbers of iterations compared to the FISTA, as shown in Figure 11. In this simulation, we consider standard RPCA where $\boldsymbol{A}_l$, $\boldsymbol{A}_s$ in (1) are equal to the identity matrix. We chose this scenario because it is the simplest non-compression scenario. Note that, for FISTA, soft-thresholding and singular value thresholding parameters are set as 0.05 and $0.1\max(\boldsymbol{\sigma}(\mathbf{Y}))$, in which $\boldsymbol{\sigma}(\mathbf{Y})$ are the singular values of $\mathbf{Y}$.

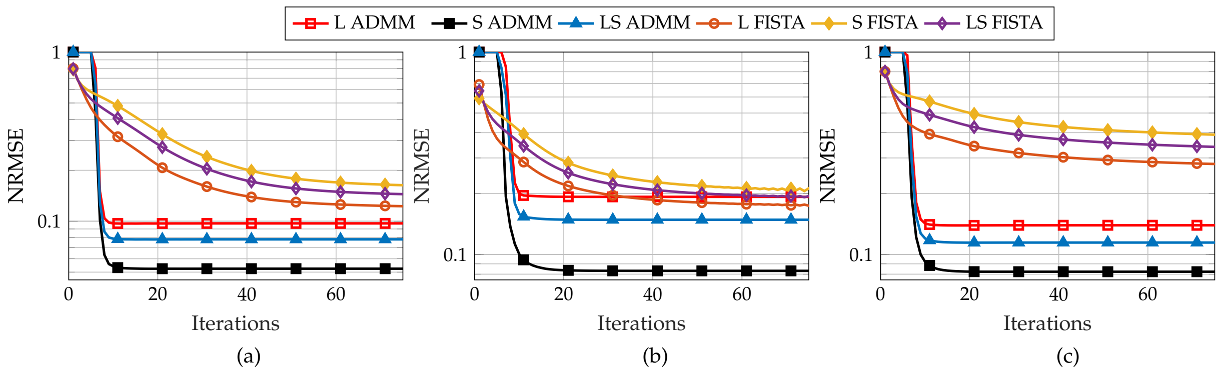


Figure 11. Average NRMSE of low-rank and sparsity contributions for $K/MN = 100\%$ for ADMM- and FISTA-based approaches. (**a**) $\text{rank}(\boldsymbol{L}) = 1$ and $p_s = 0.1$, (**b**) $\text{rank}(\boldsymbol{L}) = 2$ and $p_s = 0.1$ and (**c**) $\text{rank}(\boldsymbol{L}) = 2$ and $p_s = 0.2$. The ADMM-based approach achieves a lower NRMSE with a lower number of iterations compared to the FISTA-based approach.

4.1.4. Performance Evaluation for Experimental Ultrasound Imaging Data

To further assess the performance of ADMM- and FISTA-based approaches in the context of algorithm unfolding, we consider the FISTA-based unfolded approach in [30] (CORONA). For fair comparison, we consider two types of data: (a) experimental ultrasound imaging data used in [30] (available at https://www.wisdom.weizmann.ac.il/yonina accessed on 20 December 2021) and (b) complex-valued generic Gaussian data. Note that for the generic data, we set $M = 1024$ and $N = 20$ to match the same dimension as ultrasound data in [30]. Moreover, real and complex valued entries of both low-rank and sparse matrices are generated independently from an i.i.d. Gaussian with zero mean and unit variance. Further, the fixed number of non-zero locations of each $\boldsymbol{S}_i$ are selected uniformly. The rank of each $\boldsymbol{L}_i$ is set as 2, and the number of non-zero elements of each $\boldsymbol{S}_i$ is set as $0.01MN$. We normalized $\boldsymbol{S}_i$ and $\boldsymbol{L}_i$ to have a unit Frobenius norm (i.e., $\|\boldsymbol{L}_i\|_F^2 = \|\boldsymbol{S}_i\|_F^2 = 1$). Further, SNR during training and testing is 20 dB. To have a fair comparison, we use the same number of layers as in CORONA. Both CORONA and our approaches are trained using the Adam with 20 epochs. For CORONA, the same settings as in [30] were used. We utilized the CORONA implementation from the author's website (https://www.wisdom.weizmann.ac.il/yonina accessed on 20 December 2021). Note that experimental ultrasound data in [30] follows the standard RPCA problem: $\boldsymbol{A}_l$, $\boldsymbol{A}_s$ in (1) are equal to identity matrix. Therefore, for comparison, our ADMM-based approach is implemented without linear layers, i.e, all $\boldsymbol{W}_1^t$, $\boldsymbol{W}_2^t$, $\boldsymbol{W}_3^t$, and $\boldsymbol{W}_4^t$ are identity matrices, i.e., aforementioned linear layers are omitted from the DNN. Thus, our proposed approach only learns the thresholding parameters λ_S^t, λ_L^t, γ^t, and ρ^t with a learning rate of $L_p = 5 \times 10^{-4}$. Note that in this setting, our proposed approach is only required to learn the four parameters per layer, i.e., 40 parameters for a DNN with ten layers. However, in CORONA, for a single layer, six convolutional weights matrices and two thresholding parameters have to be learned. Based on convolutional filter sizes given in [30], CORONA with ten layers is required to learn $6 \times \{3 \times (5 \times 5) + 7 \times (3 \times 3)\} + 20 = 848$ parameters compared to 40 parameters in our approach.

First, we compared our proposed approach with CORONA for experimental ultrasound data, and corresponding results are shown in Table 2. The experimental ultrasound data in [30] consists of 2400 training samples and 800 testing samples. For performance comparison, similar to [30], average MSE is utilized as a metric for ultrasound data. Note that for experimental ultrasound data, CORONA shows slightly better performance in recovering $\boldsymbol{S}$, compared to the proposed approaches. For the low-rank matrix $\boldsymbol{L}$ recovery, our proposed approaches and CORONA show similar performance levels. For the sparse matrix $\boldsymbol{S}$ recovery, our proposed approaches show slightly worse performances compared to the CORONA. This is to be expected since, in CORONA, $\ell_{1,2}$-norm minimization is used for $\boldsymbol{S}$, which reflects the row-sparse nature of the experimental ultrasound data. Our approach is formulated for plain unstructured sparsity in the matrix, and it is, therefore, not that optimized for sparsity patterns in experimental ultrasound data. Note that it is also straightforward to modify our ADMM approaches for soft-thresholding related to the $\ell_{1,2}$-norm.

Next, we compared CORONA and our approach for the generic Gaussian data. For the Gaussian data set, we consider 2400 training samples and 1600 testing samples. Here, our proposed approach outperforms the CORONA, as shown in Table 3. This is due to the fact that the data acquisition model given in (27) follows an unstructured sparsity model and does not include convolution operation. Thus, since the generic Gaussian data does not follow the sparsity model as the ultrasound data in [30], the performance of CORONA is degraded compared to our approach. As discussed above, the ultrasound data follows the standard RPCA where there is no compression, i.e., $\boldsymbol{A}_l$, $\boldsymbol{A}_s$ in (1) are equal to identity matrix. In order to evaluate our approach on compressed data, we manually applied the compression on ultrasound data as discussed next.

Table 2. Comparison with CORONA [30] for experimental ultrasound imaging data from [30]. CORONA shows slightly better performance compared to the proposed approaches TRPCA-AT(log) and TRPCA-AT(exp) because CORONA is optimized for the structure of the ultrasound data. However, our approaches are not optimized for the structure of the experimental ultrasound data.

Method	**Average Recovery Error** $= \frac{1}{MNN_s}\sum_{i=1}^{N_s}(\lVert \boldsymbol{X}_i - \hat{\boldsymbol{X}}_i \rVert_F)$	
	Low-Rank Matrix $\boldsymbol{L}$	**Sparse Matrix** $\boldsymbol{S}$
CORONA [30]	3.23×10^{-4}	3.431×10^{-4}
TRPCA-AT(log)	3.26×10^{-4}	6.641×10^{-4}
TRPCA-AT(exp)	3.37×10^{-4}	7.101×10^{-4}
TRPCA-T	9.95×10^{-4}	7.35×10^{-4}

Table 3. Comparison with CORONA [30] for generic Gaussian data. Our proposed approach TRPCA-AT(log) outperforms the CORONA. This is due to the fact that the CORONA is optimized for structured sparsity of ultrasound data, which is not present in generic Gaussian data.

Method	**Average Recovery Error NRMSE** $= \frac{1}{N_s}\sum_{i=1}^{N_s}\left(\lVert \boldsymbol{X}_i - \hat{\boldsymbol{X}}_i \rVert_F \big/ \lVert \boldsymbol{X}_i \rVert_F\right)$	
	Low-Rank Matrix $\boldsymbol{L}$	**Sparse Matrix** $\boldsymbol{S}$
CORONA [30]	4.45×10^{-1}	4.08×10^{-1}
TRPCA-AT(log)	6.56×10^{-2}	3.29×10^{-2}

The received signal matrix of ultrasound data $\boldsymbol{Y} \in \mathbb{C}^{M\times N}$ is given by $\boldsymbol{Y} = \boldsymbol{L} + \boldsymbol{S} + \boldsymbol{N}$, where $\boldsymbol{L}$, $\boldsymbol{S}$, and $\boldsymbol{N}$ are low-rank, sparse, and noise matrices of size $M \times N$, respectively. In ultrasound data, a single measurement consists of twenty frames of size 32×32; this results in $M = 20$ and $N = 1024$. Lets denote the frame size as $m \times n$. In order to evaluate our approach to compressed data, we manually applied the compression on ultrasound data by using a Gaussian matrix $\boldsymbol{A}$ which compresses a 32×32 frame to a 16×16 frame, i.e., 50% compression. In more detail, the matrix $\boldsymbol{A}$ is a linear operator which maps the vector space $\mathbb{C}^{mn}$ to vector space $\mathbb{C}^{k}$. We set $mn = 1024$ and $k = 512$. Now, after the compression, the received signal for a single measurement is given by $\boldsymbol{Y}_{cs} \in \mathbb{C}^{M\times k}$, i.e., 20×512. Here, we consider 1800 training samples and 400 testing samples. We train our proposed approach using the Adam optimizer with 20 epochs with learning rate of 1×10^{-4}. The average normalized RMSEs for the different numbers of layers of the DNN for $k/mn = 50\%$ is shown in Figure 12. As shown in Figure 12, our proposed approach TRPCA-AT(log) outperforms the untrained approach URPCA-T in terms of NRMSE as well as the number of iterations. The proposed approach TRPCA-AT(log) is able to achieve much lower NRMSE by only using 15 layers compared to the 200 iterations in the untrained approach URPCA-T. Therefore, our approach is 13 times faster than the untrained approach.

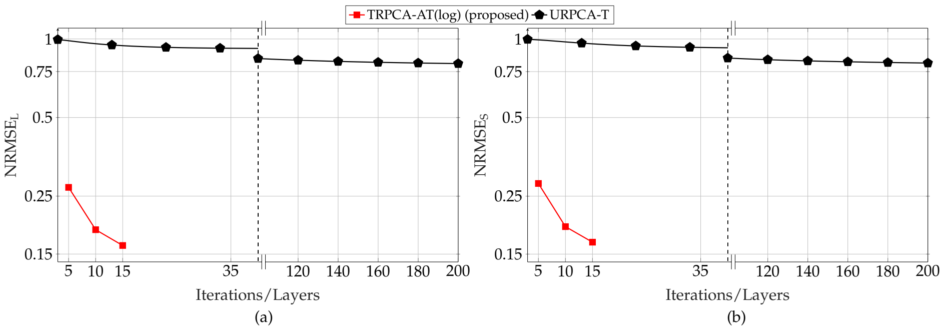


Figure 12. Average NRMSE of low-rank (**a**) and sparsity (**b**) contributions for experimental ultrasounds data with 50% compression ratio for the proposed ADMM-based trained RPCA with adaptive thresholding based on logarithm heuristic (TRPCA-AT(log)) and ADMM-based untrained RPCA with thresholding (URPCA-T).

4.2. SFCW Radar Model

In this subsection, the performance evaluation of the ADMM-based trained RPCA with adaptive thresholding is now performed for the SFCW radar model given in Section 2.1. In the simulations, we set the carrier frequency f_c of 300 GHz and bandwidth B as 5 GHz. Here, we consider two types of simulations: (a) small scale and (b) large scale. For the small scale, we consider $N = M = 30$, i.e., 30 antennas and 30 frequency bands. Both height and length of the layered structure are 0.5 m. In the simulations, we consider six defects, and the scene is partitioned into a 16×16 grid with equal grid size (i.e., $Q = 256$). The grid size is selected according to the Rayleigh resolution of the radar. For the large scale, we consider $N = M = 100$, i.e., 100 antennas and 100 frequency bands. In addition to that, we have increase both height and length of the layered structure to 2.5 m. This results in an increase of the grid size, and the grid size for this scenario is 83×83, (i.e., $Q = 6889$). Moreover, here we consider nine defects in the radar scene.. The inter-antenna spacing is chosen as half of the wavelength of f_c. We consider a single-layered structure, and the distance to the front surface of the layered structure is 1.0 m. Denote the reflection of the layered material structure, noise matrix, and sparse vector for the i-th data sample, given in (7), by $\boldsymbol{Y}_i^l$, $\boldsymbol{Z}_i$ and $\boldsymbol{s}_i\ \forall\ i$, respectively. In the simulations, the signal-to-noise ratio of the i-th data sample for given $\boldsymbol{\Phi}$ and $\boldsymbol{D}$ is defined as $\text{SNR} := \left\|\boldsymbol{\Phi}\text{vec}(\boldsymbol{Y}_i^l) + \boldsymbol{\Phi}\boldsymbol{D}\boldsymbol{s}_i\right\|_2^2 / \|\boldsymbol{\Phi}\text{vec}(\boldsymbol{Z}_i)\|_2^2 = 20$ dB. Here, we set same SNR for all samples. Note that the SFCW data consists of complex numbers, thus, in this work, we implemented the DNN which supports complex numbers by using the PyTorch version 1.8.1 [53]. Here, we initialize $\boldsymbol{W}_1^t = \boldsymbol{A}_l^H$, $\boldsymbol{W}_2^t = \boldsymbol{A}_s$, $\boldsymbol{W}_3^t = \boldsymbol{A}_s^H$, and $\boldsymbol{W}_4^t = \boldsymbol{A}_l$ to mimic the ADMM Algorithm 1.

Interestingly, in contrast to the generic Gaussian model, only learning the λ_S^t and λ_L^t does not achieve satisfactory average NRMSEs of the low-rank and sparse components. Therefore, we enable learning all parameters given in Θ. Further, we notice that the stochastic gradient descent (SGD) [55] performs better than the Adam in learning all the parameters given in Θ together. Therefore, we consider a three-stage training process for better learning. A detailed breakdown of this three-stage training process is given in Appendix A.4.

For defect detection by SFCW radar, we considered a data set of 600 samples. Here, 500 data samples are used for training and validation, and 100 data samples are used for testing. We used Matlab [51] to generate the SFCW data based on (7). First, we present the results related to the small-scale simulations. Next, results related to the large-scale simulations are presented.

4.2.1. SFCW Small-Scale Simulations

Here, we present the results for $M = N = 30$ configuration, i.e., 30 antenna elements and 30 frequency bands. The average normalized RMSEs for the different numbers of layers of the DNN for $K/MN = 20\%$ is shown in Figure 13. The figure shows that the proposed TRPCA-AT outperforms both URPCA-T and the LRPSRC given in (8). Further, in terms of the average RMSE, the TRPCA-AT and TRPCA-T with five layers outperform URPCA-T with 200 iterations. Therefore, as we compare the number of layers of the TRPCA-AT to the number of iterations of the URPCA-T, the TRPCA-AT achieves a $1:40$ improvement for the SFCW radar data, i.e., our proposed approach (TRPCA-AT) is forty times faster than the conventional untrained approach (URPCA-T). Moreover, based on the results shown in Figure 13, the TRPCA-AT shows better performance compared to the TRPCA-T. In addition, note that with 20% compression ratio, the estimation of $\boldsymbol{Y}^l$ and $\boldsymbol{s}$ from $\boldsymbol{y}_{cs}$ in (7) is more challenging. Therefore, the average RMSE of the LRPSRC is higher than 0.5. However, the DNN-based TRPCA-AT is able to achieve average RMSE in the range of 0.1 for both sparse and low-rank components. Since $\boldsymbol{A}_s$ and $\boldsymbol{A}_l$ are unequal in the SFCW radar model, we did not consider the CRB benchmark given in (A4).

Next, to further illustrate defect detection, images of the recovered defects are formed, as shown in Figure 14 for a single data sample. As a benchmark, we consider the state-of-the-art subspace projection (SP) [9] method with the full data set, $K/MN = 100\%$. Further, for SP, it is assumed that the number of defects is known. Figure 14a shows the actual defect locations. The recovered locations of the defects for the ADMM-based trained RPCA TRPCA-AT(log), TRPCA-AT(exp), TRPCA-T, LRPSRC, ADMM-based untrained RPCA with thresholding URPCA-T, and the SP are shown in Figure 14b–g, respectively. It can be seen that the proposed TRPCA-AT approaches are able to identify all six defects. Further, the proposed TRPCA-AT approaches are even able to estimate amplitudes of the recovered defects (vector $\boldsymbol{s}$) closer to the actual defects. Therefore, the proposed TRPCA-AT approaches outperform state-of-the-art SP even with 20% of data.

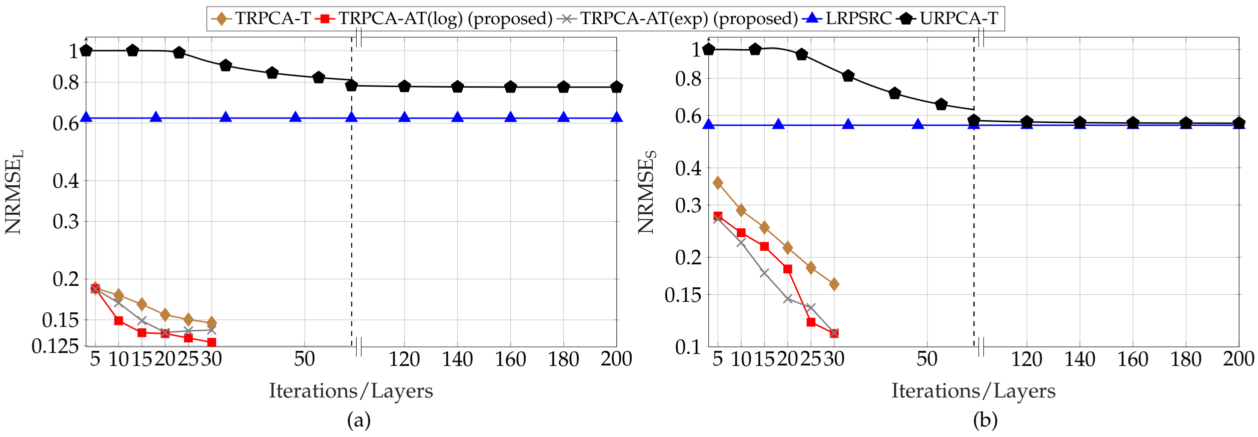


Figure 13. Average NRMSE of low-rank (**a**) and sparsity (**b**) contributions of SFCW Radar model for $K/MN = 20\%$ with $M = N = 30$ for ADMM-based trained RPCA with thresholding (TRPCA-T), proposed ADMM-based trained RPCA with adaptive thresholding based on logarithm heuristic (TRPCA-AT(log)), proposed ADMM-based trained RPCA with adaptive thresholding based on exponential heuristic (TRPCA-AT(exp)), low-rank plus sparse recovery with convex relaxation (LRPSRC), and ADMM-based untrained RPCA with thresholding (URPCA-T).

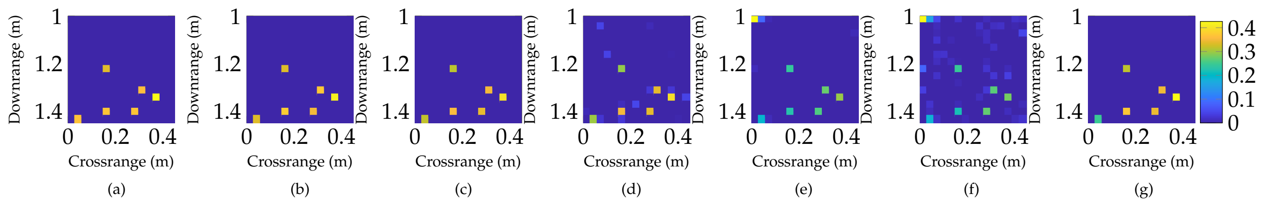


Figure 14. Object recovery for a single case with compression ratio $K/MN = 20\%$ with $M = N = 30$. (**a**) Ground-truth, (**b**) TRPCA-AT(log) (proposed), (**c**) TRPCA-AT(exp) (proposed), (**d**) TRPCA-T, (**e**) LRPSRC, (**f**) URPCA-T, and (**g**) SP with 100% of data. The proposed TRPCA-AT approaches are able to identify all six objects successfully by only utilizing 20% of data compared to the unweighted approach TRPCA-T.

4.2.2. SFCW Large-Scale Simulations

Here, we present the results for $M = N = 100$ configuration, i.e., 100 antenna elements and 100 frequency bands. In small-scale simulations, our proposed approaches TRPCA-AT(log) and TRPCA-AT(exp) achieve similar results, therefore, we chose one of them for the large-scale simulations to compare with the untrained approach URPCA-T.

The average normalized RMSEs for the different numbers of layers of the DNN for $K/MN = 20\%$ are shown in Figure 15. The figure shows that the proposed TRPCA-AT(log) outperforms the untrained approach URPCA-T. Further, in terms of the average RMSE, the proposed TRPCA-AT(log) with five layers outperforms the untrained approach URPCA-T with 200 iterations. Therefore, our proposed approach (TRPCA-AT) is forty times faster than the conventional untrained approach (URPCA-T). In addition, note that with 20% compression ratio, the estimations of low-rank and sparse matrices are more challenging. However, the DNN-based TRPCA-AT(log) is able to achieve a lower average NRMSE by parameter tuning compared to the conventional untrained approach (URPCA-T) with the fewer numbers of iterations.

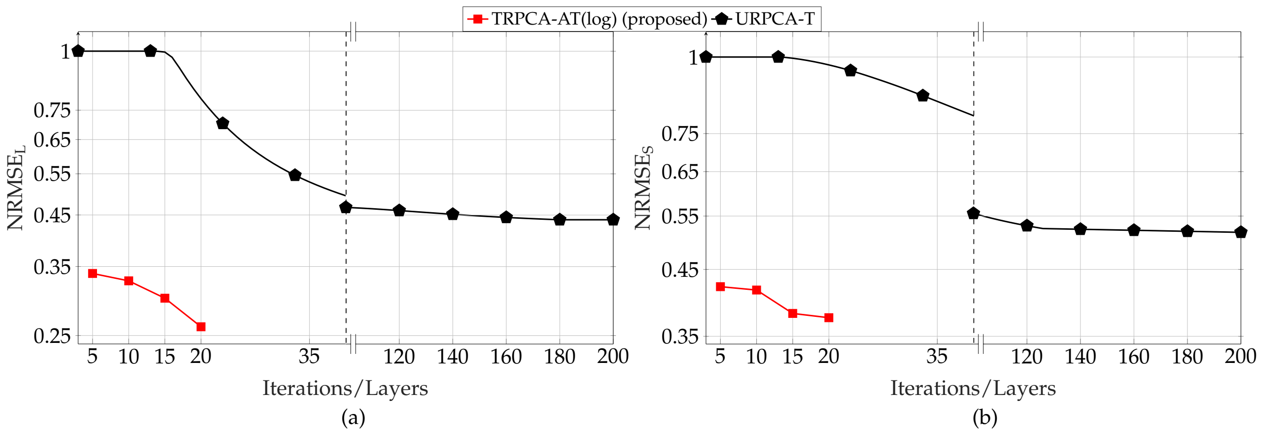


Figure 15. Average NRMSE of low-rank (**a**) and sparsity (**b**) contributions of SFCW Radar model for $K/MN = 20\%$ with $M = N = 100$ for the proposed ADMM-based trained RPCA with adaptive thresholding based on logarithm heuristic (TRPCA-AT(log)) and ADMM-based untrained RPCA with thresholding (URPCA-T).

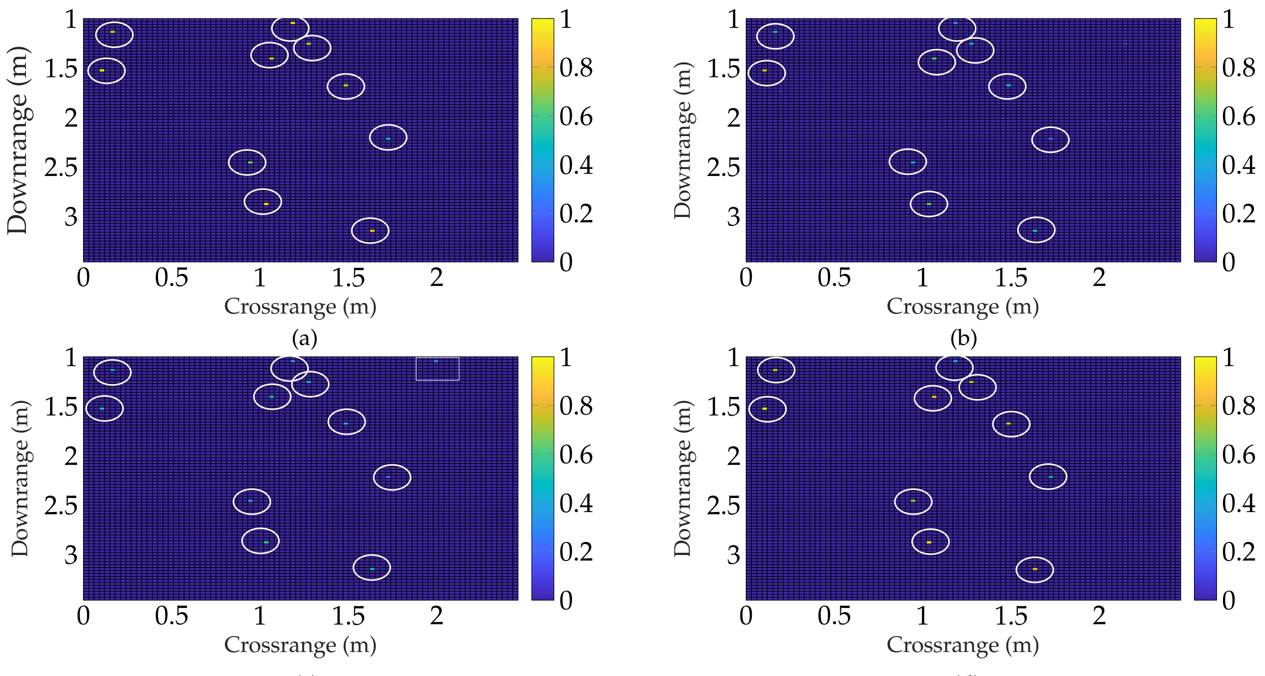
The recovered sparse matrix $\boldsymbol{S}$ contains all the complex reflection coefficients (α_p) of the defects. Therefore, to further illustrate the defect detectability, we show the total power of the recovered sparse matrix. Here, we consider two metrics: (a) total power of the true locations of the defects and (b) total power of the false detection. Here, the power of the false detection is the power of elements in the sparse matrix $\boldsymbol{S}$ that does not belong to the true locations of the defects. In addition, the total power of the true locations of the defects is the power of elements in the sparse matrix $\boldsymbol{S}$ that belong to the true locations of the defects. These results are shown in Table 4, and it is observed that the proposed approach

TRPCA-AT(log) is able to achieve much higher total power of the true locations of the defects than the untrained approach (URPCA-T). Further, it is observed that our approach achieves lower power in false detection, too.

Table 4. Total power of the true defects and false detection for 100 simulations with $M = N = 100$. Here, the total true power of the defects for all simulations is 100.

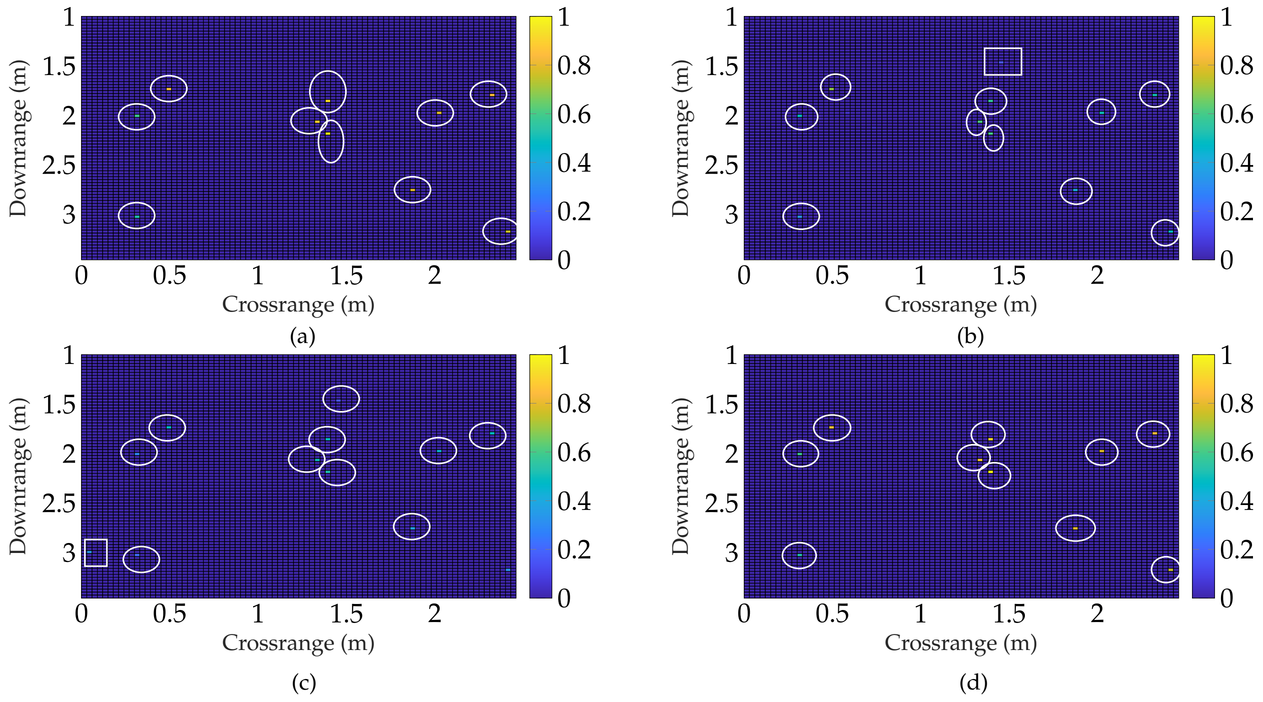
Method	**Total Power** $= \sum_{i=1}^{N_s} \|\|S_i\|\|_F^2$	
	True Locations of the Defects	**False Detection**
URPCA-T	27.8565	2.0247
TRPCA-AT(log)	44.727	1.3537

Next, to illustrate defect detection, images of the recovered defects are formed for two scenarios as shown in Figure 16. In Figure 16, (Aa) and (Ba) show the actual locations of the defects. The recovered locations of the defects by the proposed ADMM-based trained RPCA TRPCA-AT(log), ADMM-based untrained RPCA with thresholding URPCA-T, and the classical subspace projection (SP) are shown in Figure 16b–d, respectively. It can be seen that the proposed TRPCA-AT(log) approach is able to identify all defects while only utilizing 20% of data. Further, the proposed TRPCA-AT(log) approach has fewer false detections than the untrained RPCA with thresholding (URPCA-T) approach for these two scenarios. It is worth noticing that the conventional SP approach utilizes 100% of data, and for the SP, it is required to know the number of the defects prior.



(**A**) Object recovery in single case-scenario 1.

Figure 16. *Cont.*



(**B**) Object recovery in single case-scenario 2.

Figure 16. Object recovery for a single case with compression ratio $K/MN = 20\%$ with $M = N = 100$. (**a**) Ground-truth, (**b**) TRPCA-AT(log) (proposed), (**c**) URPCA-T and (**d**) SP with 100% of data. The proposed TRPCA-AT(log) approach is able to identify all nine objects successfully by only utilizing 20% of data. True locations of the objects are shown inside ellipses and false detections are shown in squares.

5. Conclusions

This paper presents a deep learning-based parameter tuning for the low-rank plus sparse recovery (RPCA). To this end, an iterative algorithm was developed based on ADMM to estimate the low-rank and sparse contributions with iterative reweighted nuclear and ℓ_1-norm minimization. Next, to improve the accuracies of the recovered low-rank and sparse components and the speed of convergence of the algorithm, we proposed a DNN to tune the parameters of the iterative algorithm, i.e., algorithm unrolling/unfolding. Our proposed approach was evaluated for two types of data. As a standard benchmark, a generic Gaussian data acquisition model was used, and for practical application, the defect detection by SFCW radar from compressive measurements was considered. For both cases, our proposed approach performed substantially better compared to the untrained iterative algorithms in terms of low-rank and sparse recovery and convergence speed. In particular, for compression ratios (K/MN) 50% and 25%, our proposed approach was 15 and 7.5 times faster than the untrained algorithm. In addition to that, we have compared our proposed approach with the state-of-the-art RPCA unfolding approach (CORONA). Our approach achieveed a similar performance level as CORONA for experimental ultrasound imaging data, and our approach outperformed CORONA for generic Gaussian data. Moreover, we analyzed the robustness of our approach for testing signal-to-noise ratio (SNR) uncertainty and the deviation in the measurement matrices ($\boldsymbol{A}_l$, $\boldsymbol{A}_s$). It was observed that the knowledge of testing SNR is an important factor, and for unknown testing SNR, it is better to train the DNN with SNR like 5 dB. Furthermore, the robust training approach (training with the distribution of deviation) decreased the impact of the deviation in the measurement matrices on the performance. In this work, we considered a model-based unfolding approach where unfolded DNN strictly follows the structure of the optimization steps/rules. As possible future work, it would be interesting to study a model-free unfolding approach which is able to learn new optimization steps/rules from data. Moreover, validation of our approach for experimental/real measurements based on defect detection is subject to future work.

Author Contributions: Conceptualization, P.J. and A.S.; methodology, U.S.K.P.M.T.; software, U.S.K.P.M.T.; validation, U.S.K.P.M.T., P.J. and A.S.; formal analysis, U.S.K.P.M.T.; investigation, U.S.K.P.M.T.; resources, U.S.K.P.M.T., P.J. and A.S.; data curation, U.S.K.P.M.T.; writing—original draft preparation, U.S.K.P.M.T.; writing—review and editing, P.J. and A.S.; visualization, U.S.K.P.M.T.; supervision, P.J. and A.S.; funding acquisition, P.J. and A.S. All authors have read and agreed to the published version of the manuscript.

Funding: The work of U.S.K.P.M.T. and A.S. is funded by the German Research Foundation ("Deutsche Forschungsgemeinschaft") (DFG) under Project-ID 287022738 TRR 196 for Project S02. The work of Jung, P. is funded by the German Federal Ministry of Education and Research (BMBF) in the framework of the international future AI lab "AI4EO–Artificial Intelligence for Earth Observation: Reasoning, Uncertainties, Ethics and Beyond" [grant number 01DD20001].

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data can be provided by the author U.S.K.P.M.T. upon reasonable request.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations and variables are used in this manuscript:

Abbreviations	
ADMM	Alternating direction method of multipliers
APG	Accelerated proximal gradient
BS	Background subtraction
CORONA	Convolutional robust principal component analysis
CRB	Cramér–Rao bound
CS	Compressive sensing
DNN	Deep neural network
EM	Electromagnetic
FISTA	Fast iterative soft-thresholding algorithm
GHz	Gigahertz
ISTA	Iterative soft-thresholding algorithm
LISTA	Learned iterative soft-thresholding algorithm
LRPSRC	Low-rank plus sparse recovery with convex relaxation
MSE	Mean squared error
NRMSE	Normalized average root mean squared error
MIMO	Multiple-input and multiple-output
RADAR	Radio detection and ranging
RF	Radio frequency
RPCA	Robust principal component analysis
SFCW	Stepped-frequency continuous wave
SGD	Stochastic gradient descent
SNR	Signal-to-noise ratio
SP	Subspace projection
SVD	Singular value decomposition
TRPCA-AT(exp)	Trained RPCA with adaptive thresholding based on exponential heuristic
TRPCA-AT(log)	Trained RPCA with adaptive thresholding based on logarithm heuristic
TRPCA-T	Trained RPCA with thresholding
URPCA-T	Untrained RPCA with thresholding

Variable	
$\gamma \in \mathbb{R}$	A positive constant used in decay functions
$\boldsymbol{u} \in \mathbb{C}^K$	ADMM auxiliary variables
$\rho \in \mathbb{R}$	ADMM penalty factor
$B \in \mathbb{R}$	Bandwidth of the SFCW radar
$f_c \in \mathbb{R}$	Carrier frequency of the SFCW radar
$\alpha_p \in \mathbb{C}$	Complex reflectivity coefficient of the p-th defect
$\alpha_l \in \mathbb{C}$	Complex reflectivity of the layered material structure
$\boldsymbol{A}_l,\ \boldsymbol{A}_s,\ \boldsymbol{A} \in \mathbb{C}^{K\times MN}$	Compression operators/measurement matrices
$K/MN \in \mathbb{R}$	Compression ratio
$L_p \in \mathbb{R}$	Learning rate of the parameters $(\lambda_L^t, \gamma^t, \rho^t)$ of the DNN
$L_w \in \mathbb{R}$	Learning rate of the weights $(\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3, \boldsymbol{W}_4)$ of the linear layers of the DNN
$\boldsymbol{L} \in \mathbb{C}^{M\times N}$	Low-rank matrix
$f_n \in \mathbb{R}$	$n-$th frequency band
$P \in \mathbb{R}$	Number of defects
$E_p \in \mathbb{R}$	Number of epochs
$N \in \mathbb{R}$	Number of frequency bands in SFCW radar system
$N_s \in \mathbb{R}$	Number of testing samples
$T_s \in \mathbb{R}$	Number of training samples
$M \in \mathbb{R}$	Number of transceivers in SFCW radar system
Θ	Parameters set that DNN learns $(\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3, \boldsymbol{W}_4, \lambda_S^t, \lambda_L^t, \gamma^t, \rho^t)$
$r \in \mathbb{R}$	Rank of the low-rank matrix $\boldsymbol{L}$
$\boldsymbol{Y} \in \mathbb{C}^{M\times N}$	Received signal matrix corresponding to all M transceivers and N frequencies
$\boldsymbol{y},\ \boldsymbol{y}_{cs} \in \mathbb{C}^K$	Reduced received data vector
$\boldsymbol{Y}^d \in \mathbb{C}^{M\times N}$	Reflection of the defects corresponding to all M transceivers and N frequencies
$\boldsymbol{Y}^l \in \mathbb{C}^{M\times N}$	Reflection of the layered material structure corresponding to all M transceivers and N frequencies
$\lambda_l,\ \lambda_s \in \mathbb{R}$	Regularization parameters
$\boldsymbol{\Phi} \in \mathbb{R}^{K\times MN}$	Selection matrix
$\lambda_L \in \mathbb{R}$	Singular value soft-thresholding parameter
$\boldsymbol{\sigma}(\boldsymbol{L}) \in \mathbb{R}^M$	Singular values of $\boldsymbol{L}$
$Q \in \mathbb{R}$	Size of the rectangular grid of the radar scene
$\lambda_S \in \mathbb{R}$	Soft-thresholding parameter
$\boldsymbol{S} \in \mathbb{C}^{M\times N}$	Sparse matrix
$\boldsymbol{D} \in \mathbb{C}^{MN\times Q}$	The grid matrix of the radar scene
$\boldsymbol{s} \in \mathbb{C}^{Q\times 1}$	Vector that contains all the α_p values of the defects
$\boldsymbol{\lambda}_{LT}^t \in \mathbb{R}^M$	Vector that contains all the singular-value threshold values for $\boldsymbol{L}$ in the $t+1$-th iteration
$\boldsymbol{\lambda}_{ST}^t \in \mathbb{R}^{MN}$	Vector that contains all the soft-threshold values for $\boldsymbol{S}$ in the $t+1$-th iteration
$\boldsymbol{W}_1^t, \boldsymbol{W}_2^t, \boldsymbol{W}_3^t, \boldsymbol{W}_4^t$	Weights of the t-th layer of the DNN

Appendix A

Appendix A.1. Element-Wise Soft-Thresholding and Singular Value Soft-Thresholding

The element-wise or adaptive soft-thresholding operation is applied to each element of the vector or matrix individually. Here, the main difference of the element-wise or adaptive soft-thresholding compared to the non-adaptive soft-thresholding is that in the element-wise or adaptive soft-thresholding, threshold value is different from one element to another element. However, in non-adaptive or standard soft-thresholding, the same threshold value is applied to all elements. Let us consider a matrix $\boldsymbol{X} \in \mathbb{C}^{M\times N}$. To this end, the value after the element-wise or adaptive soft-thresholding $\boldsymbol{X}^{\text{st}}$ is given by

$$\boldsymbol{X}^{\text{st}} = \text{ST}_{\boldsymbol{\lambda}_{ST}}(\boldsymbol{X}). \tag{A1}$$

Here, $\boldsymbol{\lambda}_{ST} = [\lambda_{ST}^{1,1}, \ldots, \lambda_{ST}^{m,n}, \ldots, \lambda_{ST}^{M,N}]$ contains the element-wise thresholds for $\boldsymbol{X}$. Now, the m-th row and n-th column element of $\boldsymbol{X}^{\text{st}}$ ($x_{m,n}^{\text{st}}$) is given by

$$x_{m,n}^{\text{st}} = \text{ST}_{\lambda_{ST}^{m,n}}(x_{m,n}) = \exp(j\theta)\,\max\big(|x_{m,n}| - \lambda_{ST}^{m,n}, 0\big). \tag{A2}$$

The m-th row and n-th column element of $\boldsymbol{X}$ is $x_{m,n}$. Further, θ is the phase angle of the $x_{m,n}$ in radians.

In the element-wise or adaptive singular value soft-thresholding, the same concept is applied to the singular values of the matrix as discussed next. The singular value decomposition (SVD) of $\boldsymbol{X} \in \mathbb{C}^{M\times N}$ with $M \le N$ is given as $\boldsymbol{X} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^H$. Here, $\mathbf{U} \in \mathbb{C}^{M\times M}$ and $\mathbf{V} \in \mathbb{C}^{N\times N}$ are the matrices of the left and right singular vectors. $\boldsymbol{\Lambda} \in \mathbb{R}^{M\times N}$ is a rectangular diagonal matrix with $\boldsymbol{\sigma}(\boldsymbol{X}) = [\sigma_1, \ldots \sigma_m, \ldots, \sigma_M]$ on the diagonal and zeros elsewhere. Next, the value after the element-wise or adaptive singular value soft-thresholding $\boldsymbol{X}^{\text{svt}}$ is given by

$$\boldsymbol{X}^{\text{svt}} = \text{SVT}_{\boldsymbol{\lambda}_{LT}}(\boldsymbol{X}) = \mathbf{U}\text{diag}\big(\text{ST}_{\boldsymbol{\lambda}_{LT}}(\boldsymbol{\sigma}(\boldsymbol{X}))\big)\mathbf{V}^H. \tag{A3}$$

Note that diag$(\cdot)$ takes a vector and returns the corresponding diagonal matrix. Here, $\boldsymbol{\lambda}_{LT} = [\lambda_{LT}^1, \ldots, \lambda_{LT}^m, \ldots \lambda_{LT}^M]$ contains the different thresholds for each singular value of the $\boldsymbol{X}$. Next, we briefly discuss the relationship between the adaptive and non-adaptive/non-uniform soft-thresholding. Interestingly, as given in [48], the non-uniform soft-thresholding of a vector $\boldsymbol{a} \in \mathbb{R}^M$ is equivalent to the uniform soft-thresholding (same threshold for all elements) of another vector $\boldsymbol{b} \in \mathbb{R}^M$. Here, $\boldsymbol{b} = \text{sign}(\boldsymbol{a}) \odot (|\boldsymbol{a}| + w_0\mathbf{1}_M - \boldsymbol{w})$. $\boldsymbol{w} \in \mathbb{R}^M$ is the non-negative weight vector, and equivalent uniform soft-thresholding value is given by $w_0 = \max(\boldsymbol{w})$. A similar statement is valid for element-wise singular value soft-thresholding, and more detail can be found in [48]. Next, we discuss the computation complexity of the proposed approach.

Appendix A.2. Computation Complexity

The computational complexity of the proposed DNN is discussed in this subsection. Note that the t-th iteration of Algorithm 1 is shown in Figure 2. Here, a single layer of the DNN consists of four dense linear layers, and their weight matrices are given by $\boldsymbol{W}_1^t$, $\boldsymbol{W}_3^t \in \mathbb{C}^{K\times MN}$ and $\boldsymbol{W}_2^t$, $\boldsymbol{W}_4^t \in \mathbb{C}^{MN\times K}$. In the feed-forward propagation, data propagation is given in Equations (19)–(21). Now, for T_s, number of training samples and E_p, number of epochs, the computational complexity of the feed-forward propagation is $\mathcal{O}(6T_sE_p(MNK) + \mathcal{O}(T_sE_p(M^2N + MN^2 + N^3)) + \mathcal{O}(T_sE_p(M^2N + N^2M)) \approx \mathcal{O}(T_sE_p(MNK + M^2N + N^2M + N^3))$. When $M = N$, the computational complexity of the feed-forward propagation is given by $\mathcal{O}(T_sE_p(N^2K + N^3))$. Here, $\mathcal{O}(\cdot)$ is the Big O notation for asymptotic computational complexity analysis [49].

For the back propagation, the computational complexity of the linear layers is given by $\mathcal{O}(6T_sE_p(MNK)$, and for the back propagation through SVD, it is given by $\mathcal{O}(T_sE_p(M^2N + MN^2 + N^3)) + \mathcal{O}(T_sE_p(M^2N + N^2M))$. Hence, the training complexity of the DNN is the addition of the feed-forward and the back propagation complexities. Now, for $M = N$, the training complexity of the DNN is given by $\mathcal{O}(2T_sE_p(N^2K + N^3)) \approx \mathcal{O}(T_sE_p(N^2K + N^3))$. This computational complexity corresponds to the single iteration of the Algorithm 1. Now, for T iterations/layers, the training computational complexity is given by $\mathcal{O}(TT_sE_p(N^2K + N^3))$. The testing computational complexity is the feed-forward propagation complexity of data through the DNN. It is given by $\mathcal{O}(TN_s(N^2K + N^3))$; here N_s is the number of testing samples. Next, for completeness, we discuss the recovery guarantees of the standard RPCA problem in the following.

Appendix A.3. Cramér–Rao Bound (CRB) and Recovery Guarantees of RPCA

We briefly discuss recovery guarantees of the standard RPCA problem ($\boldsymbol{A}_l = \boldsymbol{A}_s = \boldsymbol{I}$ in (1)) in this subsection. We consider the above scenario because it is well studied in the literature and well understood, i.e., when the separation of the low-rank matrix $\boldsymbol{L}$ and sparse matrix $\boldsymbol{S}$ is possible. Based on [17], informally, if $\boldsymbol{L}$ is sufficiently low-rank but not sparse and $\boldsymbol{S}$ is sufficiently sparse but not low-rank, the matrices $\boldsymbol{L}$ and $\boldsymbol{S}$ can be estimated exactly with a high probability of success. Here, to solve the RPCA problem, convex relaxations of sparsity and rank in terms of ℓ_1-norm of a matrix and nuclear norm of a matrix as given in (8) is utilized with $\lambda_l = 1$ and $\lambda_s = 1/\sqrt{\max(M,N)}$ [17]. Let $\bar{K} = \max(N,M)$, $\underline{K} = \min(N,M)$ and positive constants c_o, p_s and p_r. We consider the following theorem from [17].

Theorem A1. *It is possible to recover* $\boldsymbol{L}$ *and* $\boldsymbol{S}$ *from noiseless observation* $\boldsymbol{L}+\boldsymbol{S}$ *with a probability at least* $1 - c_o\bar{K}^{-10}$ *when* $\mathrm{rank}(\boldsymbol{L}) \leq p_r\underline{K}(\mu)^{-1}(log(\bar{K}))^{-2}$ *and* $\|\boldsymbol{S}\|_0 \leq p_s\bar{K}\underline{K}$.

Note that $\bar{K}\underline{K} = MN$. Moreover, μ is an incoherence condition parameter of the low-rank matrix $\boldsymbol{L}$ [17]. As discussed in [56–58], when μ is small, the singular value vector of the matrix $\boldsymbol{L}$ is spread out.

Further results are known in terms of the Cramér–Rao bound (CRB) for the RPCA given in [18]. Here, similar to [17], the RPCA problem is solved using the ℓ_1-norm and nuclear norm minimization. Let the received data vector $\boldsymbol{y}$ in (1) follows $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{A}\mathrm{vec}(\boldsymbol{L}+\boldsymbol{S}), \sigma^2\boldsymbol{I}_K)$. Here, the matrix $\boldsymbol{A} = \boldsymbol{A}_l = \boldsymbol{A}_s$ is assumed to be a selection operator which selects a uniformly random subset of size K from MN entries. Now, the CRB of unbiased estimation for $\boldsymbol{L}$ and $\boldsymbol{S}$ is bounded by [18]. Since this is the closest matching CRB to our model given in (1), we have considered this formulation as a benchmark:

$$\left\{s_0 - N_0 + \frac{1}{3}\frac{KN_0}{K-s_0} + \frac{2}{3}\frac{MNN_0}{K-s_0}\right\}\sigma^2 \leq \mathrm{CRB}(\boldsymbol{L},\boldsymbol{S}) \leq \left\{s_0 - N_0 + \frac{3KN_0}{K-s_0} + \frac{2MNN_0}{K-s_0}\right\}\sigma^2, \tag{A4}$$

with a probability higher than $1 - 10\,\exp(-c_1/\epsilon_1^2)$, where $\|\mathrm{vec}(\hat{\boldsymbol{L}} - \boldsymbol{L})\|_2^2 + \|\mathrm{vec}(\hat{\boldsymbol{S}} - \boldsymbol{S})\|_2^2 \leq \epsilon_1$, and estimated low-rank and sparse matrices are given by $\hat{\boldsymbol{L}}$ and $\hat{\boldsymbol{S}}$, respectively. Here, $N_0 = (M+N)r - r^2$ with $\mathrm{rank}(\boldsymbol{L}) \leq r$ and $\|\boldsymbol{S}\|_0 \leq s_0$. As discussed in [18], when $M = N$ and $\boldsymbol{A}$ is an identity matrix, r and s_0 are given by $\mathrm{rank}(\boldsymbol{L}) \leq r = p_rN(\log(N))^{-5}$ and $\|\boldsymbol{S}\|_0 \leq s_0 = p_sN^2$, respectively.

Appendix A.4. Three-Stage Training Process for SFCW Data

Here, we describe the three-stage training process that was used to train the DNN for SFCW data. In the first stage, we only learn the λ_S^t and λ_L^t for 50 epochs using Adam. In the second stage, we learn all parameters given in Θ using SGD optimizer for 50 epochs. Finally, we only learn the λ_S^t and λ_L^t for 15 epochs using Adam. In addition, we slightly adjusted the learning rate as the number of layers of the DNN increased. For the first stage, we employed learning rates 1×10^{-1}, 5×10^{-2}, and 5×10^{-3} for the DNN with 5/10, 15/20, and 25/30 layers, respectively. Here, the only exception is the TRPCA-T. For the TRPCA-T, we considered a learning rate of 5×10^{-2} for the DNN with 25/30 layers as well. This is due to the fact that the non-adaptive thresholding based TRPCA-T is less sensitive to the change of parameters compared to the adaptive thresholding based TRPCA-AT. For the second and third stages, we employed learning rates 1×10^{-3}, 2.5×10^{-4} for all layers combinations of the DNN, respectively. The main reason to consider the third training stage is that it achieves higher performance gains with respect to continuation of the second training stage for another 15 epochs. In addition, note that there is a specific reason to use the first stage without directly using the second stage. This is due to the imbalance of $\boldsymbol{A}_s$ and $\boldsymbol{A}_l$ of the SFCW model compared to the generic Gaussian model. Note that $\boldsymbol{A}_s = \boldsymbol{\Phi}\boldsymbol{D}$ and $\boldsymbol{A}_l = \boldsymbol{\Phi}$ where $\boldsymbol{\Phi}$ is the selection matrix. This matrix has a single non-zero element of value 1 in each row to indicate the selected frequency of a particular antenna if that antenna is selected. However, $\boldsymbol{A}_s = \boldsymbol{\Phi}\boldsymbol{D}$ is a combination of the selection matrix and $\boldsymbol{D}$, where $\boldsymbol{D}$ is generated based on the time delays of the grid (as described in Section 2.1). Therefore, $\boldsymbol{A}_s$ has a very specific structure compared to $\boldsymbol{A}_l$ and is more difficult to learn. This results in an imbalance in the training phase if we directly start with the stage two, since the NRMSE of the low-rank component tends to be much smaller compared to the sparse component.

References

1. Tang, V.H.; Bouzerdoum, A.; Phung, S.L. Multipolarization Through-Wall Radar Imaging Using Low-Rank and Jointly-Sparse Representations. *IEEE Trans. Image Process.* **2018**, *27*, 1763–1776. [CrossRef] [PubMed]
2. Kariminezhad, A.; Sezgin, A. Spatio-Temporal Waveform Design in Active Sensing Systems with Multilayer Targets. In Proceedings of the 2019 27th European Signal Processing Conference (EUSIPCO), A Coruna, Spain, 2–6 September 2019; pp. 1–5. [CrossRef]
3. Miriya Thanthrige, U.S.K.P.; Barowski, J.; Rolfes, I.; Erni, D.; Kaiser, T.; Sezgin, A. Characterization of Dielectric Materials by Sparse Signal Processing With Iterative Dictionary Updates. *IEEE Sens. Lett.* **2020**, *4*, 1–4. [CrossRef]
4. Chopard, A.; Sleiman, J.B.; Cassar, Q.; Guillet, J.; Pan, M.; Perraud, J.; Susset, A.; Mounaix, P. Terahertz waves for contactless control and imaging in aeronautics industry. *NDT E Int.* **2021**, *122*, 102473. [CrossRef]
5. Zahran, O.; Kasban, H.; El-Kordy, M.; Abd El-Samie, F. Automatic weld defect identification from radiographic images. *NDT E Int.* **2013**, *57*, 26–35. [CrossRef]
6. Stoik, C.D.; Bohn, M.J.; Blackshire, J.L. Nondestructive evaluation of aircraft composites using transmissive Terahertz time domain spectroscopy. *Opt. Express* **2008**, *16*, 17039–17051. [CrossRef]
7. Unnikrishnakurup, S.; Dash, J.; Ray, S.; Pesala, B.; Balasubramaniam, K. Nondestructive evaluation of thermal barrier coating thickness degradation using pulsed IR thermography and THz-TDS measurements: A comparative study. *NDT E Int.* **2020**, *116*, 102367. [CrossRef]
8. Huang, Q.; Qu, L.; Wu, B.; Fang, G. UWB through-wall imaging based on compressive sensing. *IEEE Trans. Geosci. Remote Sens.* **2010**, *48*, 1408–1415. [CrossRef]
9. Khan, U.S.; Al-Nuaimy, W. Background removal from GPR data using eigenvalues. In Proceedings of the XIII Int. Conf. on Ground Penetrating Radar, Lecce, Italy, 21–25 June 2010; pp. 1–5. [CrossRef]
10. Sánchez-Pastor, J.; Miriya Thanthrige, U.S.; Ilgac, F.; Jiménez-Sáez, A.; Jung, P.; Sezgin, A.; Jakoby, R. Clutter Suppression for Indoor Self-Localization Systems by Iteratively Reweighted Low-Rank Plus Sparse Recovery. *Sensors* **2021**, *21*, 6842. [CrossRef]
11. Qiao, Z.; Elhattab, A.; Shu, X.; He, C. A second-order stochastic resonance method enhanced by fractional-order derivative for mechanical fault detection. *Nonlinear Dyn.* **2021**, *106*, 707–723. [CrossRef]
12. Qiao, Z.; Liu, J.; Xu, X.; Yin, A.; Shu, X. Nonlinear resonance decomposition for weak signal detection. *Rev. Sci. Instrum.* **2021**, *92*, 105102. [CrossRef]
13. Yun, X.; Mei, X.; Jiang, G. Time-delayed feedback stochastic resonance enhanced minimum entropy deconvolution for weak fault detection of rolling element bearings. *Chin. J. Phys.* **2022**, *76*, 1–13. [CrossRef]
14. Civera, M.; Surace, C. A comparative analysis of signal decomposition techniques for structural health monitoring on an experimental benchmark. *Sensors* **2021**, *21*, 1825. [CrossRef] [PubMed]
15. Jahromi, M.G.; Parsaei, H.; Zamani, A.; Stashuk, D.W. Cross comparison of motor unit potential features used in EMG signal decomposition. *IEEE Trans. Neural Syst. Rehabil. Eng.* **2018**, *26*, 1017–1025. [CrossRef] [PubMed]
16. Donoho, D.L. Compressed sensing. *IEEE Trans. Inf. Theory* **2006**, *52*, 1289–1306. [CrossRef]
17. Candès, E.J.; Li, X.; Ma, Y.; Wright, J. Robust principal component analysis? *J. ACM* **2011**, *58*, 1–37. [CrossRef]
18. Tang, G.; Nehorai, A. Constrained Cramér–Rao bound on robust principal component analysis. *IEEE Trans. Signal Process.* **2011**, *59*, 5070–5076. [CrossRef]
19. Bruckstein, A.M.; Donoho, D.L.; Elad, M. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM Rev.* **2009**, *51*, 34–81. [CrossRef]
20. Cai, J.F.; Candès, E.J.; Shen, Z. A singular value thresholding algorithm for matrix completion. *SIAM J. Optim.* **2010**, *20*, 1956–1982. [CrossRef]
21. Fazel, M.; Hindi, H.; Boyd, S.P. A rank minimization heuristic with application to minimum order system approximation. In Proceedings of the 2001 American Control Conference, Arlington, VA, USA, 25–27 June 2001; Volume 6, pp. 4734–4739. [CrossRef]
22. Gu, S.; Xie, Q.; Meng, D.; Zuo, W.; Feng, X.; Zhang, L. Weighted Nuclear norm minimization and its applications to low level vision. *Int. J. Comput. Vis.* **2017**, *121*, 183–208. [CrossRef]
23. Candes, E.J.; Wakin, M.B.; Boyd, S.P. Enhancing sparsity by reweighted ℓ_1 minimization. *J. Fourier Anal. Appl.* **2008**, *14*, 877–905. [CrossRef]
24. Daubechies, I.; DeVore, R.; Fornasier, M.; Güntürk, C.S. Iteratively reweighted least squares minimization for sparse recovery. *Commun. Pure Appl. Math. J. Issued Courant Inst. Math. Sci.* **2010**, *63*, 1–38. [CrossRef]
25. Mohan, K.; Fazel, M. Reweighted Nuclear norm minimization with application to system identification. In Proceedings of the 2010 American Control Conference, Baltimore, MD, USA, 30 June–2 July 2010; pp. 2953–2959. [CrossRef]
26. Zhao, Y.B.; Li, D. Reweighted ℓ_1-Minimization for Sparse Solutions to Underdetermined Linear Systems. *SIAM J. Optim.* **2012**, *22*, 1065–1088. [CrossRef]
27. Yuan, X.; Yang, J. Sparse and low-rank matrix decomposition via alternating direction methods. *Preprint* **2009**, *12*. Available online: https://www.google.com.hk/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwiu0vHD_ZT3AhVMa94KHZKYCekQFnoECAMQAQ&url=http%3A%2F%2Fwww.optimization-online.org%2FDB_FILE%2F2009%2F11%2F2447.pdf&usg=AOvVaw3_eiF4RSDg53xlwdI7C6sF (accessed on 10 January 2022).

28. Lin, Z.; Chen, M.; Ma, Y. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *arXiv* **2010**, arXiv:1009.5055.
29. Lin, Z.; Ganesh, A.; Wright, J.; Wu, L.; Chen, M.; Ma, Y. Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix (Report no. UILU-ENG-09-2214, DC-246). In *Coordinated Science Laboratory*; 2009. Available online: https://www.ideals.illinois.edu/bitstream/handle/2142/74352/B40-DC_246.pdf?sequence=2 (accessed on 10 January 2022).
30. Solomon, O.; Cohen, R.; Zhang, Y.; Yang, Y.; He, Q.; Luo, J.; van Sloun, R.J.G.; Eldar, Y.C. Deep Unfolded Robust PCA With Application to Clutter Suppression in Ultrasound. *IEEE Trans. Med. Imag.* **2020**, *39*, 1051–1063. [CrossRef]
31. Gregor, K.; LeCun, Y. Learning fast approximations of sparse coding. In Proceedings of the 27th International Conference on International Conference on Machine Learning, Haifa, Israel, 21–24 June 2010; pp. 399–406.
32. Kim, D.; Park, D. Element-Wise Adaptive Thresholds for Learned Iterative Shrinkage Thresholding Algorithms. *IEEE Access* **2020**, *8*, 45874–45886. [CrossRef]
33. Musa, O.; Jung, P.; Caire, G. Plug-And-Play Learned Gaussian-mixture Approximate Message Passing. In Proceedings of the ICASSP 2021–2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Toronto, ON, Canada, 6–11 June 2021; pp. 4855–4859. [CrossRef]
34. Li, Y.; Tofighi, M.; Geng, J.; Monga, V.; Eldar, Y.C. Efficient and Interpretable Deep Blind Image Deblurring Via Algorithm Unrolling. *IEEE Trans. Comput. Imag.* **2020**, *6*, 666–681. [CrossRef]
35. Monga, V.; Li, Y.; Eldar, Y.C. Algorithm Unrolling: Interpretable, Efficient Deep Learning for Signal and Image Processing. *IEEE Signal Process. Mag.* **2021**, *38*, 18–44. [CrossRef]
36. Gu, S.; Zhang, L.; Zuo, W.; Feng, X. Weighted Nuclear norm minimization with application to image denoising. In Proceedings of the IEEE Conf. on Comput. Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 2862–2869. [CrossRef]
37. Cohen, R.; Zhang, Y.; Solomon, O.; Toberman, D.; Taieb, L.; van Sloun, R.J.; Eldar, Y.C. Deep Convolutional Robust PCA with Application to Ultrasound Imaging. In Proceedings of the ICASSP 2019–2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 3212–3216. [CrossRef]
38. Gabay, D.; Mercier, B. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Comput. Math. Appl.* **1976**, *2*, 17–40. [CrossRef]
39. Lu, C.; Feng, J.; Yan, S.; Lin, Z. A Unified Alternating Direction Method of Multipliers by Majorization Minimization. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 527–541. [CrossRef]
40. Mu, Y.; Dong, J.; Yuan, X.; Yan, S. Accelerated low-rank visual recovery by random projection. In Proceedings of the CVPR 2011, Colorado Springs, CO, USA, 20–25 June 2011; pp. 2609–2616. [CrossRef]
41. Wei, C.; Chen, C.; Wang, Y.F. Robust Face Recognition With Structurally Incoherent Low-Rank Matrix Decomposition. *IEEE Trans. Image Process.* **2014**, *23*, 3294–3307. [CrossRef] [PubMed]
42. Rangan, S.; Schniter, P.; Fletcher, A.K.; Sarkar, S. On the convergence of approximate message passing with arbitrary matrices. *IEEE Trans. Inf. Theory* **2019**, *65*, 5339–5351. [CrossRef]
43. Yang, Z.; Xie, L.; Zhang, C. Off-grid direction of arrival estimation using sparse Bayesian inference. *IEEE Trans. Signal Process.* **2012**, *61*, 38–43. [CrossRef]
44. Wipf, D.; Nagarajan, S. Iterative Reweighted ℓ_1 and ℓ_2 Methods for Finding Sparse Solutions. *IEEE J. Sel. Top. Signal Process.* **2010**, *4*, 317–329. [CrossRef]
45. Malek-Mohammadi, M.; Babaie-Zadeh, M.; Skoglund, M. Iterative Concave Rank Approximation for Recovering Low-Rank Matrices. *IEEE Trans. Signal Process.* **2014**, *62*, 5213–5226. [CrossRef]
46. Fazel, M.; Hindi, H.; Boyd, S.P. Log-det heuristic for matrix rank minimization with applications to Hankel and Euclidean distance matrices. In Proceedings of the 2003 American Control Conf., Denver, CO, USA, 4–6 June 2003; Volume 3, pp. 2156–2162. [CrossRef]
47. Lu, C.; Tang, J.; Yan, S.; Lin, Z. Nonconvex Nonsmooth Low Rank Minimization via Iteratively Reweighted Nuclear Norm. *IEEE Trans. Image Process.* **2016**, *25*, 829–839. [CrossRef]
48. Peng, Y.; Suo, J.; Dai, Q.; Xu, W. Reweighted low-rank matrix recovery and its application in image restoration. *IEEE Trans. Cybernetics* **2014**, *44*, 2418–2430. [CrossRef]
49. Chivers, I.; Sleightholme, J. An introduction to Algorithms and the Big O Notation. In *Introduction to Programming with Fortran*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 359–364.
50. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2017**, arXiv:1412.6980.
51. The MathWorks Inc. *MATLAB: Version 9.6.0 (R2019a)*; The MathWorks Inc.: Natick, MA, USA, 2019.
52. Grant, M.; Boyd, S. *CVX: Matlab Software for Disciplined Convex Programming, Version 2.1*; 2014. Available online: http://cvxr.com/cvx (accessed on 10 January 2022).
53. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 8024–8035.
54. Ahmed, A.M.; Thanthrige, U.S.K.M.; El Gamal, A.; Sezgin, A. Deep Learning for DOA Estimation in MIMO Radar Systems via Emulation of Large Antenna Arrays. *IEEE Commun. Lett.* **2021**, *25*, 1559–1563. [CrossRef]
55. Sutskever, I.; Martens, J.; Dahl, G.; Hinton, G. On the importance of initialization and momentum in deep learning. In Proceedings of the International Conference on Machine Learning, Atlanta, GA, USA, 17–19 June 2013; pp. 1139–1147.

56. Candès, E.J.; Recht, B. Exact matrix completion via convex optimization. *Found. Comput. Math.* **2009**, *9*, 717–772. [CrossRef]
57. Gross, D. Recovering low-rank matrices from few coefficients in any basis. *IEEE Trans. Inf. Theory* **2011**, *57*, 1548–1566. [CrossRef]
58. Candès, E.J.; Tao, T. The power of convex relaxation: Near-optimal matrix completion. *IEEE Trans. Inf. Theory* **2010**, *56*, 2053–2080. [CrossRef]