DEEP-MLE: FUSION BETWEEN A NEURAL NETWORK AND MLE FOR A SINGLE SNAPSHOT DOA ESTIMATION

Marcio L. Lima de Oliveira [1] *and Marco J. G. Bekooij* [1,2]

[1] University of Twente - Enschede, The Netherlands
[2] NXP Semiconductors - Eindhoven, The Netherlands

ABSTRACT

In this paper, we propose a novel framework called Deep-MLE, which gives a solution to the single-snapshot Direction Of Arrival (DOA) estimation problem, up to 4 distinct targets, using a radar equipped with a Minimum Redundancy antenna Array (MRA). This framework works by fusing a Deep Learning (DL) technique - 1D Residual Neural Network (1D ResNet) - with a classical DOA algorithm - Maximum Likelihood Estimation (MLE). By combining two very different approaches, we can address some of their limitations, such as the computational complexity of MLE. On the other hand, our proposed Deep-MLE uses MLE to correct, to some degree, the estimations made by the Neural Network (NN). The results from our framework are promising as it seems to be a viable solution to the DOA estimation problem, having a better performance than models using pure MLE or NN.

Index Terms— Direction of Arrival, Maximum Likelihood Estimation, Residual Neural Network, Single Snapshot

1. INTRODUCTION

Estimating the Direction of Arrival (DOA) has been a challenging problem in different fields, and there are several developed ways to approach it with various algorithms and techniques [1, 2], being signal processing a fundamental part of most of them. Although we have used radar equations and configurations, our work can potentially be expanded to other areas, such as sonars and wireless communication signals.

The employment of Deep Learning (DL) in the signal processing world has already yielded some interesting results, such as reducing noise in radar images [3, 4], sensor fusion [5], among others [6, 7]. Other researchers have developed DOA estimation solutions using Deep Learning. The works by [8, 9] fed snapshots into several Convolutional Neural Networks, in parallel, to create a MUSIC-like spectrum. On the other hand, the researches of [10, 11] utilized Fully Connected Neural Networks with a single-snapshot and Uniform Linear antenna Array (ULA) to estimate DOAs.

For our work, we have explored the multiple DOA problem using a single snapshot and a Minimum Redundancy antenna Array (MRA). Our motivation to chose this scenario

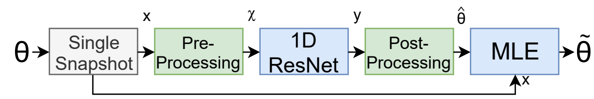

Fig. 1: An illustration of all required steps of Deep-MLE.

was driven by the trend towards autonomous systems. Often, just a single snapshot is available for automotive usage due to the maximum latency requirements [12]. At the same time, MRA can amplify the resolution without adding more antennas to the array [13]. According to Hacker and Yang [14], the Maximum Likelihood Estimation (MLE) algorithm is the most reliable technique to estimate the DOA when using a single snapshot. Therefore, we have used this algorithm both as a comparison and within our proposed framework. In addition, the traditional MLE [15] was used, rather than a different MLE algorithm [16, 17] that could have another trade-off between performance and complexity. MLE works by testing all possible angle combinations and finding the one that gives the least amount of residual error for a previously known antenna model [18]. However, the computational complexity of MLE grows exponentially with the number of sources in the scene [19], making it too costly for many real-time applications.

In this paper, we propose a novel way to solve the DOA estimation problem by fusing a 1D Residual Neural Network (1D ResNet) [20] with the classical MLE algorithm and creating a framework called Deep-MLE. Fig. 1 shows an illustration our proposed framework. Deep-MLE receives a single snapshot provided by an eight antenna MRA and estimates the unknown target angles. Firstly, the snapshot passes through the 1D ResNet. From the output of the neural network, the pre-selected angles are chosen. Sequentiality, only these selected angles are tested and picked by the MLE algorithm, which drastically reduces its computational complexity by up to ten thousand times less than the required by the traditional MLE.

The results of our simulations show that Deep-MLE is a viable solution to solve the DOA problem, taking advantage of both methods as MLE potentially corrects the estimated angles that ResNet would choose.

2. SIGNAL MODEL

Considering M antennas, with the minimum distance between two antennas being $d = \lambda/2$, where λ is the wavelength of the transmitting signal. We can construct an array containing all the distances between antennas as $D(m) = [0, 1, 4, 6, 13, 14, 17, 19]d$, where $m = 1, ..., M$. Furthermore, consider that all N sources, where $0 \leq N \leq M-1$, are in the far-field. The received snapshot, $x \in \mathbb{C}^M$, follows the expression:

$$x = \mathbf{A}(\theta) \cdot s + \eta \,, \tag{1}$$

where $s \in \mathbb{C}^N$ is the sources' signal, $\eta \in \mathbb{C}^M$ is the added Gaussian noise, and $\mathbf{A}(\theta) \in \mathbb{C}^{M \times N}$ is the steering matrix. $\mathbf{A}(\theta)$ is a function of the angles $\theta = \theta_1, ..., \theta_N$. The steering matrix $\mathbf{A}(\theta)$ can be represented as N concatenated steering vectors $\bar{a} \in \mathbb{C}^M$: $\mathbf{A}(\theta) = [\bar{a}_1(\theta_1), \cdots, \bar{a}_N(\theta_N)]$, and each steering vector $\bar{a}$ can be expressed as:

$$\bar{a}_i(\theta_i) = [\alpha_i e^{j2\pi\lambda^{-1}D(1)\theta_i}, \cdots, \alpha_i e^{j2\pi\lambda^{-1}D(M)\theta_i}]^T \,, \tag{2}$$

in which $i = 1, ..., N$, $0 < \alpha_i \leq 1$ is the normalized amplitude of the source s_i, and T is the transpose operation.

The Maximum Likelihood Estimation (MLE) [15] algorithm performs well for estimating the DOAs [14]. The MLE algorithm works by trying all possible steering matrices $\mathbf{A}(\theta)$ containing every possible angles combination that satisfies eq. 1. In other words, MLE searches for the angle combination $\tilde{\theta}$ that produces the least amount of residual error (difference between estimated and actual steering vectors) for a received snapshot x:

$$\tilde{\theta} = \underset{\theta}{\operatorname{argmin}} \, ||x - \mathbf{A}(\theta) \cdot (\mathbf{A}^+(\theta) \cdot x)|| \,, \tag{3}$$

in which $\tilde{\theta} = \tilde{\theta}_1, ..., \tilde{\theta}_N$ are the estimated angles, and $\mathbf{A}^+(\theta)$ is the pseudo-inverse (Moore-Penrose pseudo-inverse) of the matrix $\mathbf{A}(\theta)$. The fact that multiple pseudo-inversions need to be computed is what makes MLE computational intensive [19].

The number of times that MLE needs to perform the eq. 3, and therefore a pseudo-inversion, given by a binomial coefficient, as it explores all possible combinations of angles, is $\binom{G}{N}$; where G is the number of angles MLE has to search, and N is the number of sources.

Since the most computationally heavy operation for MLE is the pseudo-inverse, its operational complexity depends on it. According to Courrieu [21], the Moore-Penrose pseudo-inverse relies on the Singular Value Decomposition (SVD) matrix operation and has a complexity of $O(M^3)$. Since MLE computes $\binom{G}{N}$ times a matrix pseudo-inverse, the operational complexity for MLE can be estimated as: $O_{\text{MLE}}(M^3\binom{G}{N})$. Using eight antennas ($M = 8$) and a Field Of View (FOV) of 180 degrees with a resolution of 1 degree ($G = 181$), we can calculate the operational complexity for different cases by varying the number of DOAs. The complexity for 1D MLE (one target) is 9.2×10^4, 2D MLE is 8.3×10^6, 3D MLE is 4.9×10^8, and 4D MLE is 2.2×10^{10}. As a consequence, the classical MLE is impracticable for many applications.

Table 1: Comparison between different NNs.

NN Type	Operational Complexity	Number of Layers	Accuracy for 2 angles with SNR = 30 dB*
FCNN	3.3×10^6	8 FC Layers	82.29 %
CNN	2.5×10^7	4 2D Conv. + 2 FC Layers	78.7 %
2D ResNet	5.1×10^6	12 2D Conv. + 2 FC Layers	90.1 %
1D ResNet	1.68×10^6	12 1D Conv. + 2 FC Layers	90.1 %

*Trained and evaluated with a reduced dataset.

The computational complexity of neural networks depend on the number and type of its layers. For a convolutional layer, the complexity depends on the number of filters (or channels) of its input and output (C_{in} and C_{out} respectively), the size of the layer's input I_{in}, the size of the kernel filter F, and the number of strides S done during the convolution. On the other hand, the operational complexity of a fully connected layer just depends on the size of the input and output (I_{in} and I_{out} respectively). Therefore, the computation complexity for one convolutional layer is $O_{\text{conv}}(C_{in}C_{out}\frac{I_{in}}{S}F)$, and for one fully connected layer is $O_{\text{fc}}(I_{in}I_{out})$.

During the development of this work, we have tried various NN architectures. Using the O_{conv} and O_{fc} equations to calculate the operational complexity of each layer, we can estimate the operational complexity of these neural networks. As seen in Table 1, we chose the 1D ResNet (architecture can be seen in Fig. 2), as it is the one with the least computational complexity while retaining one of the best accuracies. An advantage of using neural networks is the fact that their operational complexity does not grow depending on the number of targets.

3. DEEP-MLE - DATA AND ARCHITECTURE

In this paper, we propose a novel framework, called Deep-MLE, that fuses deep learning (1D ResNet) and a classical algorithm (MLE) for estimating the direction of arrival. By doing so, we can generally achieve better performance than models that use only DL or MLE.

When the received snapshot enters our Deep-MLE, the first step is a fast pre-processing before heading to the neural network. Unlike the original ResNet proposed by He et al. [20], our 1D ResNet requires a vector (1D) as an input, not a matrix (2D). In addition, since most machine learning methods do not utilize complex numbers directly, the received snapshot x needs to be modified. For that, our neural network is fed by a vector containing both the real and imaginary parts of x. The input of our ResNet $\chi \in \mathbb{R}^{2M}$ is given as: $\chi_i = \text{Re}\{x_i\}$ and $\chi_{i+M} = \text{Im}\{x_i\}$, which means that the input χ has a size of $2M$, with both the real and imaginary

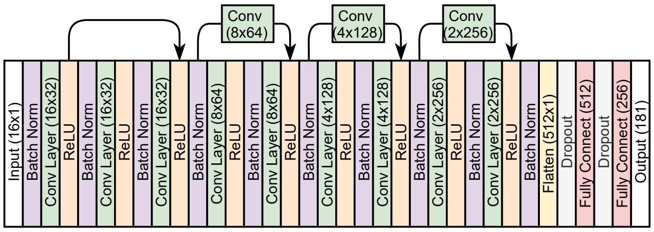

Fig. 2: The proposed 1D ResNet architecture inside our Deep-MLE framework

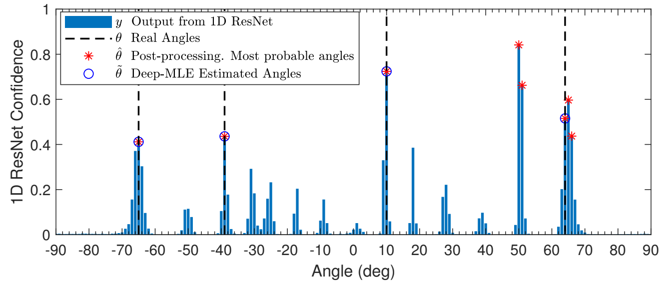

Fig. 3: ResNet's output, post-processing, and MLE steps of our Deep-MLE framework.

parts of the received snapshot x concatenated together into a single vector.

The output layer of the neural network is a fully connected layer. The number of neurons in this last layer depends on the used resolution and FOV. The output layer will contain $(\mathrm{FOV}/\mathrm{resolution})+1$ neurons. For example, if $\mathrm{FOV} = 180°$ and the used resolution is $1°$, the NN will have 181 neurons in its output. This means that each output neuron represents a possible angle. The output value of these neurons can vary between 0 and 1. They represent the neural network's confidence that a given target exists (or does not) at that particular angle. The output y from ResNet represents its reliability distribution (confidence) at all possible angles within a given FOV and resolution.

Further, the output y from the 1D ResNet passes through the post-processing step, which selects the $2N$ most probable angles $\hat{\theta} = \hat{\theta}_1, ..., \hat{\theta}_{2N}$. Note that the number of selected angles varies with the number of incoming angles N. For example, when solving the DOA for 3 targets, the post-processing step will select 6 probable angles from the output y. The $2N$ was chosen empirically, as it produced the best results.

Finally, the last step is the MLE algorithm which receives $2N$ angles $(\hat{\theta})$ and computes $\binom{2N}{N}$ combinations, searching for the one that results in the least amount of residual error, thus obtaining the N estimated angles $(\tilde{\theta})$ in the output.

Another way to demonstrate the integration between ResNet and MLE is through Fig. 3. In this example, we have 4 distinct targets. Here, it is possible to observe the neural network's output y. The strongest peaks, marked with a red asterisk, are the ones selected by the post-processing step $\hat{\theta} = \hat{\theta}_1, ..., \hat{\theta}_8$. They represent the angles where ResNet is most confident that a target exists. The output from Deep-MLE is marked with blue circles, and it represents the estimated angles by our framework $\tilde{\theta} = \tilde{\theta}_1, ..., \tilde{\theta}_4$. The dashed vertical line represents the actual angles $\theta = \theta_1, ..., \theta_4$. Note how the most strong peak (at 50°) was not selected by Deep-MLE. This means that the MLE step corrected the ResNet output. In other words, in this example, Deep-MLE correctly estimated the angles, using a fraction of the computational requirement of a complete MLE, by correcting a prediction from the neural network.

The operational complexity of our Deep-MLE is the combination of the complexities between 1D ResNet and MLE, where the number of angles that the MLE step searches is $G = 2N$. Therefore, the complexity of our Deep-MLE is given by: $O_{\mathrm{dMLE}}(1.68 \times 10^6 + M^3\binom{2N}{N})$, hence, the computational complexity of 4 targets drops from 2.2×10^{10} (MLE) to 1.72×10^6 (Deep-MLE). This is more than ten thousand times less complex.

4. EVALUATION AND RESULTS

For evaluating our work, we have utilized a few metrics. Root Mean Square Error (RMSE), accuracy (percentage of correctly estimated DOAs), and outliers (percentage of incorrectly estimated DOAs by more than 5 degrees) were the metrics utilized by us to evaluate our results.

As many researchers before [8, 11], we have used Root Mean Square Error (RMSE) as a metric. The RMSE used by us is given by:

$$\mathrm{RMSE} = \frac{1}{L}\sum_{l=1}^{L}\sqrt{\frac{1}{K \times N}\sum_{k=1}^{K}\sum_{n=1}^{N}(\tilde{\theta}_{n,k,l} - \theta_{n,k})^2}\,, \quad (4)$$

where K is the number of samples, N is the number of DOAs, L is the number of Monte-Carlos trials, $\tilde{\theta}_{n,k,l}$ is the estimated angle and $\theta_{n,k}$ is the actual angle.

Although RMSE is a reliable metric, it does not show everything about a given scenario. For example, a high RMSE can be due to a single very incorrectly estimated angle. In reality, it does not matter if an estimated angle is off by 10 or 50 degrees since both answers are equally wrong. Therefore, we have used two other metrics as well. What we call accuracy is the percentage of correctly estimated angles within the resolution. If the resolution of $1°$ is used, anything below 0.5° should be considered a correct estimation. What we call outliers is the percentage of incorrectly estimated angles that are off by more than $5°$.

Before our experiments, we have trained our 1D ResNet using 50 million of generated data points, each data point con-

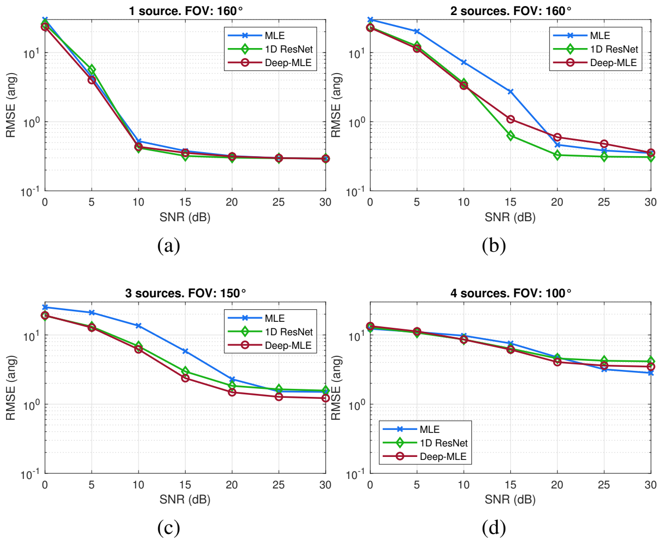

Fig. 4: RMSE vs. SNR for MLE, ResNet, and our proposed Deep-MLE between 1 and 4 sources.

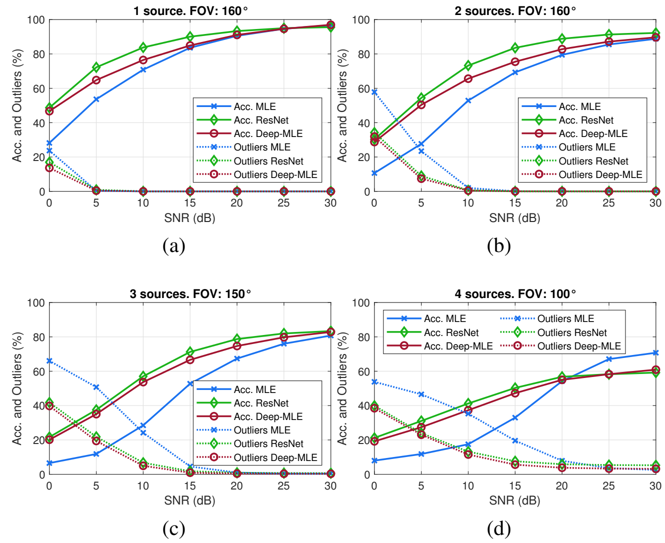

Fig. 5: Accuracy and Outliers vs. SNR for MLE, ResNet, and our proposed Deep-MLE between 1 and 4 sources.

taining between 0 and 5 sources, SNR (Signal-to-Noise Ratio) ranging between 0 and 30 dB, and normalized signal amplitude varying between 0.5 and 1. Although 50 million data points might seem like a lot of data, they represent an insignificantly small number of every possible combination between targets, SNR, and amplitude. Therefore, the neural network needed to learn how to generalize the training data.

Our experiment results can be seen in Fig. 4. We can observe that our Deep-MLE tends to follow the ResNet performance for low SNR levels while following the performance of MLE for high SNR situations. In some cases, our framework can outperform both of the other techniques. For one source, Deep-MLE was the best one for low SNR and had a similar performance to both MLE and ResNet for high SNR. For two sources, Deep-MLE also had the best performance for noisy situations, and it performed slightly worse than the others when in a low noise environment. For three sources, our framework always performed best. And for four targets, it closely followed the performance of ResNet for low SNR and then followed MLE for high SNR.

We can also analyze the results of Fig. 4 by taking the average RMSE of all experiments - for low and high noise - and calculating the difference between them. For low SNR, the difference of averages RMSE between Deep-MLE and MLE is $-3.4°$, while between Deep-MLE and 1D ResNet is $-0.3°$. For high SNR, the difference of averages RMSE between Deep-MLE and MLE is $-0.06°$, while between Deep-MLE and 1D ResNet is $-0.2°$. Hence, in general, Deep-MLE has a better performance than the tested models that use only MLE or 1D ResNet.

Sequentially, we have repeated the same experiment but changed the metrics for accuracy and outliers, as seen in Fig. 5. The solid line represents the accuracy within resolution. In other words, the percentage of correctly estimated angles within the resolution. The dotted line represents the percentage of targets that were incorrectly estimated by more than 5°. The same trends observed in Fig. 4 can also be seen in Fig. 5. It is important to note that, for most of these scenarios, we can observe that Deep-MLE has the least number of outliers (thus better RMSE results), even though 1D ResNet can be more accurate for most situations.

5. CONCLUSION

In this paper, we have proposed a novel approach to solve the DOA estimation problem by fusing the classical MLE approach with a deep-learning-based approach. Using a 1D Residual Neural Network, we have got a model that has a low computational complexity compared to the classical MLE while providing more accurate results than pure neural network models. We have experimented with various SNR levels, and different numbers of sources. The results of our simulations showed that our framework generally performs better than MLE or 1D ResNet. In short, our model is robust to changes in noise, being a viable solution for the DOA estimation problem. However, this framework also comes with disadvantages. The training part is very computational complex and requires a fast computer. Retraining is also necessary if there are changes to the model, such as changes in the antennas' position. In addition, it is also necessary to store the trained neural network.

For future work, we would like to test Deep-MLE using measured data instead of generated data.

6. REFERENCES

[1] A. G. Raj and J. H. McClellan, "Super-resolution DOA Estimation for Arbitrary Array Geometries Using a Single Noisy Snapshot," *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 2019-May, no. 1, pp. 4145–4149, 2019.

[2] E. Aboutanios, A. Hassanien, M. G. Amin, and A. M. Zoubir, "Fast Iterative Interpolated Beamforming for Accurate Single-Snapshot DOA Estimation," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 4, pp. 574–578, 2017.

[3] M. L. Lima de Oliveira and M. J. G. Bekooij, "Deep convolutional autoencoder applied for noise reduction in range-Doppler maps of FMCW radars," *IEEE International Radar Conference (RADAR 2020)*, pp. 630–635, 2020.

[4] A. Fuchs, J. Rock, M. Toth, P. Meissner, and F. Pernkopf, "Complex-valued Convolutional Neural Networks for Enhanced Radar Signal Denoising and Interference Mitigation," in *IEEE Radar Conference*, no. 17774193. IEEE, 2021.

[5] K. Aziz, E. De Greef, M. Rykunov, A. Bourdoux, and H. Sahli, "Radar-camera Fusion for Road Target Classification," *IEEE International Radar Conference (RADAR 2020)*, vol. 2020-Septe, 2020.

[6] Z. Zhang, Z. Tian, M. Zhou, and S. Member, "Latern : Dynamic Continuous Hand Gesture Recognition Using FMCW Radar Sensor," *IEEE Sensors Journal*, vol. 18, no. 8, pp. 3278–3289, 2018.

[7] M. L. Lima De Oliveira and M. J. G. Bekooij, "Generating Synthetic Short-Range FMCW Range-Doppler Maps Using Generative Adversarial Networks and Deep Convolutional Autoencoders," *IEEE Radar Conference (RadarConf20)*, pp. 1–6, 2020.

[8] A. M. Elbir, "DeepMUSIC: Multiple Signal Classification via Deep Learning," *IEEE Sensors Letters*, vol. 4, no. 4, 2020.

[9] C. Liu, W. Feng, H. Li, and H. Zhu, "Single Snapshot DOA Estimation Based on Spatial Smoothing MUSIC and CNN," *IEEE International Conference on Signal Processing, Communications and Computing*, no. 3, 2021.

[10] J. Fuchs, R. Weigel, and M. Gardill, "Single-snapshot direction-of-arrival estimation of multiple targets using a multi-layer perceptron," *2019 IEEE MTT-S International Conference on Microwaves for Intelligent Mobility, ICMIM 2019*, pp. 0–3, 2019.

[11] O. Bialer, N. Garnett, and T. Tirer, "Performance Advantages of Deep Neural Networks for Angle of Arrival Estimation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 3907–3911.

[12] M. Schoor and B. Yang, "High-resolution angle estimation for an automotive FMCW radar sensor," *Proceedings International Radar Symposium (IRS)*, 2007.

[13] Z. Szalay and L. Nagy, "ULA and MRA beamforming using conventional and particle filter based algorithms," *International Conference on Applied Electromagnetics and Communications (ICECom)*, pp. 8–11, 2013.

[14] P. Häcker and B. Yang, "Single snapshot DOA estimation," *Advances in Radio Science*, vol. 8, pp. 251–256, 2010.

[15] H. L. Van Trees, *Optimum Array Processing*, 2002, vol. 4.

[16] B. M. Radich and K. M. Buckley, "Single-snapshot DOA estimation and source number detection," *IEEE Signal Processing Letters*, vol. 4, no. 4, pp. 109–111, 1997.

[17] J. Zeng, Z. He, and B. Liu, "Maximum likelihood DOA estimation using particle swarm optimization algorithm," *CIE International Conference of Radar Proceedings*, no. 5, pp. 0–3, 2007.

[18] P. Stoica and K. C. Sharman, "Maximum Likelhood Methods for Direction-of- Arrival Estimation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 7, pp. 1132–1143, 1990.

[19] H. Krim and M. Viberg, "Two Decades of Array Signal Processing Research," *IEEE Signal Processing Magazine*, pp. 67–94, 1996.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

[21] P. Courrieu, "Fast Computation of Moore-Penrose Inverse Matrices," *Neural Information Processing - Letters and Reviews*, vol. 8, no. 2, pp. 25–29, 2008.