Robust Beamforming Based on Complex-Valued Convolutional Neural Networks for Sensor Arrays

Saeed Mohammadzadeh†, Vítor H. Nascimento†, Rodrigo C. de Lamare†† and Noushin Hajarolasvadi†††

***Abstract*—Robust adaptive beamforming (RAB) plays a vital role in modern communications by ensuring the reception of high-quality signals. This paper proposes a deep learning approach to robust adaptive beamforming. In particular, we propose a novel RAB approach where the sample covariance matrix (SCM) is used as the input of a deep 1D Complex-Valued Convolutional Neural Network (CVCNN). The network employs complex convolutional and pooling layers, as well as a Cartesian Scaled Exponential Linear Unit activation function to directly compute the nearly-optimum weight vector through the training process and without prior knowledge about the direction of arrival of the desired signal. This means that reconstruction of the interference plus noise (IPN) covariance matrix is not required. The trained CVCNN accurately computes the nearly-optimum weight vector for data not used during training. The computed weight vector is employed to estimate the signal-to-interference plus noise ratio. Simulations show that the proposed RAB can provide performance close to that of the optimal beamformer.**

***Index Terms*—Convolutional neural network, Robust adaptive beamforming, Sample matrix inversion.**

I. Introduction

ADAPTIVE beamforming has applications in a wide range of signal processing domains, including radar, sonar, and wireless communication systems. Beamformers provide spatial information about signals in the presence of different types of interference [1]. The minimum variance distortionless response (MVDR) beamformer was proposed to recover the signal-of-interest (SOI) in the array input while minimizing the array output power. However, the MVDR beamformer faces problems in practice due to several reasons, which include short data records, imprecise assumptions on the models for the source, environment, and the array steering vector of the desired signal (DS) [2], [3]. Adaptive beamformers are quite sensitive to errors caused by an inaccurate estimation of the covariance matrix, especially when the SOI component is present in the training data. Hence, various techniques have been developed to improve the performance of adaptive beamformers. These Robust adaptive beamforming (RAB) techniques can be divided into two categories.

First, the conventional approaches, which are based on diagonal loading [4], [5], the eigenspace-based beamformer [6], [7], worst-case optimization and estimation with presumed prior knowledge [8], [9]. RAB methods designed based on these approaches have disadvantages like their ad hoc nature, high probability of subspace swap at low signal-to-noise ratio (SNR) and high computational complexity.

The second category is based on an approach in which the influence of the SOI component from the sample covariance matrix (SCM) is removed by reconstructing the interference-plus-noise (IPN) covariance matrix. In these methods the IPN covariance matrix is reconstructed based on the Capon or maximum entropy spectral estimator by integrating over an angular sector that excludes the direction-of-arrival (DoA) of the SOI [10]–[20].

More recently, neural networks have been employed to estimate nearly-optimal weight vectors for beamforming. The parallel structure of these networks and their ability to learn non-linear features using activation functions make them an effective approach for estimating the desired output in the presence of interference and noise. In [21] a neural network has been trained on an optimized data set extracted by a modified adaptive dispersion variant of the invasive weed optimization algorithm In [22], a radial-basis-function network has been employed to estimate the weight vector of a weather radar antenna array in the presence of interference. In [23], a deep neural network has been employed to estimate the DoA of DSs. A model-aware deep learning strategy for ultrasound image reconstruction has been proposed in [24] which utilizes the knowledge of minimum variance beamforming. Two CNNs based on the parallel processing ability of neural networks with identical structures have been designed to estimate beamformer weights without knowing the DoAs of signals in [25].

However, all these methods are designed based on Real-Valued Convolutional Neural Networks (RVCNN). Recently, [26] reported that RVCNNs do not provide the best result when applied to complex-valued (CV) problems. A Complex-Valued Convolutional Neural Network (CVCNN) seems a more natural choice to learn from CV features because layers of CVCNNs can perform complex filtering operations. Notably, CVCNNs are more adapted to extract phase information, which could be helpful to retrieve features of signals. The results in [26] show that CVCNNs perform better than their counterpart RVCNNs by presenting a larger mean and median and lower variance than RVCNNs.

The contributions of this paper are two-fold: first, we introduce a novel deep one-dimensional (1D) CVCNN for RAB that can compute nearly-optimal beamformers accurately using SCM elements and without requiring the DoA of the SOI or interference signals. Second, we show that applying CVCNNs to CV problems is a better approach than RVCNNs for weight vector estimation due to complex filtering operations. In contrast to prior works, we address the problem of estimating the weight vector as a regression problem by applying a complex model to SCM without separating real and imaginary parts. The network employs complex convolutional and pooling layers, as well as Cartesian Scaled Exponential Linear Unit (CSeLU) activation function to directly compute the weight vector through the training process. The proposed CVCNNs algorithm is capable of creating notches in the directions of the

interference with sufficient depths so that interference signals can be effectively suppressed while avoiding the reconstruction of the interference-plus-noise covariance matrix.

II. Problem Background

Consider a uniform linear array of M omnidirectional sensors with interelement spacing d. The receiving signals are narrowband plane waves impinging from angle θ. The array observation vector at time t can be modeled as

$$\mathbf{x}(t) = s(t)\mathbf{a}(\theta_s) + \mathbf{v}(t), \tag{1}$$

where the signals $x_l(t)$ observed at each antenna are collected in the vector $\mathbf{x}(t) = [x_0(t) \dots x_{M-1}(t)]^T$, and $(\cdot)^{\mathrm{T}}$ is the transpose while $\mathbf{v}(t)$, $s(t)$ denote the sum of the interference plus noise and the waveform of the SOI, respectively. The SV corresponding to the DoA of the SOI is, $\mathbf{a}(\theta_{\mathrm{s}}) = \left[1,\ e^{-j\frac{2\pi}{\lambda}d\sin\theta_s},\ \cdots,\ e^{-j(M-1)\frac{2\pi}{\lambda}d\sin\theta_s}\right]^T$, where λ is the wavelength. Assuming that the SV $\mathbf{a}(\theta_s)$ is known, then for a given beamformer weight vector $\mathbf{w}$, the beamformer performance is measured using the output signal-to-interference-plus-noise ratio (SINR) as follows

$$\mathrm{SINR} = \sigma_{\mathrm{s}}^2 |\mathbf{w}^{\mathrm{H}}\mathbf{a}(\theta_s)|^2 / \mathbf{w}^{\mathrm{H}}\mathbf{R}_{\mathrm{i+n}}\mathbf{w}, \tag{2}$$

where σ_{s}^2 is the DS power, $\mathbf{R}_{\mathrm{i+n}}$ is the IPN covariance matrix, and $(\cdot)^{\mathrm{H}}$ stands for Hermitian transpose. The weight vector is found from the maximum of the SINR and it is equivalent to

$$\min_{\mathbf{w}} \mathbf{w}^{\mathrm{H}}\mathbf{R}_{\mathrm{i+n}}\ \mathbf{w} \quad \text{s.t.} \quad \mathbf{w}^{\mathrm{H}}\mathbf{a}(\theta_s) = 1. \tag{3}$$

The solution to (3) yields the optimal beamformer given by

$$\mathbf{w}_{\mathrm{opt}} = \zeta\mathbf{R}_{\mathrm{i+n}}^{-1}\mathbf{a}(\theta_s). \tag{4}$$

where ζ is a scale factor. When $\zeta = \sigma_s^2$, the weights correspond to spatial Wiener filter and when $\zeta = \mathbf{a}^{\mathrm{H}}(\theta_s)\mathbf{R}_{\mathrm{i+n}}^{-1}\mathbf{a}(\theta_s)$, the MVDR weights are obtained. Since in practice, the exact IPN covariance matrix $\mathbf{R}_{\mathrm{i+n}}$ is unavailable even in signal-free applications, they are replaced by the SCM, $\hat{\mathbf{R}} = (1/K)\sum_{t=1}^{K}\mathbf{x}(t)\mathbf{x}^{\mathrm{H}}(t)$, where K is the number of snapshots. When the covariance matrix of interference and noise is replaced by the sample matrix obtained using a set of training data, the adaptive version of the MVDR beamformer is referred to as the sample matrix inversion (SMI) beamformer. We focus on the SMI beamformer throughout this work.

III. The Proposed CVCNN Algorithm

In the SMI beamformer, the amount of required data is necessary to achieve a desired level of performance. In fact, the basic reason that an SMI-based adaptive beamformer can not converge rapidly is the dispersion of noise eigenvalues under the condition of few snapshots and high SNR. Under these conditions, the performance of SMI-based adaptive beamformer decreases seriously.

To improve the robustness of the beamformer with a poorly conditioned SCM, the diagonal loading [27] technique has been exploited to suppress pattern distortion. Generally, diagonal loading techniques can reduce the influence of small eigenvalues on the noise beam and improve the pattern distortion, but the determination of the loading value is always a difficult problem to address [28]. Therefore, in this work, we propose a novel deep neural network-based method using the SCM with no prior processing. The network is trained to learn the dependency of the optimal weights on the properties of the received signals at different DoAs. The trained network can compute a nearly-optimal weight vector without requiring knowledge about the DoA of the DS. The architecture of the CNN and the necessary input/output pairs used for beamforming are detailed in the following sections.

A. Architecture of the CVCNN

CVCNN can be considered as an extension of the conventional RVCNNs with neurons and weights that can handle complex values. Just like RVCNNs, a CVCNN includes an input layer followed by several alternations of convolutional and pooling layers, fully connected layers, and a regression or classification layer. Phase introduction is the main advantage of CV operations over the real-valued networks while processing a signal. A typical process of feature extraction in CNNs is a convolution layer with nonlinear activation followed by a pooling layer. Convolution layers perform convolution with multiple learnable filters in parallel and results are fed into nonlinear activation functions to generate feature maps. Then, a pooling function reduces spatial dimension by downsampling the feature maps. In a CVCNN framework that suits sensor array processing, all such operations should be based on complex values. Below, we discuss the most important elements of the proposed 1D CVCNN.

1) **Convolution:** Hidden units of the convolutional layers are connected to the feature maps of the previous layer through a complex weight matrix known as kernel. The units are convolved with the weight matrix, and then passed through a nonlinear activation function. The weights are chosen so that they minimize a certain loss function. In the CV convolutional layer, the RV convolutional operation is generalized to the complex domain. Let us define a convolution operation between the input $\mathbf{x}(t)$ and a kernel $\mathbf{f}$ by $\mathbf{x}(t) \circledast \mathbf{f}$ where both $\mathbf{x}(t)$ and $\mathbf{f}$ are complex vectors expressed as follows:

$$\mathbf{x}(t) = \mathbf{x}_r(t) + j\mathbf{x}_c(t) \qquad , \qquad \mathbf{f} = \mathbf{f}_r + j\mathbf{f}_c \tag{5}$$

Here, $\mathbf{x}_r(t)$, $\mathbf{x}_c(t)$, $\mathbf{f}_c$, and $\mathbf{f}_c$ are real-valued vectors. So the the CV convolution operation can be formulated as $\mathbf{x}(t) \circledast \mathbf{f} = (\mathbf{x}_r(t) \circledast \mathbf{f}_r - \mathbf{x}_c(t) \circledast \mathbf{f}_c) + j(\mathbf{x}_c(t) \circledast \mathbf{f}_r + \mathbf{x}_r(t) \circledast \mathbf{f}_c)$ which can in turn be represented as:

$$\begin{bmatrix} \Re(\mathbf{x}(t) \circledast \mathbf{f}) \\ \Im(\mathbf{x}(t) \circledast \mathbf{f}) \end{bmatrix} = \begin{bmatrix} \mathbf{x}_r(t) & -\mathbf{x}_c(t) \\ \mathbf{x}_c(t) & \mathbf{x}_r(t) \end{bmatrix} \circledast \begin{bmatrix} \mathbf{f}_r \\ \mathbf{f}_c \end{bmatrix} \tag{6}$$

where $\Re(.)$ and $\Im(.)$ represent the real and imaginary parts of a complex number. According to equation 6, a CV convolution operation with filter $\mathbf{f} = \mathbf{f}_r + j\mathbf{f}_c$ can be considered as an RV convolution with the following two kernels: $[\mathbf{f}_r, -\mathbf{f}_c]$ and $[\mathbf{f}_c, \mathbf{f}_r]$. Consequently, a CV convolutional layer can be considered as an extended version of a RV convolutional layer with twice as many kernels.

2) **Activation function:** Activation functions are nonlinear functions that characterize the behavior of a neural network. The rectified linear unit function is the most used example in RVCNNs and it can be defined as $\max(0, x)$. To stay close to the real case, the Complex Rectified Linear Unit (CReLU) [29] should be defined for some connected $A \subseteq \mathbb{C}$. Then the output is $\max(0, z)$ for $z \in A$, where the max operator should be interpreted as follows:

$$\text{CReLU(z)} = \max(0, z) = \begin{cases} z & \text{if } \Re(z),\, \Im(z) \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

which is used in all convolutional layers of the proposed method. The fully connected layer uses CSeLU [30] as an activation function that can be formulated as:

$$\text{CSeLU(z)} = \beta \begin{cases} z & \text{if } \Re(z),\, \Im(z) > 0 \\ \alpha e^z - \alpha & \text{if } \Re(z),\, \Im(z) \leq 0 \end{cases} \tag{8}$$

with z being a complex input of the function, $\beta > 1$ to have a slope greater than one for positive inputs [30] and $\alpha > 0$ to control the value to which the function saturates in case of negative inputs [31].

Various approaches to apply RVCNNs to CV problems have been investigated. For instance, [32] separates and rearranges the real and imaginary parts into a single vector that is employed later as the input of the RVCNN. Recently, Barrachina et al. [26] reported that CVCNN shows higher accuracy and less overfitting than RVCNN regardless of the model architecture and hyper-parameters. Therefore, to solve the problem in the complex domain, we propose a sixteen-layer CVCNN where all layers, including convolution, pooling, and dense layers perform complex filtering operations.

The proposed network is a deep 1D CVCNN that starts with an input layer. The size of input nodes for this layer and the data fed to this layer is discussed in more detail in section III-B. The input layer is followed by the first complex convolution layer that has 8 kernels of size 3 and a complex average pooling layer with kernel size 2 and stride 2. The convolution and pooling layer is repeated then by having 16, 32, 64, and 128 kernels of size 3 and exact same pooling layers. Next, two convolution layers with 256 filters are added where the output is squeezed to a complex dense layer. This is a followed by a drop out layer with rate 0.25 to avoid the overfitting problem and a general average pooling layer. Finally, a fully connected layer with 50 nodes is employed to extract the best set of features and pass them to a regression layer which applies Cartesian Hyperbolic Tangent for prediction. The proposed CVCNN is shown in Fig. 1. We specifically propose this configuration to show we can achieve deeper architectures using CV networks. Also, the number of kernels is increased toward the final layers so the network is able to capture low-level features in the first layers and elaborate complex features in the final ones.

B. Training and Testing

For generating the input vectors of the CVCNN, we consider two scenarios. First, we generate an input vector where all the elements of the sample covariance matrix $\hat{\mathbf{R}}$ are reshaped

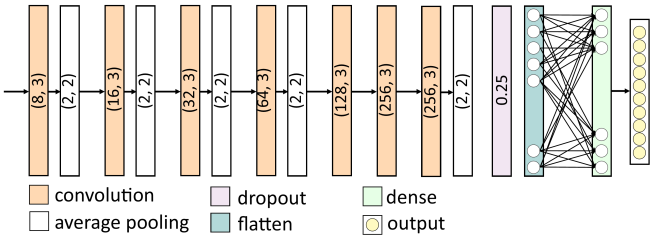


Fig. 1. Block diagram of the proposed CVCNN architecture

by concatenating the rows into a vector of size $1 \times M^2$ as $\mathbf{r}_{\text{R}} = [\hat{R}_{11}, \cdots, \hat{R}_{1M}\ \hat{R}_{m1}, \cdots, \hat{R}_{mM}\ \cdots\ \hat{R}_{M1}, \cdots, \hat{R}_{MM}]$. From here on, we call this network CVCNN-SCM. Where as in the second scenario, an input vector containing only the lower triangular elements of the sample covariance matrix (LTSCM) are reshaped to a vector of size $1 \times \frac{M(M+1)}{2}$ as $\mathbf{r}_{\text{LR}} = [\hat{R}_{11}\ \cdots \hat{R}_{m1}, \cdots, \hat{R}_{mm} \cdots \hat{R}_{M1}, \cdots, \hat{R}_{MM}]$.

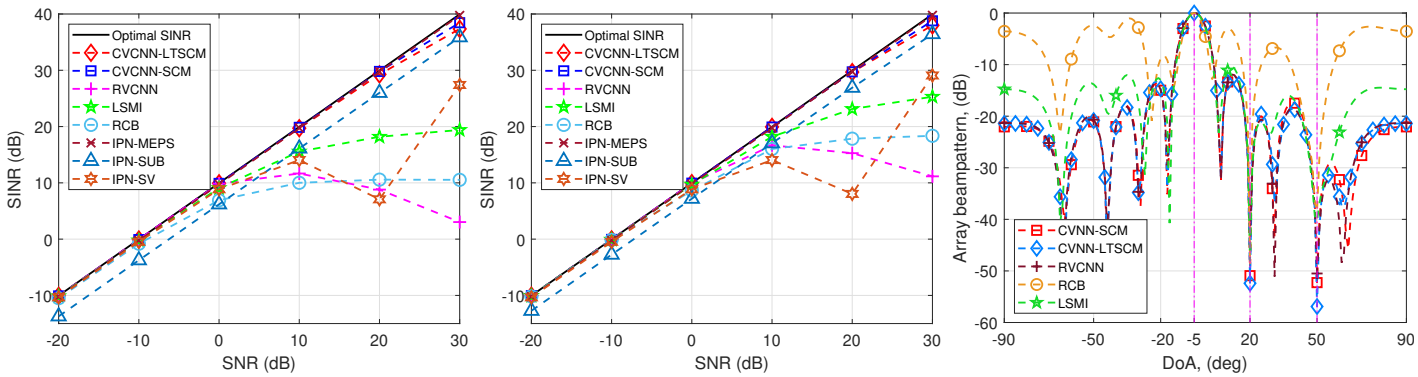
We refer to this network as CVCNN-LTSCM. These two scenarios are studied to show that using only lower triangular elements of the SCM suffices to achieve high performance in a reasonable computational time by reducing the data redundancy. In order to train the networks in a supervised learning manner, one requires pairs of inputs and ground truth labels. The accuracy of the networks significantly depends on the generated labels. So as to generate the corresponding ground truth labels $\mathbf{w}$ of each sample $\mathbf{r}$, we proceed as follows:

$$\mathbf{w} = \hat{\mathbf{R}}^{-1}\mathbf{a}(\hat{\theta}_s) / \mathbf{a}^{\text{H}}(\hat{\theta}_s)\hat{\mathbf{R}}^{-1}\mathbf{a}(\hat{\theta}_s) \tag{9}$$

which is used as training label and has the size $M \times 1$. Here, $\mathbf{a}(\hat{\theta}_s)$ is the assumed DS steering vector. It is important to mention that input vectors $\mathbf{r}_{\text{R}}$ and $\mathbf{r}_{\text{LR}}$ and labels generated by (9) are normalized before applying to the network.

The network should be fully trained, and converge to optimum weights by minimizing loss function based on Mean Absolute Error (MAE) while using early stopping criteria. At the performance stage, the trained network can be directly used to produce weight vectors for real-time beamforming. We produced 4500 sample pairs of $(\mathbf{r}_{\text{R/LR}}, \mathbf{w})$ for the training data from a Gaussian distribution of zero mean and unit variance while in order to assess the generalization ability of the model considering unseen data, 500 sample pairs for test are generated with a Gaussian distribution of zero mean and variance 4. The training data is then used in a 5-fold cross-validation fashion to avoid overfitting. The network is computationally trained for 100 epochs with various optimizer algorithms, namely, SGD, Nadam, RMS, and ADAM. We illustrate the learning curve for SNR -10 dB in Fig 3. The learning curve represents the correct convergence procedure and generalization capabilities of the model. In Table I MAE values are compared for RVCNN and CVCNN-LTSCM under two snapshot scenarios and using different number of samples. We also performed a case study using different optimization metrics like mean square error, MAE, and accuracy. The best performance was achieved by using Adam while optimizing MAE and accuracy.

In all experiments, the learning rate and batch size were set to 1e-04 and 4, respectively. We employed an adaptive



(a) (b) (c)

Fig. 2. (a) Output SINR versus SNR for 100 snapshots; (b) Output SINR versus SNR for 400 snapshots; (c) Normalized Beampattern

snapshots	500 samples		5000 samples	
	CVCNN	RVCNN	CVCNN	RVCNN
100	0.099	0.186	0.038	0.157
400	**0.086**	0.157	**0.021**	0.135

TABLE I
MAE VALUE FOR CVCNN-LTSCM AND RVCNN ON VALIDATION DATA

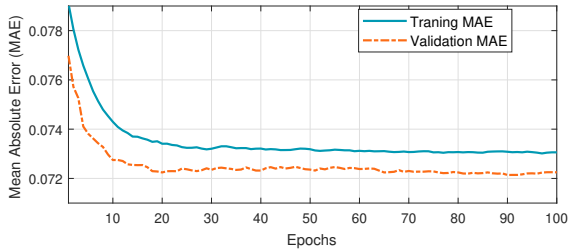


Fig. 3. Learning Curve of the CVCNN-LTSCM model with Adam optimizer

exponential learning rate decay with a factor of 0.96 and decay step 10. In order to achieve the best performance in a reasonable computational time, we employ early stopping criteria where for some SNRs an under-trained CVCNN occurred. As mentioned before, the test data are generated from a Gaussian distribution with zero mean and variance 4. The trained models predict the near-optimum weight vectors for the unseen data (test). We use the average predicted weight vector to calculate the SINR values.

IV. SIMULATIONS

In this section, a ULA with $M = 10$ omnidirectional sensors is used. The additive noise is modeled as spatially white Gaussian with zero mean and unit variance. The angles of incidence of the DS and two interfering sources are $\bar{\theta}_s = -5°$, $20°$ and $50°$ respectively. The input interference to noise ratios of the two interferers are both set to 20 dB. The proposed CVCNN-SCM and CVCNN-LTSCM methods are compared with the RCB method in [27], the loaded sample matrix inverse (LSMI), the RVCNN method in [32], the beamformer in [16] (IPN-MEPS), the method in [17] (IPN-SUB) and the algorithm in [33] (IPN-SV). For the IPN covariance matrix reconstruction methods, the angular sector of the DS is set to $[\bar{\theta}_s - 4°, \bar{\theta}_s + 4°]$ where the interference angular sector is $[-90°, \bar{\theta}_s - 4°) \cup (\bar{\theta}_s + 4°, 90°]$. The energy percentage ρ set as 0.9 in IPN-SUB. The SINR performance of the proposed methods is assessed versus the SNRs. Fig. 2(a) demonstrates the performance for 100 snapshots while Fig. 2(b) illustrates the SINR performance for 400 snapshots. It is seen that the proposed methods attain the optimal output SINR in both low and high SNRs and for different numbers of snapshots. The excellent performance of CVCNN-SCM and CVCNN-LTSCM is due to the extraction of high-level features during the learning procedure. This enhances the robustness of the proposed methods to estimate the weight vectors regardless of the number of snapshots. However, RVCNNs do not provide the best results when applied to complex-valued problems even after increasing the number of snapshots from 100 to 400 since it is not adapted to extract phase information, which could be helpful to retrieve features of sensor array signals. To demonstrate the IPN suppression capability of the proposed algorithms, we plot the normalized beampattern in comparison with those of the LSMI, RCB, and the RVCNN [32]. It is assumed that the input SNR and the number of snapshots are fixed at 10 dB and 400, respectively. Fig. 2(c) shows the normalized beampattern plots. The CVCNN-SCM and CVCNN-LTSCM beampatterns indicate that the desired directional response shaping has been achieved with lower side-lobe levels. Moreover, it demonstrates that the proposed methods are able to recover the desired array steering vector by pointing its main-lobe in the DS direction while the interference signals are suppressed with deep nulls.

V. CONCLUSION

In this letter, a deep learning approach to RAB is proposed to compute nearly optimal beamformers. Unlike prior works, we addressed the drawback of the sample matrix inversion by estimating the weight vector as a regression problem using a 1D CVCNN. Notably, this is achieved without requiring the knowledge of the number of sources, the corresponding DoA, or addressing the weight vector estimation as a real-valued problem. Simulation results show the performance of the proposed 1D CVCNN algorithm is robust to short data records and outperforms the state-of-the-art in the literature.

REFERENCES

[1] H. L. Van Trees, *Detection, Estimation, and Modulation Theory*. John Wiley & Sons, New York, 2004.

[2] S. Shahbazpanahi, A. B. Gershman, Z.-Q. Luo, and K. M. Wong, "Robust adaptive beamforming for general-rank signal models," *IEEE Trans. on Signal Process.*, vol. 51, no. 9, pp. 2257–2269, 2003.

[3] L. Zhang and W. Liu, "Robust forward backward based beamformer for a general-rank signal model with real-valued implementation," *Signal Process.*, vol. 92, no. 1, pp. 163–169, 2012.

[4] X. Mestre and M. A. Lagunas, "Finite sample size effect on minimum variance beamformers: Optimum diagonal loading factor for large arrays," *IEEE Trans. on Signal Process.*, vol. 54, no. 1, pp. 69–82, 2006.

[5] O. Kukrer and S. Mohammadzadeh, "Generalised loading algorithm for adaptive beamforming in ulas," *Electronics Letters*, vol. 50, no. 13, pp. 910–912, 2014.

[6] F. Huang, W. Sheng, and X. Ma, "Modified projection approach for robust adaptive array beamforming," *Signal Process.*, vol. 92, no. 7, pp. 1758–1763, 2012.

[7] S. Mohammadzadeh and O. Kukrer, "Modified robust Capon beamforming with approximate orthogonal projection onto the signal-plus-interference subspace," *Circuits, Systems, and Signal Process.*, pp. 1–18, 2018.

[8] S. A. Vorobyov, A. B. Gershman, and Z.-Q. Luo, "Robust adaptive beamforming using worst-case performance optimization: A solution to the signal mismatch problem," *IEEE Trans. on Signal Process.*, vol. 51, no. 2, pp. 313–324, 2003.

[9] S. E. Nai, W. Ser, Z. L. Yu, and H. Chen, "Iterative robust minimum variance beamforming," *IEEE Trans. on Signal Process.*, vol. 59, no. 4, pp. 1601–1611, 2011.

[10] Y. Gu and A. Leshem, "Robust adaptive beamforming based on interference covariance matrix reconstruction and steering vector estimation," *IEEE Trans. on Signal Process.*, vol. 60, no. 7, pp. 3881–3885, 2012.

[11] H. Ruan and R. C. de Lamare, "Robust adaptive beamforming using a low-complexity shrinkage-based mismatch estimation algorithm." *IEEE Signal Process. Lett.*, vol. 21, no. 1, pp. 60–64, 2014.

[12] X. Yuan and L. Gan, "Robust algorithm against large look direction error for interference-plus-noise covariance matrix reconstruction," *Electronics Letters*, vol. 52, no. 6, pp. 448–450, 2016.

[13] Z. Zhang, W. Liu, W. Leng, A. Wang, and H. Shi, "Interference-plus-noise covariance matrix reconstruction via spatial power spectrum sampling for robust adaptive beamforming," *IEEE Signal Processing Letters*, vol. 23, no. 1, pp. 121–125, 2016.

[14] S. Mohammadzadeh and O. Kukrer, "Adaptive beamforming based on theoretical interference-plus-noise covariance and direction-of-arrival estimation," *IET Signal Process.*, vol. 12, no. 7, pp. 819–825, 2018.

[15] Z. Zheng, Y. Zheng, W.-Q. Wang, and H. Zhang, "Covariance matrix reconstruction with interference steering vector and power estimation for robust adaptive beamforming," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 9, pp. 8495–8503, 2018.

[16] S. Mohammadzadeh, V. H. Nascimento, R. C. de Lamare, and O. Kukrer, "Maximum entropy-based interference-plus-noise covariance matrix reconstruction for robust adaptive beamforming," *IEEE Signal Process. Lett.*, vol. 27, pp. 845–849, 2020.

[17] X. Zhu, X. Xu, and Z. Ye, "Robust adaptive beamforming via subspace for interference covariance matrix reconstruction," *Signal Processing*, vol. 167, p. 107289, 2020.

[18] S. Mohammadzadeh, V. H. Nascimento, R. C. de Lamare, and O. Kukrer, "Robust adaptive beamforming based on virtual sensors using low-complexity spatial sampling," *Signal Processing*, vol. 188, p. 108172, 2021.

[19] S. Mohammadzadeh, V. H. Nascimento, R. C. De Lamare, and O. Kukrer, "Robust adaptive beamforming based on low-complexity discrete fourier transform spatial sampling," *IEEE Access*, 2021.

[20] ——, "Robust adaptive beamforming based on power method processing and spatial spectrum matching," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 4903–4907.

[21] Z. D. Zaharis, C. Skeberis, T. D. Xenos, P. I. Lazaridis, and J. Cosmas, "Design of a novel antenna array beamformer using neural networks trained by modified adaptive dispersion invasive weed optimization based data," *IEEE Transactions on Broadcasting*, vol. 59, no. 3, pp. 455–460, 2013.

[22] T. Sallam, A. B. Abdel-Rahman, M. Alghoniemy, Z. Kawasaki, and T. Ushio, "A neural-network-based beamformer for phased array weather radar," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 9, pp. 5095–5104, 2016.

[23] Z.-M. Liu, C. Zhang, and S. Y. Philip, "Direction-of-arrival estimation based on deep neural networks with robustness to array imperfections," *IEEE Transactions on Antennas and Propagation*, vol. 66, no. 12, pp. 7315–7327, 2018.

[24] B. Luijten, R. Cohen, F. J. de Bruijn, H. A. Schmeitz, M. Mischi, Y. C. Eldar, and R. J. van Sloun, "Deep learning for fast adaptive beamforming," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 1333–1337.

[25] A. M. Elbir, "Cnn-based precoder and combiner design in mmwave mimo systems," *IEEE Communications Letters*, vol. 23, no. 7, pp. 1240–1243, 2019.

[26] J. A. Barrachina, C. Ren, C. Morisseau, G. Vieillard, and J.-P. Ovarlez, "Complex-valued vs. real-valued neural networks for classification perspectives: An example on non-circular data," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 2990–2994.

[27] J. Li, P. Stoica, and Z. Wang, "On robust Capon beamforming and diagonal loading," *IEEE Trans. on Signal Process.*, vol. 51, no. 7, pp. 1702–1715, 2003.

[28] X. Mestre and M. A. Lagunas, "Finite sample size effect on minimum variance beamformers: Optimum diagonal loading factor for large arrays," *IEEE Transactions on Signal Processing*, vol. 54, no. 1, pp. 69–82, 2005.

[29] S. Scardapane, S. Van Vaerenbergh, A. Hussain, and A. Uncini, "Complex-valued neural networks with nonparametric activation functions," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 4, no. 2, pp. 140–150, 2018.

[30] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," *Advances in neural information processing systems*, vol. 30, 2017.

[31] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[32] P. Ramezanpour, M. J. Rezaei, and M. R. Mosavi, "Deep-learning-based beamforming for rejecting interferences," *IET Signal Processing*, vol. 14, no. 7, pp. 467–473, 2020.

[33] A. Khabbazibasmenj, S. A. Vorobyov, and A. Hassanien, "Robust adaptive beamforming based on steering vector estimation with as little as possible prior information," *IEEE Trans. on Signal Process.*, vol. 60, no. 6, pp. 2974–2987, 2012.