JOINT DIRECTION OF ARRIVAL ESTIMATION AND SOURCE ENUMERATION USING TRANSFORMER FOR SPARSE LINEAR ARRAYS WITH ROBUSTNESS TO SENSOR MALFUNCTIONS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

BURAK HAYATI MUSLU

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

AUGUST 2024

Approval of the thesis:

JOINT DIRECTION OF ARRIVAL ESTIMATION AND SOURCE ENUMERATION USING TRANSFORMER FOR SPARSE LINEAR ARRAYS WITH ROBUSTNESS TO SENSOR MALFUNCTIONS

submitted by **BURAK HAYATI MUSLU** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Naci Emre Altun
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. İlkay Ulusoy
Head of Department, **Electrical and Electronics Engineering** _____

Prof. Dr. Tolga Çiloğlu
Supervisor, **Electrical and Electronics Engineering, METU** _____

Examining Committee Members:

Prof. Dr. Umut Orguner
Electrical and Electronics Engineering, METU _____

Prof. Dr. Tolga Çiloğlu
Electrical and Electronics Engineering, METU _____

Prof. Dr. Emre Aktaş
Electrical and Electronics Engineering, Hacettepe University _____

Assoc. Prof. Dr. S. Figen Öktem Seven
Electrical and Electronics Engineering, METU _____

Assoc. Prof. Dr. A. Melda Yüksel Turgut
Electrical and Electronics Engineering, METU _____

Date:28.08.2024

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Burak Hayati Muslu

Signature :

ABSTRACT

JOINT DIRECTION OF ARRIVAL ESTIMATION AND SOURCE ENUMERATION USING TRANSFORMER FOR SPARSE LINEAR ARRAYS WITH ROBUSTNESS TO SENSOR MALFUNCTIONS

Muslu, Burak Hayati
M.S., Department of Electrical and Electronics Engineering
Supervisor: Prof. Dr. Tolga Çiloğlu

August 2024, 184 pages

Direction of arrival (DOA) estimation and source enumeration are crucial tasks in modern array processing with a wide range of applications. For applying these tasks, sparse linear arrays have been of great interest in the recent years due to their advantages over other array types. However, these arrays are more vulnerable to sensor malfunctions because of their coarray configuration. Sensor malfunctions in these arrays may cause performance degradation in the mentioned tasks if the failures are not handled. Handling the failures by repairing/replacing the sensors is challenging in practice. Therefore, there arises a need of an estimation method that is robust to sensor malfunctions.

A transformer-based DOA estimation and source enumeration method is proposed to handle the sensor number/configuration variations occurring due to sensor malfunctions. Existing data-driven solutions cannot generalize to such variations due their input formulation and structural design. The proposed method exploits coarray-based input formulation and positional encoding and attention blocks for learning spatial arrangement of the sensors along with the nonlinear relationship between the mea-

surements and source direction. Experimental results demonstrate that it achieves error reduction in DOA estimation performed for intact array especially for low SNR levels and snapshot numbers. It shows comparable performance to the state-of-the-art methods for high SNR and snapshot regime. Introduction of sensor malfunctions causes less degradation in the performance of the proposed method compared to others which have reasonable processing time. The robustness to sensor malfunctions is complemented by performance improvement in source enumeration particularly for low SNR and snapshot regions.

Keywords: Direction of Arrival Estimation, Source Enumeration, Sparse Array, Sensor Malfunction, Deep Learning

ÖZ

SEYREK DOĞRUSAL DİZİLER İÇİN SENSÖR BOZULMALARINA DAYANIKLI DÖNÜŞTÜRÜCÜ TABANLI ORTAK GELİŞ AÇISI VE KAYNAK SAYISI KESTİRİMİ

Muslu, Burak Hayati
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Tolga Çiloğlu

Ağustos 2024 , 184 sayfa

Geliş açısı ve kaynak sayısı kestirimi modern dizi işleme içerisinde çeşitli sayıda uygulamaları bulunan önemli işlemlerdir. Diğer dizi çeşitlerine göre avantajları sebebiyle seyrek doğrusal diziler son yıllarda bu işlemleri uygulamak için büyük ilgi görmektedir. Fakat bu diziler "coarray" konfigürasyonları sebebiyle sensör bozulmalarına karşı daha hassastırlar. Bu dizilerde sensör bozulmaları eğer düzeltilmezse bahsi geçen işlemlerde performans düşmelerine sebep olabilmektedir. Bozulmaları sensörleri tamir ederek veya değiştirerek düzeltmek uygulamada zorlayıcıdır. Bu yüzden sensör bozulmalarına karşı dayanıklı kestirim yöntemlerine ihtiyaç oluşmuştur.

Sensör bozulmaları sebebiyle ortaya çıkan sensör sayısı/konfigürasyonu farklılıklarının üstesinden gelmek için dönüştürücü tabanlı bir geliş açısı ve kaynak sayısı kestirimi yöntemi önerilmiştir. Mevcut veri güdümlü çözümler girdi formülasyonları ve yapısal tasarımları sebebiyle bu tarz farklılıklar için genellenebilir performans gösterememektedir. Önerilen yöntem, sensörlerin konumsal düzenlerini ve ölçümler ile kaynak açısı arasındaki doğrusal olmayan ilişkiyi öğrenmek için "coarray" tabanlı

girdi formülasyonundan ve konumsal kodlama ve dikkat bloklarından faydalanmaktadır. Deneysel sonuçlar, özellikle düşük SNR seviyeleri ve ölçüm sayıları için sağlam dizide gerçekleştirilen geliş açısı kestiriminde hata azaltımına ulaşıldığını göstermektedir. Yüksek SNR ve ölçüm sayısı rejimi için en gelişkin yöntemlerle benzer performans gösterilmektedir. Sensör bozulmalarının ortaya çıkması, makul işlem süresine sahip diğer yöntemlere kıyasla önerilen yöntemin performansında daha az bozulmaya neden olmuştur. Sensör arızalarına karşı dayanıklılık, özellikle düşük SNR ve ölçüm sayısı seviyelerinde kaynak sayısı kestirimindeki performans iyileştirmesi ile desteklenmektedir.

Anahtar Kelimeler: Geliş Açısı Kestirimi, Kaynak Sayısı Kestirimi, Seyrek Dizi, Sensör Bozulması, Derin Öğrenme

To my family...

ACKNOWLEDGMENTS

Firstly, I am deeply grateful to my supervisor, Prof. Tolga Çiloğlu, for his invaluable guidance, encouragement, and constructive feedback. His expertise and unwavering support have been crucial in shaping this research and bringing it to completion.

I would like to express my sincere gratitude to the members of my thesis committee, Prof. Dr. Umut Orguner, Prof. Dr. Emre Aktaş, Assoc. Prof. Dr. S. Figen Öktem Seven, and Assoc. Prof. Dr. A. Melda Yüksel Turgut. Their invaluable feedback and insightful suggestions have greatly enhanced the quality of this work.

I would like to also thank my friends and colleagues, for their support and for providing a stimulating environment for academic and intellectual discussions. Their camaraderie and encouragement have been a source of motivation.

Lastly, I would like to express my heartfelt thanks to my family for their unconditional love, patience, and understanding throughout this journey. Their support has been a constant source of strength and motivation.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS

AdaGrad	Adaptive Gradient Algorithm
Adam	Adaptive Moment Estimation
AIC	Akaike Information Criterion
AREG	Accumulated Ratio of Eigenvalues Gaps
BIC	Bayesian Information Criterion
CA	Coprime Array
CNN	Convolutional Neural Network
COMET	Covariance Matching Techniques
CRN	Covariance Reconstruction Network
DAE	Denoising Autoencoder
DAEN	Direction of Arrival Estimation Network
DNN	Deep Neural Network
DOA	Direction of Arrival
ELU	Exponential Linear Unit
ESPRIT	Estimation of Signal Parameters via Rotational Invariance Techniques
EVD	Eigenvalue Decomposition
FCL	Fully Connected Layer
GAN	Generative Adverserial Network
GNN	Graph Neural Network
LASSO	Least Absolute Shrinkage and Selection Operator
LSTM	Long Short-Term Memory
MDL	Minimum Description Length

Min-Norm	Minimum Norm Method
MLP	Multilayer Perceptron
MRA	Minimum Redundant Array
MSE	Mean Square Error
MUSIC	Multiple Signal Classification
NE	Nested Array
NN	Neural Network
OMP	Orthogonal Matching Pursuit
RBF	Radial Basis Function
ReLU	Rectified Linear Unit
RMSE	Root Mean Square Error
RMSProp	Root Mean Square Propagation
RNN	Recurrent Neural Network
SEN	Source Enumeration Network
SGD	Stochastic Gradient Descent
SNR	Signal-to-noise Ratio
SS	Spatial Smoothing
SORTE	Second Order Statistic of the Eigenvalues
SVD	Singular Value Decomposition
SVM	Support Vector Machines
SVR	Support Vector Regression
SVT	Singular Value Thresholding
T-GANE	Threshold for Gap of Normalized Eigenvalues
ULA	Uniform Linear Array

CHAPTER 1

INTRODUCTION

Sensor array signal processing is an important area of study which takes place in applications such as radar, sonar, direction finding, communications, seismology, radio astronomy, and medical diagnosis [1]. It exploits the data collected using several sensors for performing specific tasks [2]. Sensors may form various geometries in an array, among which sparse arrays are of great interest in the past two decades [3]. Sparse arrays are constructed by placing the sensors in a non-uniform manner. The ability to localize more sources than the number of sensors made these array types the focus of lots of studies [4].

Direction of arrival (DOA) estimation is one of the forefront tasks belonging to sensor array signal processing due to its numerous application areas [5]. It involves estimation of the direction of impinging source signals by utilizing temporally and spatially sampled sensor data. DOA estimation methods operate on the received signals or their second order statistics. The angle estimates generated by these methods might be based on predefined grid angles, which is called as on-grid approach; might be refined estimate of predefined grid angles, which is called as off-grid approach; or might be direct angle, which is called as gridless approach. These methods can be grouped into three primary categories; model-based, data-driven and hybrid. In the recent years, data-driven methods achieved great success due to the advancements in deep learning field [6].

Deep learning methods are comprised of neural networks which can approximate nonlinear functions. They apply nonlinear mapping between the input and target variable and this nonlinear mapping is learned using vast amount of data. According to universal approximation theorem, a deep network with sufficient number of neurons can

approximate any nonlinear function [7]. In the domain of DOA estimation, the nonlinear relationship between the received signals (or their second order statistics) and source direction is learned using deep learning. Multilayer perceptron (MLP), convolutional neural network (CNN), recurrent neural network (RNN) and long short-term memory (LSTM) [8] are some of the preferred deep networks for DOA estimation task [6].

Similar to DOA estimation, source enumeration has also an important place in sensor array signal processing due to the fact that the number of existing sources is generally unknown a priori. An inaccurate source number estimation degrades the performance in DOA estimation as well [9]. Existing source enumeration techniques exploit information theoretic criteria or eigenvalues of the covariance matrix of the received signals. As in DOA estimation, data-driven methods are employed for source enumeration too and deep learning models are preferred in some of the studies [10], [11].

In practice, sensor malfunctions may occur which lead to performance degradation in DOA estimation and source enumeration if the failures are not handled. Sparse arrays are vulnerable against such malfunctions due to their coarray configuration. Sensor malfunctions may lead to changes in the sensor number/configuration of physical array as well as difference coarray. Handling the failures by repairing/replacing the sensors is challenging in practice [12], [13]. For this reason, there arises a need for development of estimation methods which are not affected by malfunctions or which mitigates the effect of malfunctions.

Most of the existing data-driven solutions cannot generalize in case of sensor malfunctions due to their input formulation and network structure. Therefore, these methods need retraining using data collected from array with missing sensors. The ones that are able to generalize are not designed to handle unknown number of sources.

In this thesis, we propose a transformer-based deep learning network for DOA estimation and source enumeration which can function in case of sensor malfunctions. The reason for employing transformer is that it can process sequential data with varying length and embed the positional information of each sequence step. This ability is utilized by formulating the input in sequential form based on difference coarray. The performance of this method is investigated for different types of sparse linear arrays

in the presence of sensor malfunctions and unknown number of sources.

Contributions of this thesis can be summarized as follows:

- For the first time, a transformer-based DOA estimation method with coarray formulated input is proposed. This input formulation and positional encoding ability of transformer model enable the network to generalize for arrays with different sensor numbers/configurations without the requirement of retraining.
- Covariance reconstruction network, which is the first stage of the proposed method, is developed to refine the noisy covariance matrix of the received sensor signals. The reconstruction ability of this network is illustrated with examples and demonstrated with numerical comparisons.
- DOA estimation network, which is the second stage of the proposed method, is developed to generate on-grid DOA angle estimates. Performance comparisons are done for different sparse array types and in the presence of sensor malfunctions. Performance improvement especially for low SNR levels and snapshot numbers is demonstrated. Robustness of the proposed method against sensor malfunctions is shown along with its feasible processing time.
- Source enumeration network, which is the third stage of the proposed method, is developed to perform source number estimation for the joint case of sensor malfunctions and unknown number of sources. The increase in the source number estimation accuracy that is achieved by the proposed method is demonstrated through simulations.

The outline of the thesis is as follows:

- In Chapter 2, the mathematical model for array signals is derived and related assumptions are stated. Information about sparse arrays is provided. The existing DOA estimation and source enumeration methods are presented and described. Sensor malfunctions and their effects are explained and existing studies regarding it are summarized. The scope of thesis in terms of sensor malfunctions is stated.

- In Chapter 3, background information about deep learning is provided. The proposed method is explained with details.
- In Chapter 4, the performance analysis of the proposed method is carried out for different types of sparse arrays.
- In Chapter 5, the performance analysis of the proposed method is executed for the cases of sensor malfunctions.
- In Chapter 6, the performance analysis of the proposed method is realized for the joint presence of sensor malfunctions and unknown number of sources.
- In Chapter 7, concluding remarks and future research directions are stated.

The following notation is adopted in this thesis: x denotes a scalar. $\boldsymbol{x}$ is a vector. $\boldsymbol{X}$ is a matrix. $\mathbb{X}$ denotes a set. The transpose operation is $(.)^T$, conjugate operation is $(.)^*$, Hermitian conjugate operation is $(.)^H$ and orthogonal complement operation is $(.)^\perp$.

The following conventions are accepted in this thesis: Azimuth θ is defined as the angle from positive x axis. Elevation ϕ is defined as the angle from positive y axis.

CHAPTER 2

SIGNAL MODEL AND PROCESSING METHODS

In this chapter, the mathematical model for array signals is derived and assumptions are stated. Sparse arrays, which are the main focus of this study, are introduced and theoretical background and analysis of these array types are provided. Direction of arrival estimation and source enumeration methods existing in the literature are described. Sensor malfunctions and their possible effects on the array are explained. Lastly, a brief summary about the studies on sensor malfunctions is provided and the scope of this study in terms of sensor malfunctions is set.

2.1 Array Signal Model

Consider a uniform linear array (ULA) with M sensors. There exist N narrow-band sources in the far-field and their DOA angles are represented by $\boldsymbol{\Theta}_1, \boldsymbol{\Theta}_2, ..., \boldsymbol{\Theta}_N$ where $\boldsymbol{\Theta}_n = [\theta_n, \phi_n]$ is the vector containing azimuth and elevation angle of the n^{th} source respectively.

The plane wave corresponding to n^{th} source which propagates in a certain direction has the direction vector

$$\boldsymbol{g_n} = \begin{bmatrix} \cos\theta_n \cos\phi_n \\ \sin\theta_n \cos\phi_n \\ \sin\phi_n \end{bmatrix} \tag{2.1}$$

This signal arrives at the sensor m with a time delay

$$\tau_{n,m} = -\frac{\boldsymbol{g_n}^T \boldsymbol{p_m}}{c} \tag{2.2}$$

where $\boldsymbol{p_m} = [x_m, y_m, z_m]^T$ is the position vector of m^{th} sensor and c is speed of the wave.

Assuming that the first sensor is at the origin, array is placed over x-axis with an inter-element distance of d and elevation of all sources is $0°$. A system diagram for the array and sources is given in Figure 2.1.

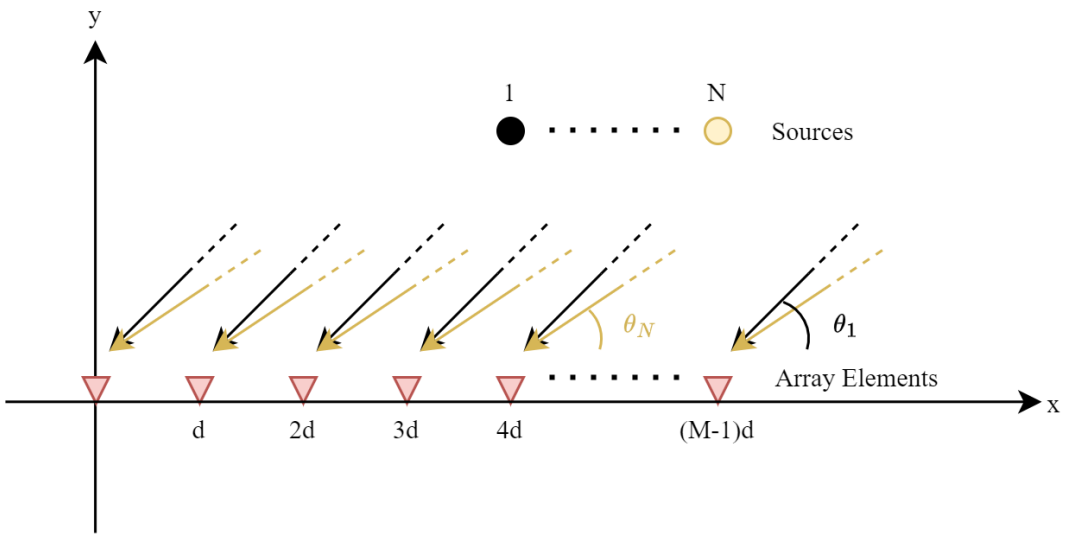

Figure 2.1: System diagram for the array and sources

Array steering vector for n^{th} source is

$$\boldsymbol{a}(\theta_n, \phi_n) = \begin{bmatrix} e^{j\omega\tau_{n,1}} \\ e^{j\omega\tau_{n,2}} \\ \vdots \\ e^{j\omega\tau_{n,M}} \end{bmatrix} = \begin{bmatrix} e^{-j\frac{2\pi}{\lambda}\boldsymbol{g_n}^T\boldsymbol{p_1}} \\ e^{-j\frac{2\pi}{\lambda}\boldsymbol{g_n}^T\boldsymbol{p_2}} \\ \vdots \\ e^{-j\frac{2\pi}{\lambda}\boldsymbol{g_n}^T\boldsymbol{p_M}} \end{bmatrix} = \begin{bmatrix} 1 \\ e^{-j\frac{2\pi}{\lambda}d\cos\theta_n} \\ e^{-j\frac{2\pi}{\lambda}2d\cos\theta_n} \\ \vdots \\ e^{-j\frac{2\pi}{\lambda}(M-1)d\cos\theta_n} \end{bmatrix} \tag{2.3}$$

where λ is the wavelength and ω is the center frequency of the source signal and $\omega = \frac{2\pi}{\lambda}c$. Steering vector denotes the spatial response of an array to a plane wave arriving from a specific direction and it encodes phase shifts occurring on the signal while traversing the array elements.

Array steering matrix $\boldsymbol{A}(\boldsymbol{\Theta})$, which contains steering vectors for all N sources as columns, can be written as

$$\boldsymbol{A}(\Theta) = \begin{bmatrix} 1 & 1 & \dots & 1 \\ e^{-j\frac{2\pi}{\lambda}d\cos\theta_1} & e^{-j\frac{2\pi}{\lambda}d\cos\theta_2} & \dots & e^{-j\frac{2\pi}{\lambda}d\cos\theta_N} \\ e^{-j\frac{2\pi}{\lambda}2d\cos\theta_1} & e^{-j\frac{2\pi}{\lambda}2d\cos\theta_2} & \dots & e^{-j\frac{2\pi}{\lambda}2d\cos\theta_N} \\ \vdots & \vdots & \ddots & \vdots \\ e^{-j\frac{2\pi}{\lambda}(M-1)d\cos\theta_1} & e^{-j\frac{2\pi}{\lambda}(M-1)d\cos\theta_2} & \dots & e^{-j\frac{2\pi}{\lambda}(M-1)d\cos\theta_N} \end{bmatrix} \tag{2.4}$$

Then, the received signals by the sensors at snapshot t can be formulated as

$$\boldsymbol{y}(t) = \boldsymbol{A}(\boldsymbol{\Theta})\boldsymbol{s}(t) + \boldsymbol{e}(t) \tag{2.5}$$

where $\boldsymbol{s}(t) = [s_1(t), s_2(t), ..., s_N(t)]^T$ is the source signals and $\boldsymbol{e}(t) = [e_1(t), e_2(t), ..., e_M(t)]^T$ represents the noise at each sensor.

The received signal formula (2.5) can be extended to cover T snapshots by

$$\boldsymbol{Y} = \boldsymbol{A}(\boldsymbol{\Theta})\boldsymbol{S} + \boldsymbol{E} \tag{2.6}$$

where $\boldsymbol{Y}$ is $M \times T$ matrix, $\boldsymbol{S}$ is $N \times T$ matrix and $\boldsymbol{E}$ is $M \times T$ matrix whose columns carrying per-snapshot values as

$$\boldsymbol{Y} = [\boldsymbol{y}(1), \boldsymbol{y}(2), ..., \boldsymbol{y}(T)] \tag{2.7}$$

$$\boldsymbol{S} = [\boldsymbol{s}(1), \boldsymbol{s}(2), ..., \boldsymbol{s}(T)] \tag{2.8}$$

$$\boldsymbol{E} = [\boldsymbol{e}(1), \boldsymbol{e}(2), ..., \boldsymbol{e}(T)] \tag{2.9}$$

Equation (2.6) models the sampling of sensor array both in temporal and spatial domain. Both domains may experience aliasing effects. For preventing time aliasing, Nyquist theorem [14] must be followed which states

$$f_s \geq 2f_c \tag{2.10}$$

where f_s is the sampling frequency and f_c is the carrier frequency. Analogously, a similar criterion is valid for spatial domain in which spatial Nyquist theorem must be obeyed to have no spatial aliasing. Spatial aliasing is encountered when $\boldsymbol{A}(\boldsymbol{\Theta})$ is not unique for different $\boldsymbol{\Theta}$ values. For having a unique $\boldsymbol{A}(\boldsymbol{\Theta})$, it is required to have

$$\frac{2\pi}{\lambda} d \cos\theta \leq \pi \tag{2.11}$$

in which $\max(|\cos\theta|) = 1$. Then, Equation (2.11) reduces to

$$d \leq \frac{\lambda}{2} \tag{2.12}$$

In the remainder of the thesis, the following assumptions are made regarding the signals:

- The elevation of the source signals is 0°.
- Source signals are narrowband and in the far-field.
- Source signals are uncorrelated.
- Noise signals are independent and identically distributed, zero-mean, temporally and spatially white, complex Gaussian and uncorrelated with source signals.

2.2 Sparse Arrays

Uniform linear arrays (ULA) are commonly adopted array types due to spatial Nyquist theorem which introduces restrictions on the inter-element distances [15]. However, a primary drawback of ULA is that the number of sources that can be resolved is limited by the number of sensors [4]. This limitation is related to degree of freedom that the array provides.

Before explaining degree of freedom, it is more appropriate to define difference coarray concept. Difference coarray is the set

$$\mathbb{D} = \{p_i - p_j\} \quad \forall i, j \in [1, 2, ..., M] \tag{2.13}$$

where M is the number of sensors and p_i indicates sensor position. Each difference value in $\mathbb{D}$ is called as lag. The weight of a lag refers to how frequently it appeared in $\mathbb{D}$. Redundancy of an array is identified by the amount of weights that are larger than 1 except that of zero lag. The lags with weights of 0 are called as holes and degree of freedom is defined as the number of unique lags with non-zero weights [1].

An example difference coarray for a ULA with $M = 8$ is illustrated in Figure 2.2. No hole is observed in the coarray, therefore the degree of freedom is equal to the number of sensors M. However, most of the weights are larger than 1 and this introduces redundancy in the coarray.

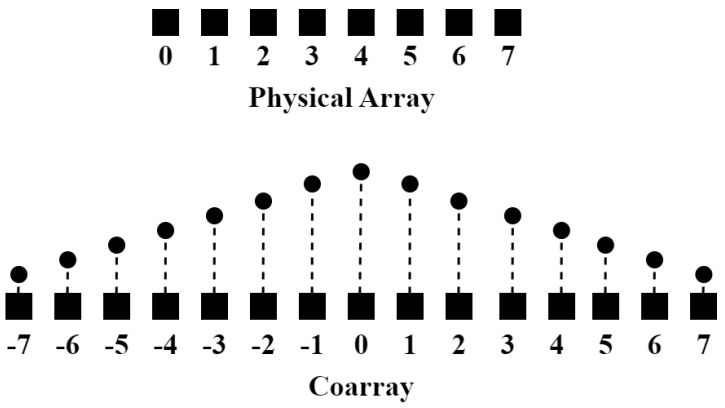

Figure 2.2: Physical array and difference coarray of ULA with 8 elements. (Rectangle represents presence of the element while cross indicates its absence. The number of dashes in the dashed lines represent the weights for each lag)

There arises a question whether the same degree of freedom can be achieved using less number of sensor. Sparse arrays are the answer of this question with their ability to provide a degree of freedom of $\mathcal{O}(M^2)$ where ULA's degree of freedom is $\mathcal{O}(M)$ [3]. Figure 2.3 depicts a sparse array with its coarray which can provide the same degree of freedom with less number of sensors compared to ULA in Figure 2.2. Therefore, sparse arrays are able to resolve more sources than the number of sensors unlike ULA.

Another advantage of the sparse arrays over ULA is the larger aperture with the same

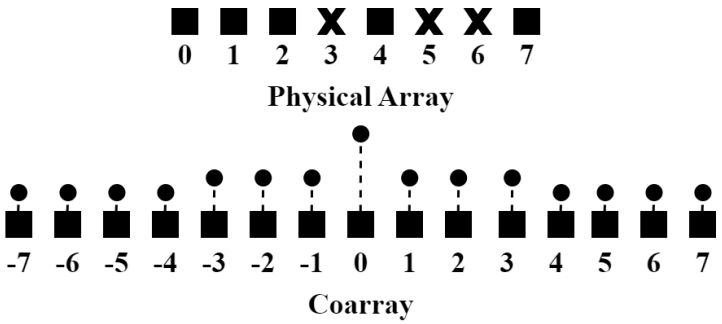

Figure 2.3: Physical array and difference coarray of sparse linear array with 5 elements. (Rectangle represents presence of the element while cross indicates its absence. The number of dashes in the dashed lines represent the weights for each lag)

number of sensors [15]. This advantage can be observed when sparse array in Figure 2.3 is compared with its ULA counterpart with the same number of sensors. The aperture of the sparse array is 7 while that of ULA with 5 elements is 4. Having larger aperture provides better resolution to sparse array compared to ULA [16].

Achieving the same aperture and degree of freedom with less number of sensors provides sparse array another benefit. The hardware complexity and cost are reduced compared to ULA.

There are various sparse array geometries such as minimum redundant array (MRA) [17], minimum hole array [18], nested array (NA) [19] and coprime array (CA) [20]. Sections 2.2.1, 2.2.2 and 2.2.3 gives information about minimum redundant array, nested array and coprime array respectively.

2.2.1 Minimum Redundant Array

Minimum redundant array is designed such that the weights of spatial lags in the difference coarray is minimized [1]. They contain no holes in the coarray and obtain the largest aperture compared to other linear arrays [17,21]. A drawback of minimum redundant array is that they do not have closed-form formulation and they cannot be systemically generated and analyzed [15].

An example minimum redundant array and corresponding difference coarray are shown

in Figure 2.4.

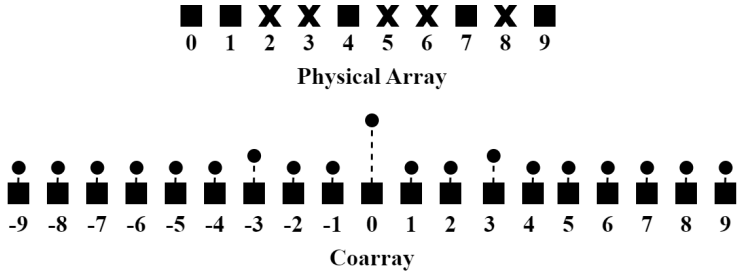

Figure 2.4: Physical array and difference coarray of a minimum redundant array. (Rectangle represents presence of the element while cross indicates its absence. The number of dashes in the dashed lines represent the weights for each lag)

For observing the pros and cons of minimum redundant array, its performance is compared with ULA with the same number of sensor and with the same aperture separately which are shown in Figure 2.5. MUSIC [22], a well-known and successful example of subspace-based DOA estimation methods, is used in these comparisons. Capon beamforming spectrums are utilized for enriching the comparisons further.

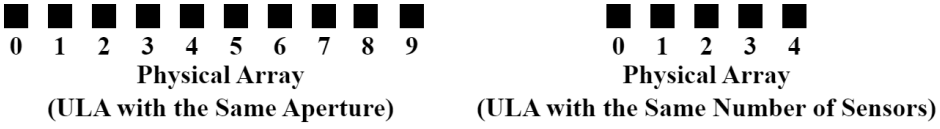

Figure 2.5: ULAs with the same aperture and number of sensors as minimum redundant array. (Rectangle represents presence of the element while cross indicates its absence)

Firstly, RMSE values obtained by minimum redundant array and above-referred ULAs in DOA estimation of single source are compared for different SNR levels in Figure 2.6. It can be observed that the minimum errors are achieved by ULA with the same aperture as MRA. This is due to additional sensors in ULA. However, minimum redundant array show better performance than ULA with the same number of sensors owing its larger aperture. An example Capon spectrum generated by these arrays are given in Figure 2.7. The highest peak is obtained by ULA with the same aperture as MRA owing to more sensors that it possesses. Minimum redundant array has narrower peak compared to ULA with the same number of sensors due to its larger

aperture. However, it has higher sidelobes.

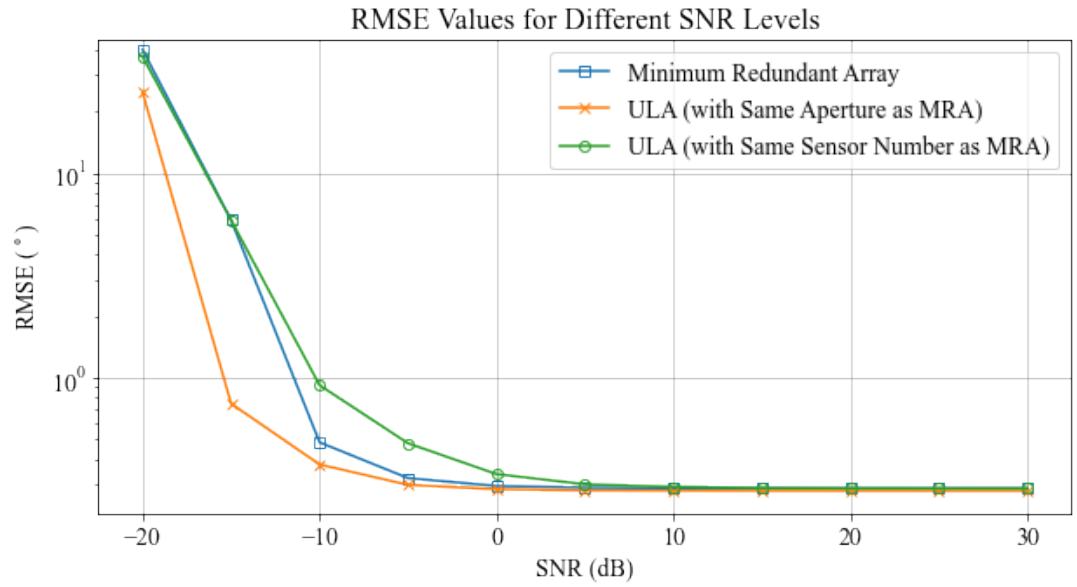

Figure 2.6: RMSE obtained by minimum redundant array and ULAs with the same aperture and number of sensors for different SNR levels in single source case

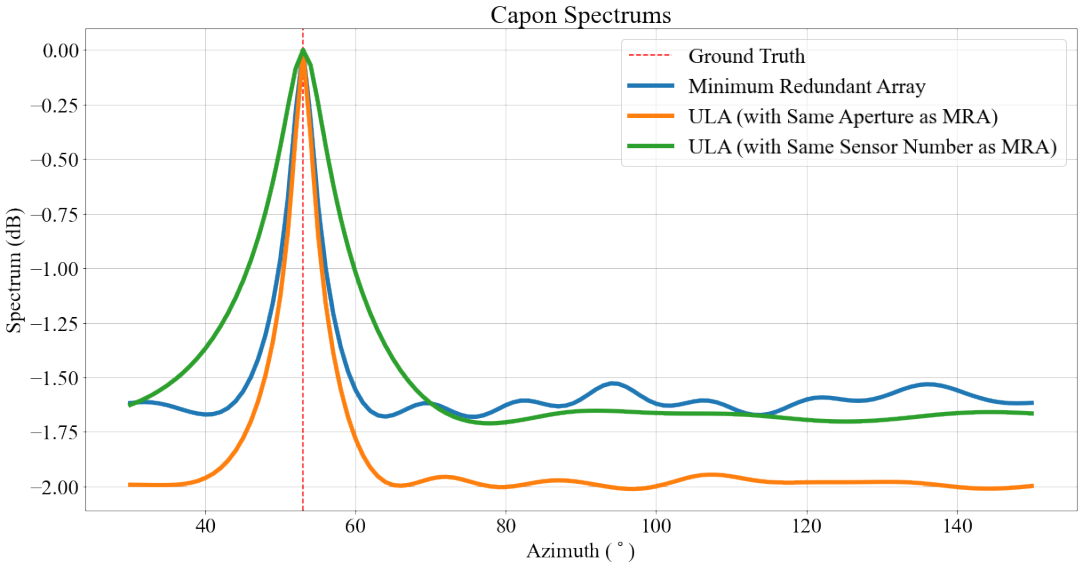

Figure 2.7: An example Capon spectrum for single source obtained by minimum redundant array and ULAs with the same aperture and number of sensors as minimum redundant array

For comparing the performance in underdetermined case, RMSE comparisons are done with 7 sources between minimum redundant array and ULA with the same

number of sensors. RMSE values for different SNR levels are given in Figure 2.8. It can be seen that minimum redundant array doesn't encounter as much performance degradation as ULA with the same number of sensors. This is due to additional degree of freedom provided by sparse array compared to ULA. An instance of Capon spectrum is illustrated in Figure 2.9. Spectrums indicate that minimum redundant array is able to generate peaks around ground truth directions while ULA cannot since it is restricted with its number of sensors.

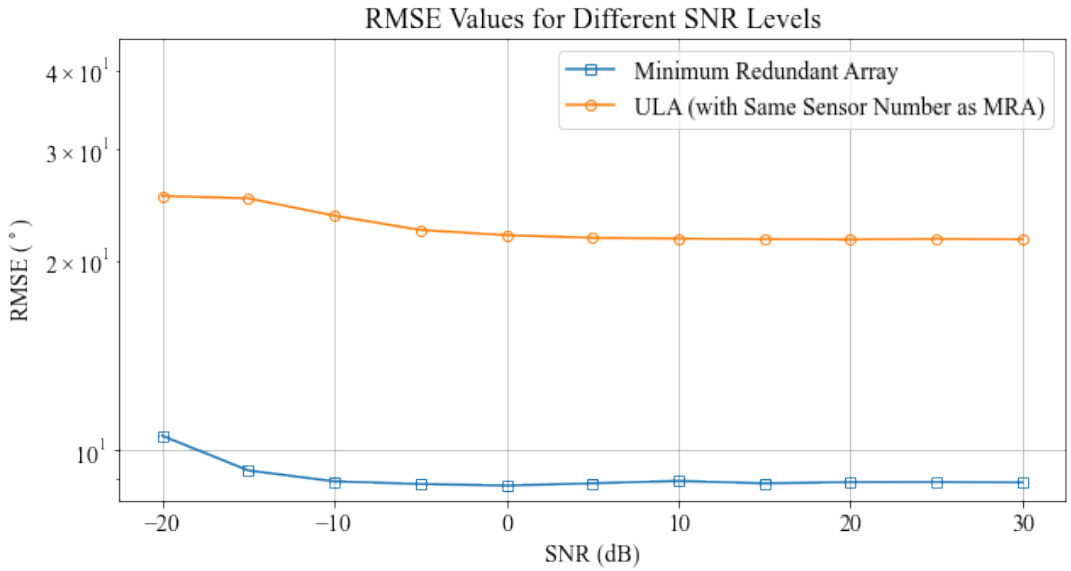

Figure 2.8: RMSE obtained by minimum redundant array and ULA with the same number of sensors for different SNR levels in underdetermined case

Resolution ability of the minimum redundant array and ULAs are compared as in Figure 2.10 for different separation angles. For closely spaced sources, minimum redundant array and ULA with the same aperture obtain similar RMSE values although ULA has more sensors. Also, it can be observed that minimum redundant array has slightly better resolution probability compared to ULA for different separation angles. This demonstrates the resolution improvement provided by sparse arrays.

2.2.2 Nested Array

Nested array is designed by combining ULAs which have increasing distances between the sensors. It can remarkably increase the degree of freedom compared to

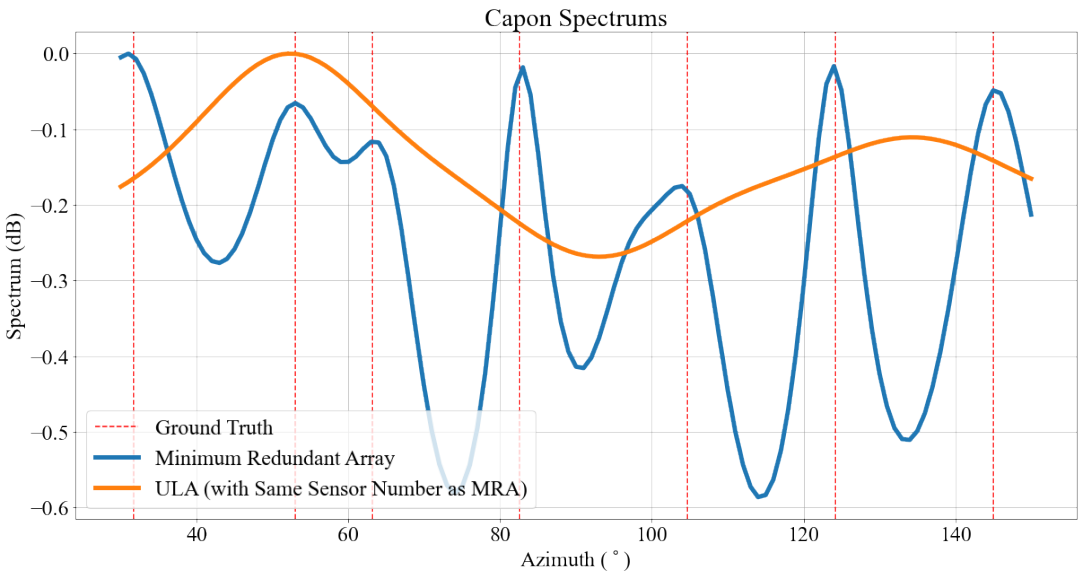

Figure 2.9: An example Capon spectrum for underdetermined case obtained by minimum redundant array and ULA with the same number of sensors as minimum redundant array

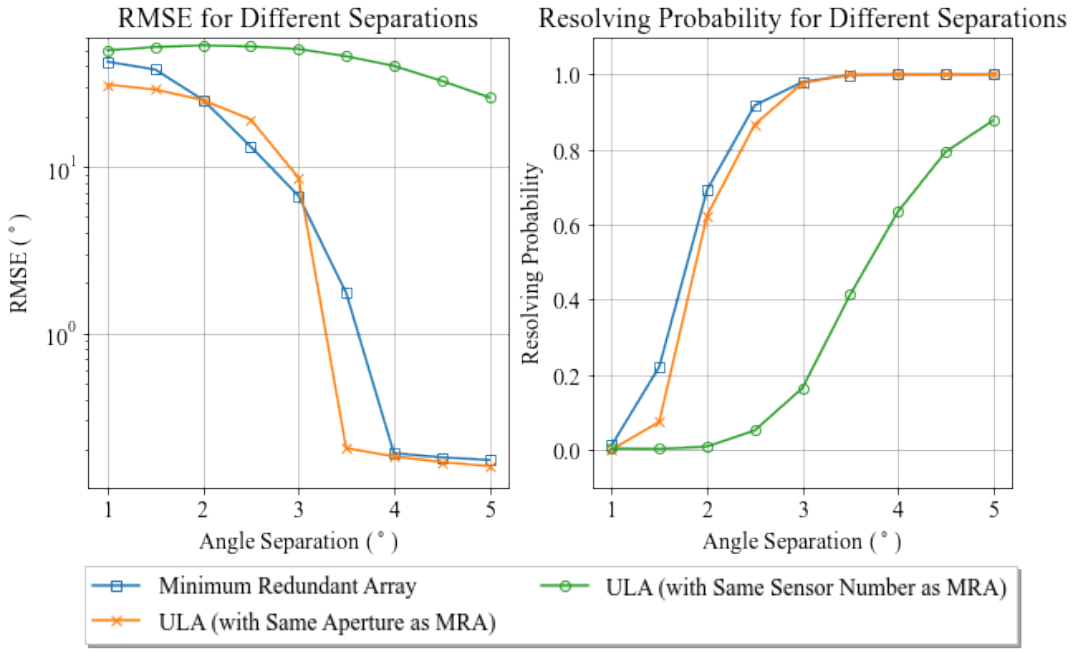

Figure 2.10: Resolving ability of minimum redundant array and ULAs with the same aperture and number of sensors for different angle separations

ULA and obtain $\mathcal{O}(M^2)$ degree of freedom [19]. Unlike minimum redundant array, there is closed form expression for its design for a given number of sensors M. A drawback of nested array is its sensitivity to mutual coupling due to the closeness of some sensors [15]. There are variants such as super nested array [23] and augmented nested array [24] proposed for decreasing mutual coupling effect.

An example nested array and corresponding difference coarray are given in Figure 2.11.

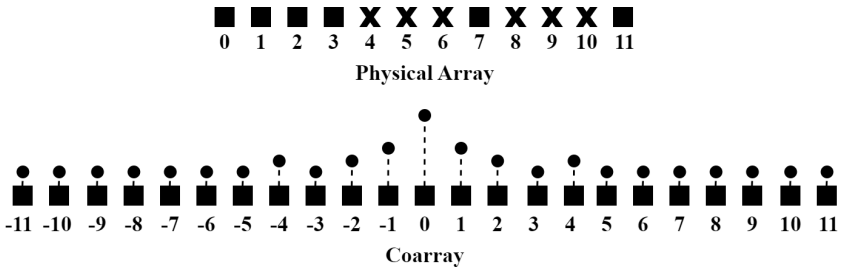

Figure 2.11: Physical array and difference coarray of a nested array. (Rectangle represents presence of the element while cross indicates its absence. The number of dashes in the dashed lines represent the weights for each lag)

The comparisons made for minimum redundant array are conducted for nested array as well. Nested array is compared with ULAs having the same aperture and the same number of sensors for which the array configurations are given in Figure 2.12.

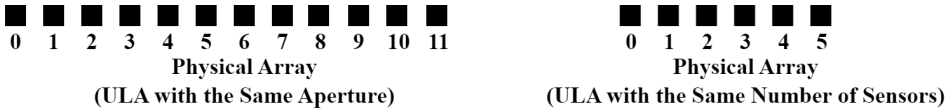

Figure 2.12: ULAs with the same aperture and number of sensors as nested array. (Rectangle represents presence of the element while cross indicates its absence)

They are compared in terms of RMSE obtained for different SNR levels as depicted in Figure 2.13. As in the case of minimum redundant array, the best performance is achieved by ULA with the same aperture as nested array. Nested array outperforms ULA with same number of sensors due to its larger aperture. Figure 2.14 illustrates example Capon spectrums generated by these arrays. Similar to the case in minimum redundant array, ULA with the same aperture as NE has higher peak due to more

sensors that it possesses. Nested array has narrower peak compared to ULA with the same number of sensors owing to its larger aperture with the expense of higher sidelobes.

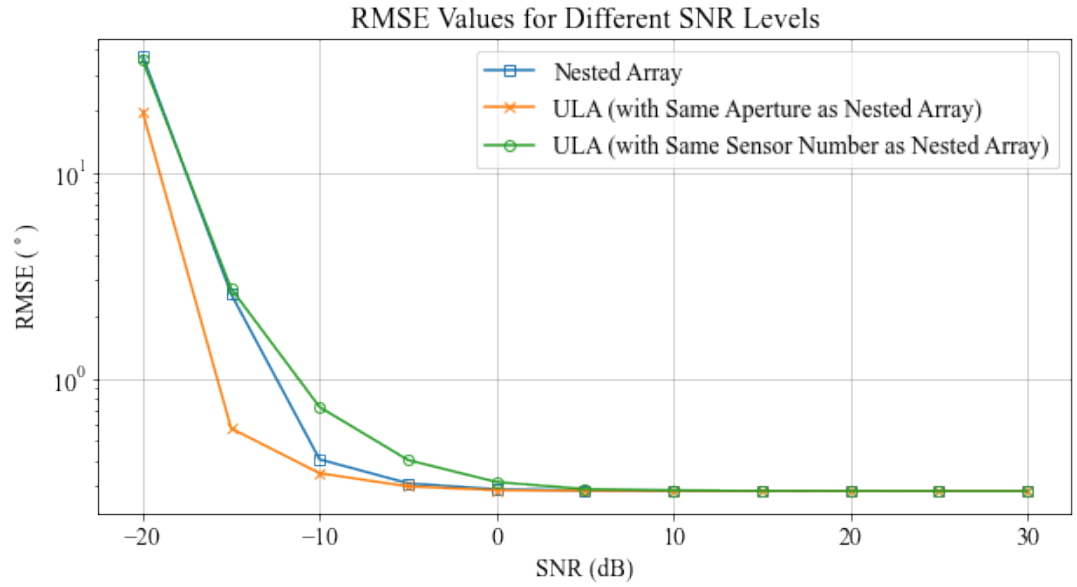

Figure 2.13: RMSE obtained by nested array and ULAs with the same aperture and number of sensors for different SNR levels in single source case

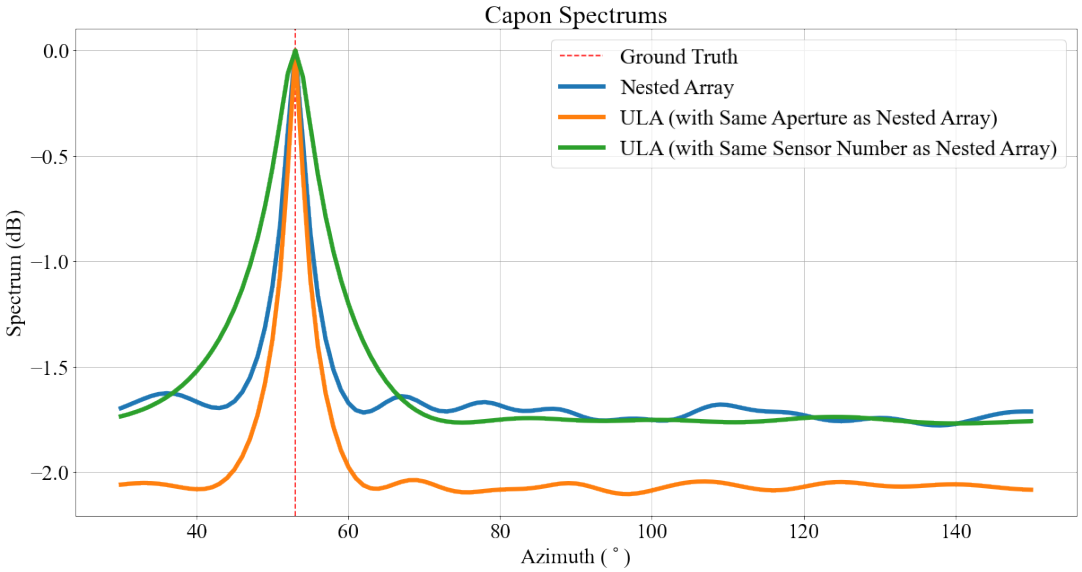

Figure 2.14: An example Capon spectrum for single source obtained by nested array and ULAs with the same aperture and number of sensors as nested array

For underdetermined case, nested array doesn't experience performance decrease as

much as ULA with the same number of sensors as shown in Figure 2.15. This is attributed to additional degree of freedom provided by sparse array compared to ULA. An example of Capon spectrum is illustrated in Figure 2.16. Similar to minimum redundant array, nested array has more apparent peaks around ground truth source directions where ULA with the same number of sensors doesn't achieve this since the number of sources is larger than its degree of freedom.

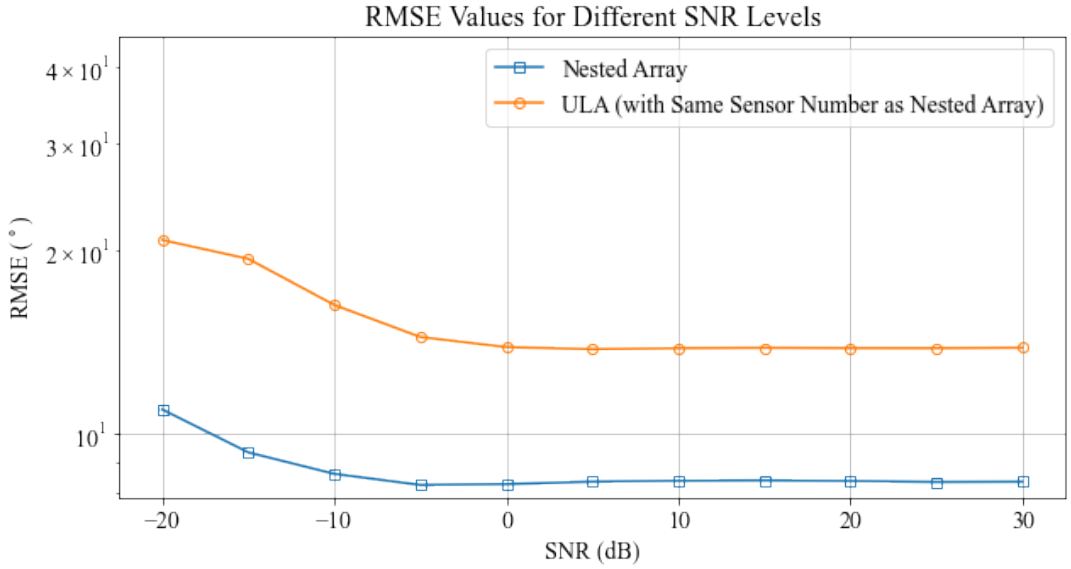

Figure 2.15: RMSE obtained by nested array and ULA with the same number of sensors for different SNR levels in underdetermined case

In terms of closely-spaced sources, ULA with the same aperture provides generally lower error while nested array has slightly better resolution probability as depicted in Figure 2.17 for different separation angles. This demonstrates the resolution ability of sparse arrays with less sensors.

2.2.3 Coprime Array

Coprime array is designed by combining two ULAs where the first one has M_1 elements with a spacing of $M_2\lambda/2$ and the second one has M_2 elements with a spacing of $M_1\lambda/2$. M_1 and M_2 are coprime integers and the first elements of the both arrays are aligned. Its degree of freedom is $\mathcal{O}(M_1M_2)$ which is higher than ULA and lower than nested array. Nested array can generate more unique self differences in its sub-

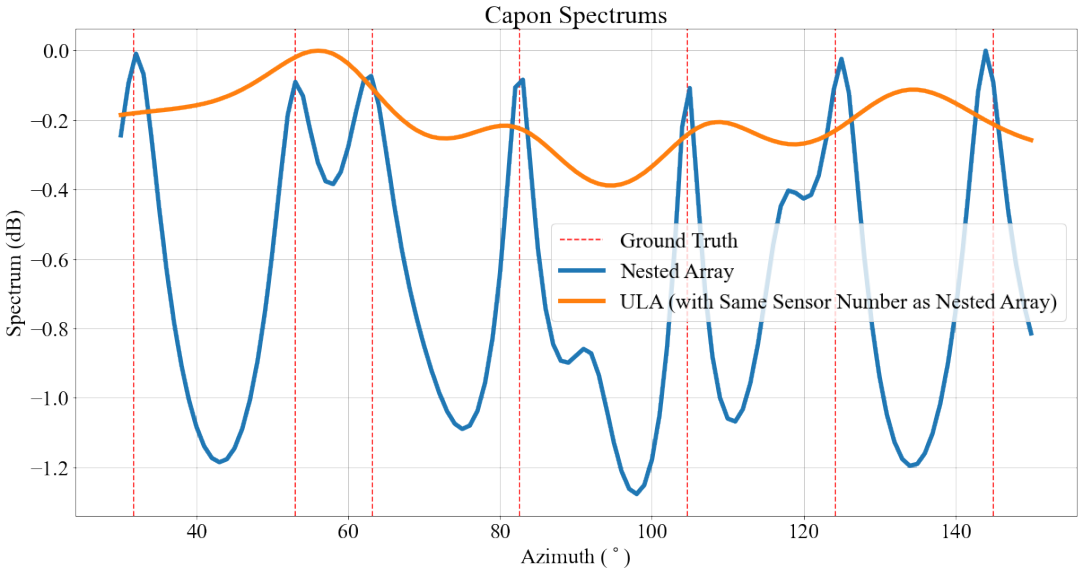

Figure 2.16: An example Capon spectrum for underdetermined case obtained by nested array and ULA with the same number of sensors as nested array

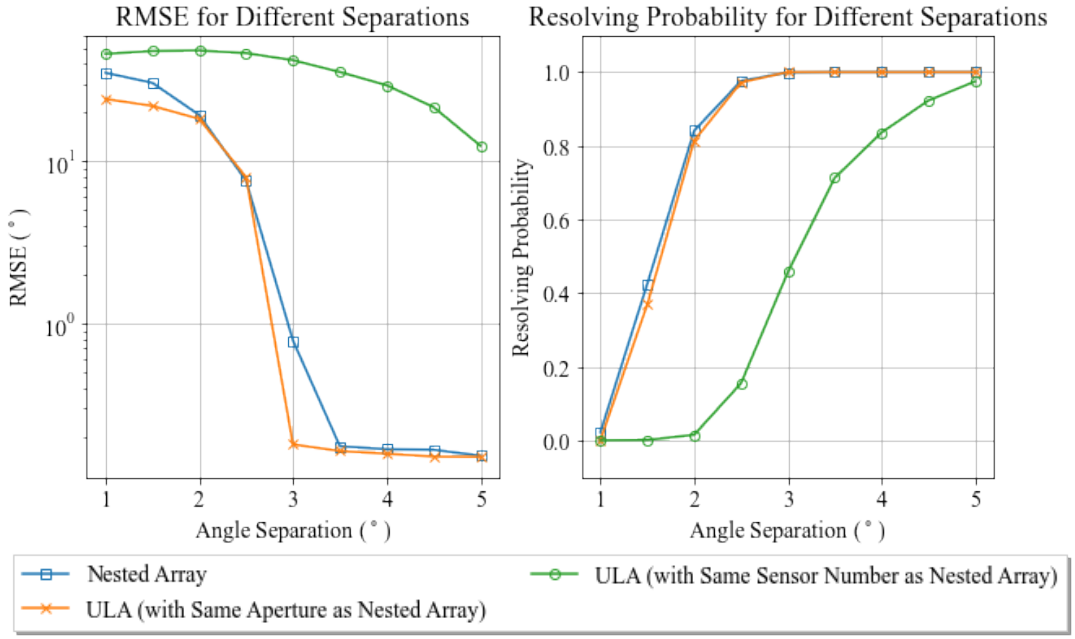

Figure 2.17: Resolving ability of nested array and ULAs with the same aperture and number of sensors for different angle separations

arrays which are not contained in cross differences and this leads to more degree of freedom. However, coprime array is more robust than nested array against mutual coupling since its inter-element spacing is larger [20].

An example coprime array and corresponding difference coarray are illustrated in Figure 2.18.

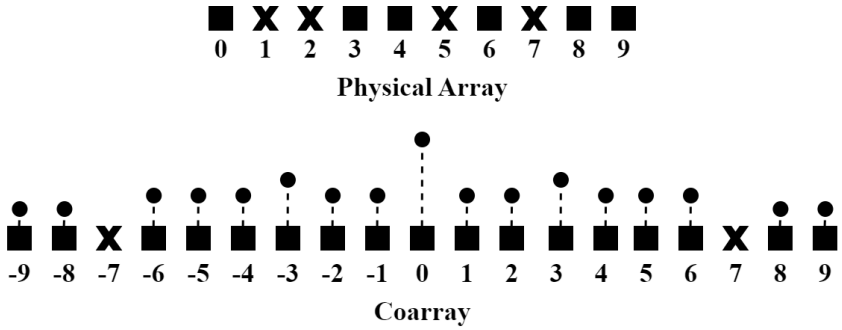

Figure 2.18: Physical array and difference coarray of a coprime array. (Rectangle represents presence of the element while cross indicates its absence. The number of dashes in the dashed lines represent the weights for each lag)

Coprime array is compared with ULAs having the same aperture and the same number of sensors which are shown in Figure 2.19.

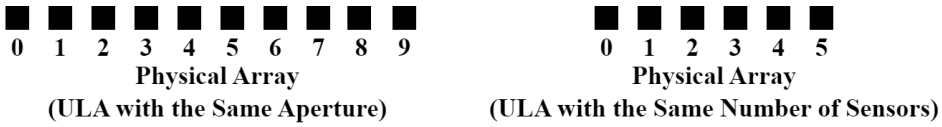

Figure 2.19: ULAs with the same aperture and number of sensors as coprime array. (Rectangle represents presence of the element while cross indicates its absence)

These arrays are compared in terms of RMSE as depicted in Figure 2.20. As in the cases of minimum redundant and nested arrays, the lowest error is obtained by ULA with the same aperture. Coprime array generally outperforms ULA with same number of sensors due to its larger aperture. Capon spectrums generated by these methods are illustrated in Figure 2.21. Similar to other sparse array types, ULA with the same aperture as CA has the highest peak. Coprime array has narrower peak compared to ULA with same number of sensors thanks to its larger aperture. However, its sidelobes are higher.

For underdetermined case, ULA encounters more performance degradation than coprime array which is also observed for other sparse array types. RMSE values for undetermined case is given in Figure 2.22. An example of Capon spectrum is shown

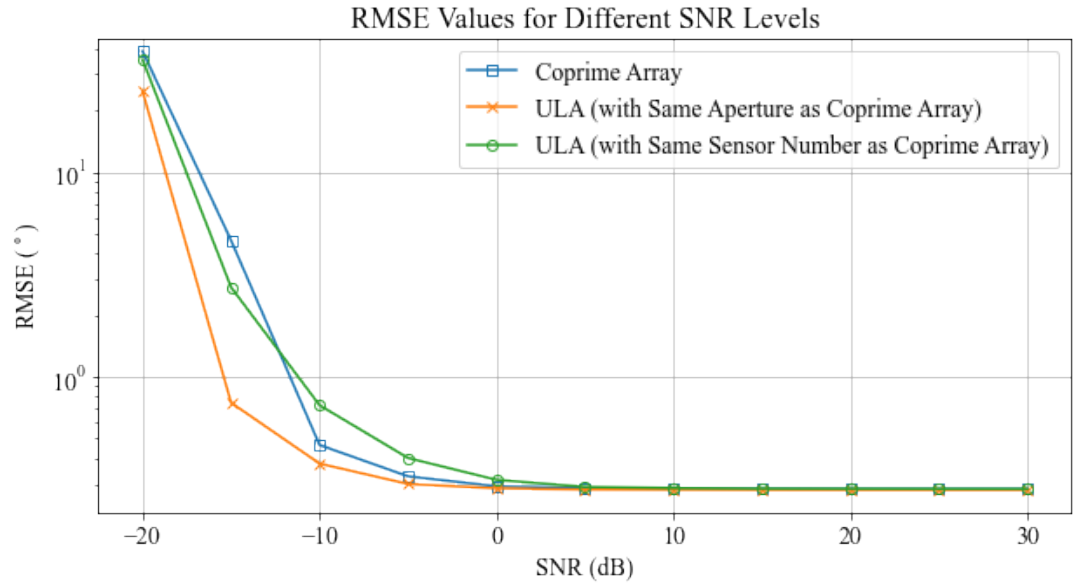

Figure 2.20: RMSE obtained by coprime array and ULAs with the same aperture and number of sensors for different SNR levels in single source case

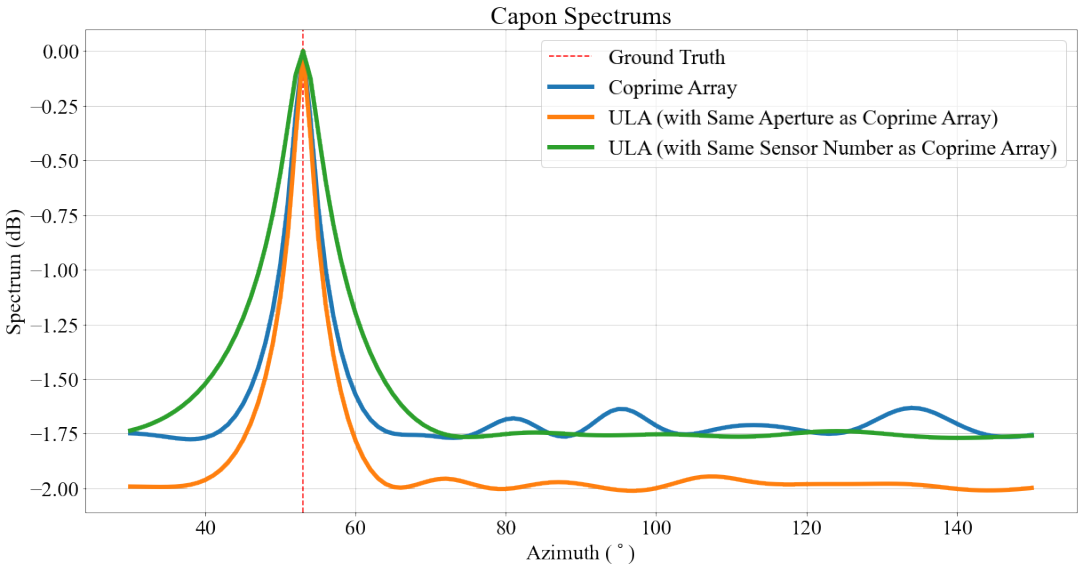

Figure 2.21: An example Capon spectrum for single source obtained by coprime array and ULAs with the same aperture and number of sensors as coprime array

in Figure 2.23. As in the case of other sparse array, coprime array can generate observable peaks at the source directions while ULA with same number of sensors cannot since ULA has lower degree of freedom than the number of sources.

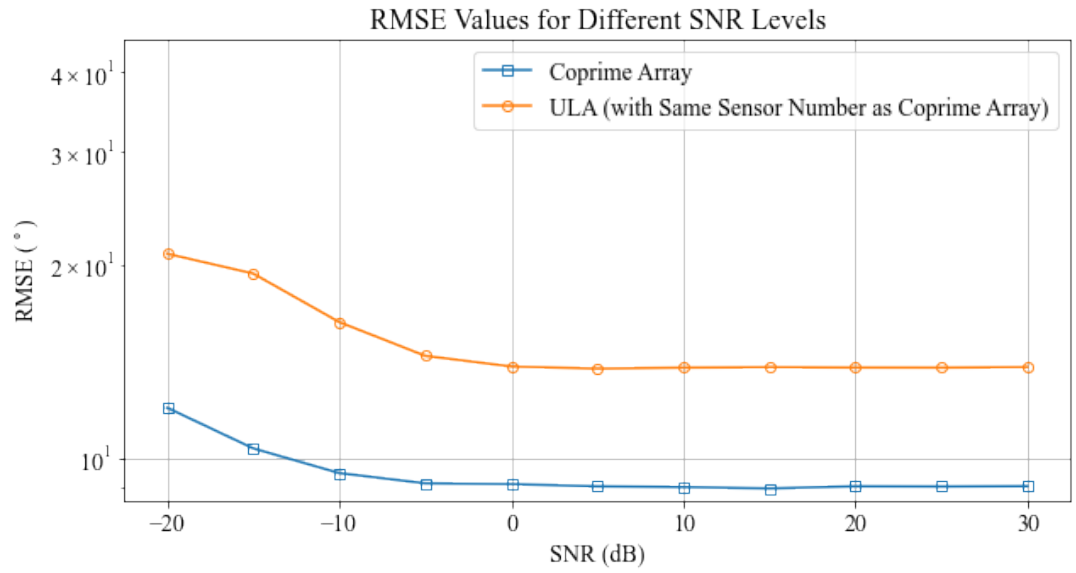

Figure 2.22: RMSE obtained by coprime array and ULA with the same number of sensors for different SNR levels in underdetermined case

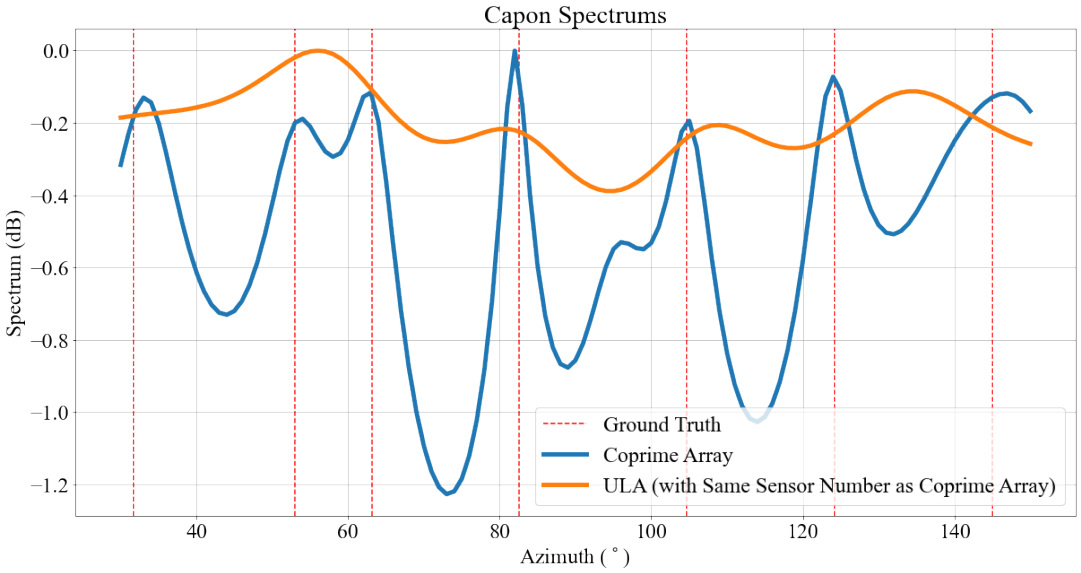

Figure 2.23: An example Capon spectrum for underdetermined case obtained by coprime array and ULA with the same number of sensors as coprime array

As in the case of other sparse arrays, coprime array provides slightly better resolution than ULA with the same aperture despite having less sensors. RMSE and resolving probability for different separation angles are depicted in Figure 2.24.

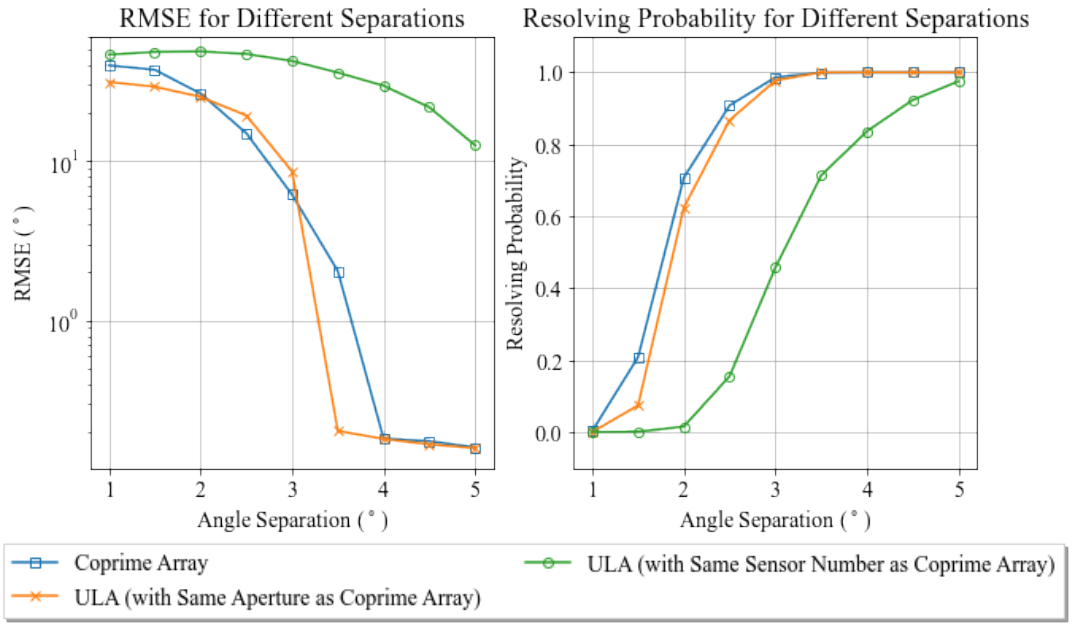

Figure 2.24: Resolving ability of coprime array and ULAs with the same aperture and number of sensors for different angle separations

2.3 Direction of Arrival Estimation Methods

Direction of arrival estimation is an important task in sensor array signal processing. There are numerous methods proposed for this task. These methods can be grouped under the following headings:

- Model-Based Methods
- Data-Driven Methods
- Hybrid Methods

Figure 2.25 illustrates the categorization of the methods and some examples for each of category.

2.3.1 Model-Based Methods

Model-based methods exploit the mathematical models relating the source directions into received signals for generating angle estimations [25]. These methods are cate-

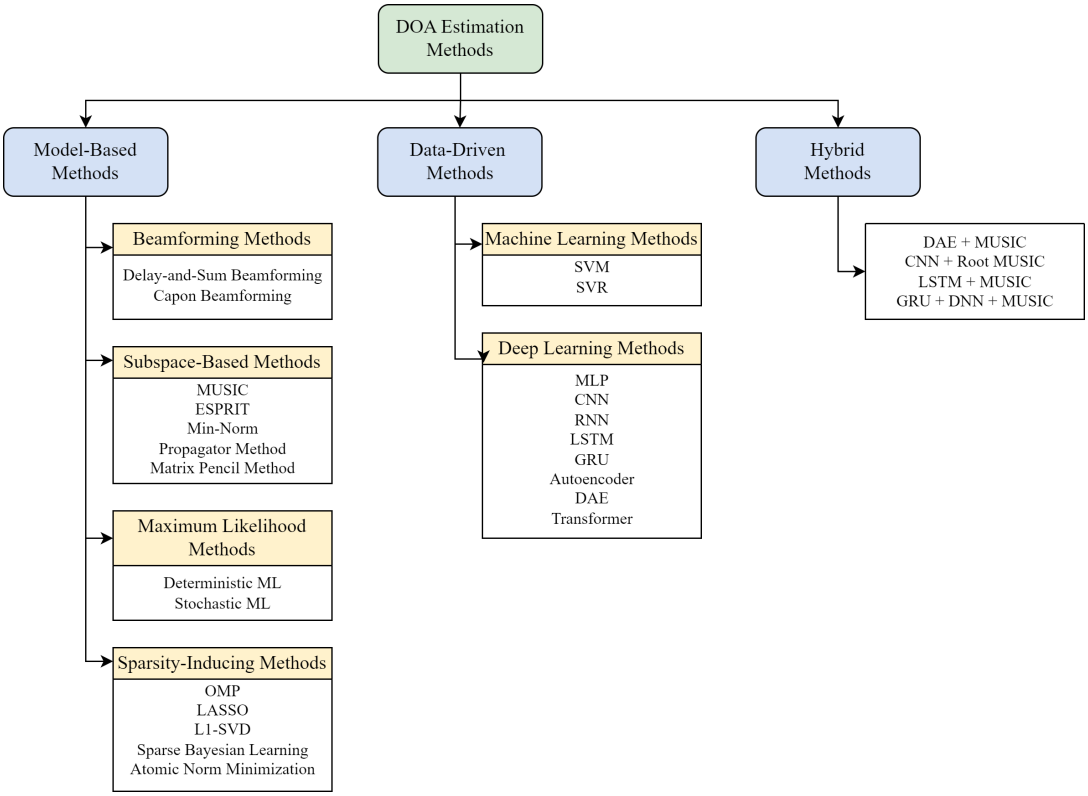

Figure 2.25: Categorization of DOA estimation methods and some examples for each category.

gorized further as below:

- Beamforming Methods
- Subspace-Based Methods
- Maximum Likelihood Methods
- Sparsity-Inducing Methods

2.3.1.1 Beamforming Methods

Beamformers are spatial filters which linearly combines the sensor signals with weights. They change the spatial response such that signals arriving from specific directions are not disturbed while others are suppressed. The weights in beamforming are usually a function of DOA angles. The output of beamforming gives power spectrum

which gets maximized at the direction of the sources. Beamforming methods utilize this relationship to provide DOA angle estimates [26].

Some examples of beamforming methods are delay-and-sum beamforming [2] and Capon beamforming [27]. More information about these methods is given in the following headings.

Delay-and-Sum Beamforming

Delay-and-sum beamforming is one of the earliest DOA estimation methods [2]. As the name implies, it applies delays and then combines the received signals as

$$y_b(t) = \frac{1}{M}\sum_{m=1}^{M} e^{j\omega\tau_m} y_m(t) = \frac{1}{M}\boldsymbol{a}^H(\theta)\boldsymbol{y}(t) \tag{2.14}$$

where $\boldsymbol{a}(\theta)$ is the steering vector corresponding to direction θ and M is the number of sensors. This operation corresponds to steering the beam into the direction θ in the array. For each direction θ, output power is calculated by

$$P(\theta) = \frac{1}{T}\sum_{t=1}^{T} |y_b(t)|^2 \tag{2.15}$$

where T is the number of snapshots. Then, DOA angle estimate is found by

$$\hat{\theta} = \arg\max_{\theta} P(\theta) \tag{2.16}$$

A major drawback of delay-and-sum beamformer is that it cannot resolve sources closer than Rayleigh limit which degrades its resolution performance [6].

Capon Beamforming

Capon beamformer minimizes the signal power in all directions except for the desired one [27]. It has sharp peaks at the source direction in its beampattern and the resolution it provides is beyond Rayleigh limit which was one of the drawbacks of delay-and-sum beamformer [26].

It is formulated using the optimization problem of

$$\min \boldsymbol{w}^H \boldsymbol{R_y} \boldsymbol{w} \quad \text{subject to } \boldsymbol{w}^H \boldsymbol{a}(\theta) = 1 \tag{2.17}$$

for which the optimal solution is

$$\boldsymbol{w_{Capon}} = \frac{\boldsymbol{R_y}^{-1} \boldsymbol{a}(\theta)}{\boldsymbol{a}^H(\theta) \boldsymbol{R_y}^{-1} \boldsymbol{a}(\theta)} \tag{2.18}$$

Power spectrum of a beamformer is formulated as [26]

$$P(\theta) = \boldsymbol{w}^H \boldsymbol{R_y}^{-1} \boldsymbol{w} \tag{2.19}$$

Substituting Equation (2.18) into (2.19), power spectrum of Capon beamformer is obtained as

$$P(\theta) = \frac{1}{\boldsymbol{a}^H(\theta) \boldsymbol{R_y}^{-1} \boldsymbol{a}(\theta)} \tag{2.20}$$

Then, DOA angle estimate is found by

$$\hat{\theta} = \arg\max_{\theta} P(\theta) \tag{2.21}$$

A drawback of Capon beamformer is that it encounters performance degradation in the presence of coherent or strongly correlated interferences. Additionally, it is not robust against errors in the array response [26].

2.3.1.2 Subspace-Based Methods

Subspace-based methods utilize the orthogonality of signal and noise subspaces to estimate source angles [28]. By leveraging this orthogonality, these methods are able to resolve sources closer than traditional Rayleigh limit, thus achieve super-resolution compared to beamforming methods [6]. Multiple Signal Classification (MUSIC) [22], Estimation of Signal Parameters via Rotational Invariance Techniques (ESPRIT) [29], Minimum-Norm Method (Min-Norm) [30], Propagator Method [31]

and Matrix Pencil Method [32] are examples of these methods. MUSIC and ESPRIT are the most promising subspace-based methods and detailed information about them is provided in the following headings [6].

MUSIC

MUSIC operates on sample covariance matrix

$$\hat{\boldsymbol{R}}_y = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{y}(t)\boldsymbol{y}^H(t) \tag{2.22}$$

where $\boldsymbol{y}$ is the received signals at the sensors and T is the number of snapshots. Eigenvalue decomposition is applied on $\hat{\boldsymbol{R}}_y$ for separating $\hat{\boldsymbol{R}}$ into addition of

$$\hat{\boldsymbol{R}}_y = \boldsymbol{U}_s\boldsymbol{\Lambda}_s\boldsymbol{U}_s^H + \boldsymbol{U}_e\boldsymbol{\Lambda}_e\boldsymbol{U}_e^H \tag{2.23}$$

where $\boldsymbol{U}_s$ and $\boldsymbol{U}_e$ hold signal and noise eigenvectors as columns and $\boldsymbol{\Lambda}_s$ and $\boldsymbol{\Lambda}_e$ contain signal and noise eigenvalues at their diagonals respectively.

Then, MUSIC spectrum is generated by

$$P_{\text{MUSIC}}(\theta) = \frac{1}{\boldsymbol{a}^H(\theta)\boldsymbol{U}_e\boldsymbol{U}_e^H\boldsymbol{a}(\theta)} \tag{2.24}$$

where $\boldsymbol{a}(\theta)$ is the steering vector corresponding to direction θ. By utilizing the orthogonality of signal and noise subspaces, peaks of the MUSIC spectrum are declared as DOA angle estimates [22].

There are variants of MUSIC algorithm and one of which is Root-MUSIC [33]. Rather than calculating the spectrum $P_{\text{MUSIC}}(\theta)$, it finds the roots of

$$z^{M-1}P_{\text{root-MUSIC}}(z) \tag{2.25}$$

where

$$P_{\text{root-MUSIC}}(z) = \boldsymbol{a}^H(z)\boldsymbol{U}_e\boldsymbol{U}_e^H\boldsymbol{a}(z) \tag{2.26}$$

and M is the number of sensors and $z = e^{j2\pi d \sin\theta}$. Once a root $\hat{z}$ is found, DOA angle estimate can be obtained by

$$\hat{\theta} = \sin^{-1}(\text{angle}(\hat{z})/(2\pi d)) \tag{2.27}$$

where d is the inter-element spacing in the array.

ESPRIT

ESPRIT benefits from the displacement invariance property of subarrays, which implies that sensors pairs at subarrays match with the same displacement vector. It doesn't require spectral search like MUSIC and this property leads to smaller computational complexity compared to MUSIC. Furthermore, it doesn't use array manifold, which introduces storage cost for MUSIC. However, rotational invariance of subarrays limits the array geometries on which ESPRIT can be applied.

It separates the array having M elements into two overlapping subarrays each with $M-1$ elements where the received signals of these subarrays can be written as

$$\boldsymbol{y_1}(t) = \boldsymbol{A_1}(\boldsymbol{\theta})\boldsymbol{s}(t) + \boldsymbol{e_1}(t) \tag{2.28}$$

$$\boldsymbol{y_2}(t) = \boldsymbol{A_2}(\boldsymbol{\theta})\boldsymbol{s}(t) + \boldsymbol{e_2}(t) \tag{2.29}$$

where $\boldsymbol{A_1}(\boldsymbol{\theta})$ and $\boldsymbol{A_2}(\boldsymbol{\theta})$ are the steering matrices of two subarrays.

Using the received signals of the whole array, sample covariance matrix is calculated by

$$\hat{\boldsymbol{R}}_y = \frac{1}{T}\sum_{t=1}^{T}\boldsymbol{y}(t)\boldsymbol{y}^H(t) \tag{2.30}$$

where $\boldsymbol{y}$ is the received signals at the sensors and T is the number of snapshots. Eigenvalue decomposition is applied on $\hat{\boldsymbol{R}}_y$ for partitioning into components belonging to

signal and noise subspaces which results in

$$\hat{\boldsymbol{R}}_y = \boldsymbol{U}_s \boldsymbol{\Lambda}_s \boldsymbol{U}_s^H + \boldsymbol{U}_e \boldsymbol{\Lambda}_e \boldsymbol{U}_e^H \tag{2.31}$$

where $\boldsymbol{U}_s$ and $\boldsymbol{U}_e$ hold signal and noise eigenvectors as columns and $\boldsymbol{\Lambda}_s$ and $\boldsymbol{\Lambda}_e$ contain signal and noise eigenvalues at their diagonals respectively.

Signal subspace gets partitioned into two parts

$$\boldsymbol{U_1} = \boldsymbol{U_s}(1 : M-1, :) \tag{2.32}$$

$$\boldsymbol{U_2} = \boldsymbol{U_s}(2 : M, :) \tag{2.33}$$

Using singular value decomposition (SVD) or least squares, rotational invariance matrix $\boldsymbol{\Psi}$ in

$$\boldsymbol{U_1 \Psi} = \boldsymbol{U_2} \tag{2.34}$$

is estimated. Eigenvalue decomposition is performed on $\boldsymbol{\Psi}$ to obtain

$$\boldsymbol{\Psi} = \boldsymbol{V \Lambda V}^H \tag{2.35}$$

Then, DOA angle estimates are found from the eigenvalues λ in $\boldsymbol{\Lambda}$ by

$$\theta_k = \arcsin\left(\frac{\arg(\lambda_k)}{2\pi d}\right) \tag{2.36}$$

where d is the inter-element spacing.

2.3.1.3 Maximum Likelihood Methods

Beamforming and subspace-based DOA estimation methods experience performance degradation in case of correlated source signals. Maximum likelihood methods do not encounter any difficulty for such cases with the price of increased computational

load due to multidimensional search over parameter space [2]. They maximize the likelihood function over DOA angle parameter to obtain the estimates [26].

There are mainly two types of maximum likelihood methods; stochastic and deterministic maximum likelihood. Information about these methods are provided in the following headings.

Deterministic Maximum Likelihood

In deterministic maximum likelihood, noise signals are generated by Gaussian random process while source signal is deterministic and known. It determines the parameters which maximizes the likelihood of given measurements.

Maximum likelihood estimate of the source direction θ can be determined by

$$\hat{\theta}_{\text{DML}} = \arg\min_{\theta} -L(\theta) \tag{2.37}$$

where

$$L(\theta) = -\log \text{Tr}\Big\{ \boldsymbol{P}^{\perp}(\theta)\hat{\boldsymbol{R}}_{\boldsymbol{y}} \Big\} \tag{2.38}$$

and $\hat{\boldsymbol{R}}_{\boldsymbol{y}}$ is the sample covariance matrix and $\boldsymbol{P}(\theta) = \boldsymbol{a}(\theta)\boldsymbol{a}(\theta)^H/M$ is the projection matrix.

Stochastic Maximum Likelihood

Stochastic maximum likelihood assumes source signal to be Gaussian random process along with noise signal.

Maximum likelihood estimate of the source directions θ is found by

$$\hat{\theta}_{\text{SML}} = \arg\min_{\theta} -L(\theta) \tag{2.39}$$

where log-likelihood is given by

$$L(\theta) = -\log\det\left[\boldsymbol{P}(\theta)\hat{\boldsymbol{R}}_{\boldsymbol{y}}\boldsymbol{P}(\theta) + \frac{1}{M-N}\,\mathrm{Tr}\Big\{\boldsymbol{P}^{\perp}(\theta)\hat{\boldsymbol{R}}_{\boldsymbol{y}}\Big\}\boldsymbol{P}^{\perp}(\theta)\right] \tag{2.40}$$

and M is the number of sensors and N is the number of sources.

2.3.1.4 Sparsity-Inducing Methods

The performance of subspace-based and beamforming methods is affected adversely when the covariance matrix of the received signals is inaccurately estimated. Small number of snapshots is one of the causes which leads to inaccurate covariance estimations. Sparsity-inducing methods are more robust against small number of snapshots since they leverage compressed sensing [34] principles [35]. Compressed sensing formulates DOA estimation problem as a sparse matrix reconstruction problem [6]. Sparse characteristics of the source signals are the main attribute which enables the usage of these methods [28, 36].

A drawback of these methods is that they require sensitive parameter tuning for different SNR and snapshot levels which has important impact on DOA estimation performance [28]. Moreover, they have long and iterative process which is impractical in real-time applications [37].

Sparsity-inducing methods can be grouped into three categories: on-grid, off-grid and gridless methods [38]. Orthogonal Matching Pursuit (OMP) [39], Least Absolute Shrinkage and Selection Operator (LASSO) [40] and L1-SVD [37] are examples of on-grid methods. Sparse Bayesian Learning [41] has extensions which can be considered as off-grid method. Atomic Norm Minimization [42] is the example for gridless methods. The following headings give information about L1-SVD and OMP.

L1-SVD

Assuming N signals are impinging into the array, then the received signals are

$$\boldsymbol{y}(t) = \boldsymbol{A}(\boldsymbol{\theta})\boldsymbol{s}(t) + \boldsymbol{e}(t) \tag{2.41}$$

An overcomplete representation $\boldsymbol{\theta_o} = \theta_{o,1}, ..., \theta_{o,G}$ which contains all the possible

source directions is introduced. The number of possible source directions G is usually bigger than the actual number of the sources N. This indicates the sparsity nature of DOA estimation problem.

Array steering matrix corresponding to overcomplete representation is generated as

$$\boldsymbol{A}(\boldsymbol{\theta_o}) = [\boldsymbol{a}(\theta_{o,1}), ..., \boldsymbol{a}(\theta_{o,G})] \tag{2.42}$$

and Equation (2.41) can be reformulated as

$$\boldsymbol{y}(t) = \boldsymbol{A}(\boldsymbol{\theta_o})\boldsymbol{s_o}(t) + \boldsymbol{e}(t) \tag{2.43}$$

where $\boldsymbol{s_o}(t)$ is the extended version of $\boldsymbol{s}(t)$ providing that $s_{o,i}(t) = s_j(t)$ if there exist a source at direction $\theta_{o,i}$ (equally θ_j), and $s_{o,i}(t) = 0$ otherwise.

Then, DOA estimation problem is reduced to sparse reconstruction problem in which the optimization problem

$$\min_{\boldsymbol{s_o}(t)} \|\boldsymbol{s_o}(t)\|_1 \quad \text{s.t.} \quad \boldsymbol{y}(t) = \boldsymbol{A}(\boldsymbol{\theta_o})\boldsymbol{s_o}(t) \tag{2.44}$$

is obtained. For multiple snapshot case, Equation (2.43) can be rewritten as

$$\boldsymbol{Y} = \boldsymbol{A}(\boldsymbol{\theta_o})\boldsymbol{S_o} + \boldsymbol{E} \tag{2.45}$$

where T is the number of snapshots, $\boldsymbol{Y} = [\boldsymbol{y}(0), ..., \boldsymbol{y}(T)]$, $\boldsymbol{S_o} = [\boldsymbol{s_o}(0), ..., \boldsymbol{s_o}(T)]$ and $\boldsymbol{E} = [\boldsymbol{e}(0), ..., \boldsymbol{e}(T)]$.

Singular value decomposition (SVD) is applied on $\boldsymbol{Y}$ to obtain

$$\boldsymbol{Y} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^H \tag{2.46}$$

and the first N columns of $\boldsymbol{U}$ is used to obtain

$$\boldsymbol{Y}_{SV} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{D}_N = \boldsymbol{Y}\boldsymbol{V}\boldsymbol{D}_N \tag{2.47}$$

where $\boldsymbol{D}_N = [\boldsymbol{I}_N, 0]^T$. Similarly, $\boldsymbol{S}_{SV} = \boldsymbol{S}\boldsymbol{V}\boldsymbol{D}_N$ and $\boldsymbol{E}_{SV} = \boldsymbol{E}\boldsymbol{V}\boldsymbol{D}_N$ are obtained and optimization problem

$$\min_{\boldsymbol{S}_{SV}} \|\tilde{\boldsymbol{s}}\|_1 \quad \text{s.t.} \quad \|\boldsymbol{Y}_{SV} - \boldsymbol{A}(\boldsymbol{\theta_o})\boldsymbol{S}_{SV}\|_f^2 \leq \beta^2 \tag{2.48}$$

where $\tilde{\boldsymbol{s}}$ is the spatial spectrum estimate whose entries are 2-norm of the rows of $\boldsymbol{S}_{SV}$ and β is the regularization parameter. After this convex optimization problem is solved, DOA angle estimates are found from $\tilde{\boldsymbol{s}}$ by applying thresholding [37].

OMP

For received signal $\boldsymbol{Y}$, the number of sources N and overcomplete dictionary $\boldsymbol{A}(\boldsymbol{\theta_o})$ where $\boldsymbol{\theta_o} = [\theta_{o,1}, ..., \theta_{o,G}]$, OMP algorithm iteratively computes the correlation of a residual matrix $\boldsymbol{R}$ with the columns of $\boldsymbol{A}(\boldsymbol{\theta_o})$.

$\boldsymbol{R}$ is initialized with $\boldsymbol{Y}$ and the column of $\boldsymbol{A}(\boldsymbol{\theta_o})$ which has the largest correlation with $\boldsymbol{R}$ is found by

$$\arg\max_{i=1,...,G} \left\| \boldsymbol{R}^H \boldsymbol{a}(\theta_{o,i}) \right\|_p \quad p \geq 1 \tag{2.49}$$

and the index i is included in a set π. Then, the optimization problem of

$$\arg\min_{\boldsymbol{P}} \|\boldsymbol{Y} - \boldsymbol{A}_\pi(\boldsymbol{\theta_o})\boldsymbol{P}\|_2^2 \tag{2.50}$$

is obtained where $\boldsymbol{P}$ is the sparse matrix containing information about DOA angles, $\boldsymbol{A}_\pi(\boldsymbol{\theta_o})$ is columns of $\boldsymbol{A}(\boldsymbol{\theta_o})$ with indexes in π.

The solution of Equation (2.50) is

$$\boldsymbol{P} = [\boldsymbol{A}_\pi^H(\boldsymbol{\theta_o})\boldsymbol{A}_\pi(\boldsymbol{\theta_o})]^{-1}\boldsymbol{A}_\pi^H(\boldsymbol{\theta_o})\boldsymbol{Y} \tag{2.51}$$

Once $\boldsymbol{P}$ is updated, $\boldsymbol{R}$ is also updated by

$$\boldsymbol{R} = \boldsymbol{Y} - \boldsymbol{A}_{\pi}(\boldsymbol{\theta_o})\boldsymbol{P} \tag{2.52}$$

and Equations (2.49-2.52) are repeated for N times.

After iterations are completed, angles in the overcomplete dictionary $\boldsymbol{\theta_o}$ corresponding to the indexes in π are declared as the DOA angle estimates.

2.3.2 Data-Driven Methods

Model-based approaches heavily rely on mathematical models which relate the received signals with the source directions. This aspect of these methods leads to performance degradation in case of non-ideal conditions such as multipath interference and array imperfections [25]. Nonideal sensor design and production, mutual interference between sensors, phase and gain inconsistencies, mutual coupling and sensor position errors are some problems which can be given under array imperfections [43]. Autocalibration techniques which are proposed for these imperfections cannot cover all of the types of imperfections since joint optimization is challenging [44].

Other non-ideal conditions which adversely affect the performance of model-based methods are low SNR, correlated sources and non-Gaussian noise [45–47]. Data-driven methods can learn the relationship between received signals and source angle without any assumptions and reliance on mathematical model or mentioned non-ideal conditions and they show robustness to these conditions [25, 48].

Another important advantage of data-driven methods over model-based ones is that they have faster inference time and lower computational complexity [28, 49]. For instance, MUSIC realizes SVD and a computationally heavy optimization is performed for maximum likelihood estimation methods. Data-driven methods generally use matrix multiplication/additions in inference and this introduces a computational advantage. A drawback of data-driven methods is that they require a massive amount of training data and training process is computationally heavy. Moreover, they are not as interpretable as model-based approaches [50].

Data-driven methods can be categorized into two groups: machine-learning based methods and deep learning based methods. Machine learning based methods use models such as SVM while deep learning based methods deploy MLP, CNN, RNN, autoencoder and transformer [51].

Randazzo et al. [52] propose support vector regression (SVR) method for DOA estimation. As input, normalized upper-triangular covariance matrix is given and gridless angle estimates are generated. Dehghanpour et al. [53] propose multiple kernel SVR for DOA estimation in the presence of mutual coupling. Similar to [52], upper-triangular covaraince matrix is utilized and gridless angle estimates are obtained. Another SVR-based method is implemented in [54] which uses the relative magnitude and phase of the received signals to a reference sensor as input.

Papageorgiou et al. [28] employ CNN for which the input is the real, imaginary and phase components of the sample covariance matrix. They formulate DOA estimation problem as a multi-label classification task and follow on-grid approach. They report performance improvement in low SNR levels. Chen et al. [44] utilize CNN for the aim of robustness to array imperfections. It takes the received signals by each sensor as input, applies 1-dimensional convolutional filters and generates a complex vector which can be used in both spectrum construction and angle estimation. During training, it aims to minimize the difference between a reference (true) spatial spectrum and the output spectrum using data which contains various imperfect effects on the array. Wu et al. [25] deploy CNN for spatial spectrum recovery in which the sparsity of the source signals is assumed and utilized. The received sensor signals are converted to a spectrum proxy using overcomplete steering matrix. The network is trained such that the sparse spectrum is obtained. The contribution of this study is that it can function in real-time on the contrary to traditional sparsity-inducing methods and provides performance improvement in low SNR. Elbir [55] employs multiple CNNs for generating MUSIC spectrum for different angle subregions. Real, imaginary and phase components of the sample covariance matrix are used as input and performance improvement is observed for correlated large number of sources. Akter et al. [56] use a CNN-based network in which 3 specialized subnetworks are employed for enriching the extracted features. Received sensor signals are used as input and a single on-grid angle estimate is generated. They report performance improvement for all SNR

levels and their method is computationally less demanding than the compared ones. Yuan et al. [57] follow a different strategy in which unsupervised learning is applied on a CNN. Sample covariance matrix is used as input and sparse power spectrum is generated at the output of the network. They formulate the loss function such that the norm of the power spectrum and the difference between the received signals and steered power spectrum is minimized. In a way, they utilize CNN for performing the optimization done by sparsity-inducing methods. The proposed method is advantageous since it doesn't require labeled data and it can function for underdetermined case in which the number of sources is larger than sensors. Yu et al. [58] focus on the existence of multipath signals and propose a CNN-based network which consists of three stages aiming to estimate multipath number, DOA angles of line-of-sight signals and non-line-of-sight signals. The input of the all stages is the sample covariance matrix. The results indicate performance improvement in terms of multipath number and DOA estimation. Lie et al. [59] use CNN for DOA estimation in the presence of non-uniform noise. They deploy attention mechanism between the convolutional layers for supressing the non-uniform noise and on-grid angle estimates are generated at the output of the network. The proposed method shows superiority over the compared methods in the presence of this noise.

Pavel et al. [4] proposes a MLP model for distributed arrays. The sample covariance matrices of each subarray are combined and upper-triangular part of the combined covariance matrix is fed into MLP. On-grid approach is followed and network can output multiple source angle estimates. It can resolve more sources compared to conventional methods while having robustness to array imperfections. Chen et al. [60] deploy a MLP consisting of two stages. The first stage divides the angular space into subregions and a MLP is used to detect the presence of source in each subregion. Then, multiple parallel MLPs are employed for each subregion to estimate on-grid DOA angle(s). The input of the network is real and imaginary components of the upper-triangular covariance matrix. The proposed method is computationally less demanding than conventional methods and requires less training data than the compared data-driven solutions.

Autoencoder based methods are generally utilized for robustness to nonideal conditions such as array imperfections, low SNR and small number of snapshots. Chen et

al. [45] utilize denoising autoencoder for reconstruction of the corrupted covariance matrix and division of the angle space into subregions. Multiple MLPs are used to refine the angle estimate in each subregion. A similar approach is proposed by [61] in which a multitask autoencoder with multiple decoders is deployed for covariance matrix reconstruction and multiple MLPs for refined angle estimation with the ultimate goal of robustness to array imperfections.

Sequential models are used for learning the patterns in received sensor signals and relate them with DOA angles. Guo et al. [62] employ a vision transformer which takes real, imaginary and phase components of covariance matrix and generated on-grid angle estimates. The proposed method has lower computational complexity compared to CNN and provides better results especially in low SNR and small number of snapshots. Wang et al. [63] utilize star transformer model which has attention connections between only adjacent sequential steps. It generates on-grid angle estimates by using real, imaginary and phase components of covariance matrix as input. Lan et al. [64] proposes a network containing two subnetworks for SNR classification and DOA estimation. A MLP is used for classifying the SNR into two levels and based on SNR level, a multi-head attention network is activated for DOA estimation. Eigenvalues are used as input of SNR classification network while DOA estimation network uses upper triangular covariance matrix. The proposed network can output gridless angles for multiple sources. Usage of attention mechanism enables more expressive feature to be extracted.

2.3.3 Hybrid Methods

The lack of interpretability in data-driven methods arise the need of methods which are robust against non-ideal conditions while having interpretability [50]. Hybrid methods are proposed for such purpose and they are combinations of data-driven and model-based approaches. In this methods, data-driven methods are generally used in initial stages for mitigating the effects of non-ideal conditions while DOA estimation is performed by model-based techniques for having interpretability.

Barhelme et al. [65] propose a hybrid method for the arrays which perform subarray sampling. It is composed of a MLP and MUSIC where MLP is used to reconstruct

the full array covariance matrix using that of subarrays and MUSIC performs DOA estimation on the reconstructed matrix. It provides ability to estimate more sources than the number of subarray sensors. Xiang et al. [48] employs an LSTM based hybrid method for achieving robustness against array imperfections. LSTM performs phase enhancement on the sequential noisy covariance matrices. The reconstructed covariance matrix is used by a model-based method to obtain angle estimates. Papageorgiou et al. [46] use a denoising autoencoder (DAE) for denoising the sample covariance matrix. A sparse array configuration is selected and real and imaginary parts of covariance matrix is used as input to DAE. Its denoised version corresponding to ULA is generated. Spatial smoothing is applied on reconstructed covariance matrix for rank recovery and used by MUSIC to obtain angle estimates. Shmuel et al. [50] deploy a CNN based denoising autoencoder for reconstructing the sample covariance matrix. Reconstructed matrix is converted to Hermitian positive definite and root-MUSIC is utilized to obtain gridless angle estimates. The constructed architecture can be trained end-to-end since operations in root-MUSIC is differentiable. Wu et al. [66] propose a CNN for reconstructing the covariance matrix belonging to ULA from that of a sparse linear array. Input of the network is real and imaginary parts of sample covariance matrix of sparse array and a column of covariance matrix of ULA is generated as the output. Through Toeplitz matrix reconstruction process, noise-free covariance matrix is recovered. Another CNN is used to estimate the number of sources and different parameters are loaded for reconstruction network based on source number estimate for increasing generalization ability. Root-MUSIC is employed to obtain gridless angle estimates. Markofer et al. [67] employ GRU and MLP models that mimick the operations performed in MUSIC. GRU is used to learn the correlations between the received sensor signals. The resultant vector from GRU is converted into a covariance matrix and eigenvalue decomposition is applied to obtain eigenvalues and eigenvectors. A MLP is applied on eigenvalues to generate weights for each eigenvector for selecting those belonging to noise subspace. An additional MLP estimates the number of sources using eigenvalues. Then, MUSIC spectrum is found using the weighted eigenvectors and another MLP is used for peak detection purpose on the spectrum and obtains gridless angle estimates. Ji et al. [68] employs a transformer for noise subspace estimation where the input is the received sensor signals. The estimated noise subspace eigenvectors are used by a MLP for source

enumeration. Furthermore, MUSIC spectrum is generated using these eigenvectors and another MLP is used as peak detector to estimate gridless DOA angles(s).

2.4 Source Enumeration Methods

In practice, the number of sources that are present in the environment is mostly unknown. Therefore, source enumeration is another essential task in array signal processing.

Source enumeration methods can be categorized into the following groups:

- Information-Theory Based Methods
- Eigenvalue Based Methods
- Data-Driven Methods

Figure 2.26 presents the categorization of the methods and some examples for each category.

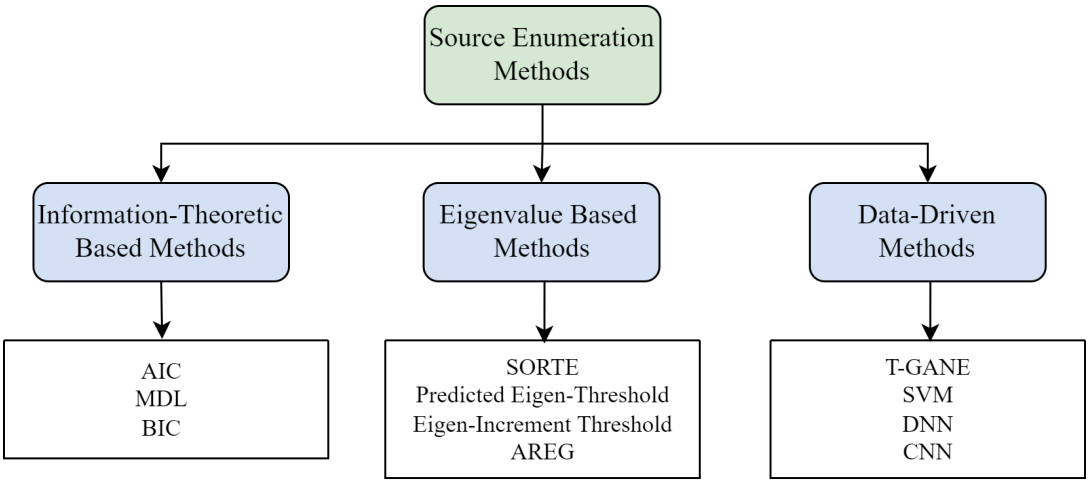

Figure 2.26: Categorization of source enumeration methods and some examples for each category.

2.4.1 Information-Theory Based Methods

Information-theory based methods estimate the number of sources by calculating a selected criterion and finding the number of sources which minimizes it [69]. A drawback of these methods is that they encounter performance degradation for small number of snapshots [70].

There are different criteria which can be employed in these methods. Some examples of these criteria are Akaike Information Criterion (AIC) [71], Minimum Description Length (MDL) [72], and Bayesian Information Criterion (BIC) [73]. AIC and MDL, which are the most famous information-theory based methods, are explained in the following headings.

AIC

AIC is determined using the eigenvalues of the covariance matrix of the received signals. For a candidate number of sources k, AIC is calculated by

$$I_{\text{AIC}}(k) = -2L(\mathbf{\Lambda}) + 2k(2M - k) \tag{2.53}$$

and

$$L(\mathbf{\Lambda}) = \log \left(\frac{\prod_{i=k+1}^{M} \lambda_i^{\frac{1}{M-k}}}{\frac{1}{M-k} \prod_{i=k+1}^{M} \lambda_i} \right)^{(M-k)T} \tag{2.54}$$

where M is the number of sensors (hence the number of eigenvalues), T is the number of snapshots and λ_i is the i^{th} eigenvalue of the covariance matrix. The number of sources k which minimizes $I_{\text{AIC}}(k)$ is selected as the source number estimate. AIC has a good performance in low SNR levels while it cannot reach the accuracy level of MDL for high SNR region [74].

MDL

As in the case of AIC, MDL is also determined using the eigenvalues of the covariance matrix of the received signals. For a candidate number of sources k, MDL is found

by

$$I_{\text{MDL}}(k) = -L(\mathbf{\Lambda}) + \frac{1}{2}k(2M-k)\log(T) \tag{2.55}$$

where $L(\mathbf{\Lambda})$ is given in Equation (2.54) and M is the number of sensors. The number of sources k which minimizes $I_{\text{MDL}}(k)$ is selected as the source number estimate. MDL is considerably successful for high SNR levels, however, its performance for noisy signals is extremely decreased [74].

2.4.2 Eigenvalue Based Methods

These methods exploit eigenvalues of the covariance matrix of received signals and estimate the number of sources by considering the magnitude, difference or ratio of the eigenvalues. Example methods which fall in this category are Second Order Statistic of the Eigenvalues (SORTE) [75], predicted eigen-threshold [76], eigen-increment threshold [77] and Accumulated Ratio of Eigenvalues Gaps (AREG) [74]. Information about these methods is provided in the following headings.

SORTE

SORTE utilizes the variances of the difference of eigenvalues and generates a source number estimate by considering the ratio of difference sets.

Firstly, differences between the consecutive eigenvalues are found by

$$\nabla\lambda_i = \lambda_i - \lambda_{i+1} \qquad i = 1, ..., M-1 \tag{2.56}$$

where M is the number of sensors, hence the number of eigenvalues. The variance of set $\{\nabla\ \lambda_i\}_{i=k}^{M-1}$ is calculated by

$$\sigma_k^2 = \frac{1}{M-k}\sum_{i=k}^{M-1}[\nabla\ \lambda_i - \frac{1}{M-k}\sum_{j=k}^{M-1}\nabla\ \lambda_j]^2 \tag{2.57}$$

Then, SORTE is defined as

$$\text{SORTE}(k) = \begin{cases} \frac{\sigma_{k+1}^2}{\sigma_k^2}, & \sigma_k^2 > 0 \\ \infty, & \sigma_k^2 = 0 \end{cases} \tag{2.58}$$

Source number estimate is determined by

$$\hat{N} = \underset{k=1,\ldots,M-3}{\arg\min}\ \text{SORTE}(k) \tag{2.59}$$

SORTE generally outperforms AIC and MDL in terms of accuracy however, it cannot detect as many sources as them since it uses the ratio of the difference set variances in the estimation [74].

Predicted Eigen-Threshold

Hypothesis testing procedure is employed in which the thresholding is applied on eigenvalues of covariance matrix. For all candidate number of sources k, average of noise-subspace eigenvalues is calculated by

$$l_k = \frac{1}{M-k+1} \sum_{j=k}^{M} \lambda_j \tag{2.60}$$

where M is the number of sensors. This average value is used to calculate a threshold for λ_{M-k} by

$$\bar{\lambda}_{M-k} = ((k+1)\frac{1 + t(T(k+1))^{-0.5}}{1 - t(Tk)^{-0.5}} - k) l_{M-k+1} \tag{2.61}$$

where T is the number of snapshots and t is a threshold which should follow $t/\sqrt{N} < 1$.

Then, hypothesis testing procedure is applied. Two binary hypotheses are formed:

- H_0: $N < M - k$
- H_1: $N = M - k$

where N is the true number of sources. H_1 is validated if $\lambda_{M-k} > \bar{\lambda}_{M-k}$. H_0 is validated if $\lambda_{M-k} < \bar{\lambda}_{M-k}$.

Starting from $k = 1$, threshold is calculated and comparison is made. If H_1 is validated, then source number estimate is declared as $\hat{N} = M - k$. Else, k is increased by one and testing continues until H_1 gets validated or $k = M$.

Eigen-Increment Threshold

In the absence of source signals, noise eigenvalues would have similar values. Based on this fact, the difference between consecutive eigenvalues is compared with a threshold to estimate the number of sources. The difference of eigenvalues is named as eigen-increment.

Threshold for eigen-increment values is calculated by

$$\bar{\nabla\lambda} = \rho(M,T)\frac{P_s}{(1+\sqrt{P_s/\lambda_M})} \tag{2.62}$$

where M is the number of sensors, T is the number of snapshots, $\rho(M,T)$ is a coefficient selected depending on M and T and P_s is signal power which is estimated by $P_s = (\lambda_1 - \lambda_M)/M$.

Starting from $i = M-1$ down to $i = 1$, eigen-increment values $\lambda_i - \lambda_{i+1}$ are calculated and compared with threshold $\bar{\nabla\lambda}$. Source number estimate is declared as $\hat{N} = i$ if $\lambda_i - \lambda_{i+1} > \bar{\nabla\lambda}$.

AREG

Eigenvalues belonging source subspace are larger than the ones belonging to noise space in magnitude. AREG utilizes this fact and compares the changes between consecutive eigenvalues to detect the number of sources.

AREG is calculated by

$$\text{AREG}(k) = \lim_{\delta\to+0} \frac{\Delta\lambda_{k+1}}{\frac{1}{k}\sum_{i=1}^{k}\Delta\lambda_i + \delta} \qquad k = 1, ..., M-2 \tag{2.63}$$

where

$$\Delta\lambda_{k+1} = \lambda_{k+1} - \lambda_k \tag{2.64}$$

and source number estimate is determined by

$$\hat{N} = M - 1 - \underset{k=1,\ldots,M-2}{\arg\max}\ \text{AREG}(k) \tag{2.65}$$

2.4.3 Data-Driven Methods

Methods utilizing the data for source enumeration are considered in this category. The utilization of the data occurs with respect to different aspects such as distribution approximation and nonlinear mapping for different methods. These methods can be a part of another data-driven method proposed for DOA estimation. These methods can achieve high performance since they do not rely on signal models but learn the nonlinear relationships from data. However, one drawback of these methods is that they require large amount of data which may not be possible to collect in all applications.

There are machine learning and deep learning based methods proposed for nonlinear mapping. Lee et al. [74] proposes Threshold for Gap of Normalized Eigenvalues (T-GANE) method which employs Gaussian mixture models for learning the distribution of eigenvalue gaps for noise-signal subspace and noise-noise subspaces. After learning distributions, an optimal threshold is selected and used in source enumeration. Yun et al. [78] uses a support vector machines (SVM) for source enumeration along with a MLP for SNR estimation. Rogers et al. [10] deploys MLP for the first time in the literature which takes the spatially smoothed covariance matrix entries and its eigenvalues as input and estimated number of sources as output. Yang et al. [11] proposed two MLP, as regression and classification networks that can be used independently, which uses eigenvalues as input and estimates the number of sources on their outputs. Papageorgiou et al. [28] proposed CNN for DOA estimation where a thresholding is applied on the output probabilities to infer the number of sources. Wu et al. [66] embeds a CNN into their hybrid method for source enumeration which

outputs the source number estimate up to $M-1$ where M is the number of sensors. Merkofer et al. [67] utilize a MLP in their augmented MUSIC approach which uses eigenvalues to estimate the number of sources up to $M-1$. Feintuch et al. [47] proposes a fully-connected network for DOA estimation which outputs on-grid angle estimates. Source enumeration is performed by applying thresholding at the output neurons. Yu et al. [58] utilize a CNN for multipath number estimation which takes sample covariance matrix as input. They realize multipath enumeration with a separate network where there are other network for multipath and source DOA estimation. Wang et al. [63] changes the last fully connected layer of their star-transformer based architecture for source enumeration such that the last layer has dimension of maximum number of sources. Ji et al. [68] employ a MLP for source enumeration which takes the eigenvectors generated by transformer as input and generates source number estimate up to $M-1$.

2.5 Sensor Malfunctions

Sensor arrays may encounter sensor malfunctions which leads to performance degradation in DOA estimation and source enumeration tasks. In the literature, the effect of the failure of a sensor on the difference coarray is used to categorize sensors. A sensor is called as essential if its removal from the array introduces hole(s) in the difference coarray. For multiple sensor malfunctions, the influence on the difference coarray gets more complex. For analyzing the multiple case, the concept of k-essentialness, which is the generalization of essentialness, is introduced. A set of sensors $\mathbb{S} = \{S_1, S_2, ..., S_k\}$ is called k-essential if the removal of this set from the array leads to hole(s) in difference coarray [13].

Sparse arrays are sensitive to sensor failures due to their difference coarray structures [13]. They have more essential elements compared to a ULA with the same number of sensors and removal of these elements may decrease the degree of freedom and lead to performance degradation [79].

Sensor failures can be separated into the following two groups in terms of awareness of the failure:

- Known Sensor Failures: The sensors which face malfunction are known.
- Unknown Sensor Failures: The sensors which face malfunction are unknown. For identifying the faulty sensors, array diagnosis methods are used.

In terms of the comprehensiveness of the failures, the following categories can be considered:

- Partial Sensor Failures: Malfunctioning occurs for a part of the all collected snapshots.
- Complete Sensor Failures: Malfunctioning occurs for the all collected snapshots.

Various studies are conducted for sensor malfunctions regarding DOA estimation. These studies can be separated into the following groups based on the type of the method that they propose. Some of these studies also contain approaches for array diagnosis.

- Matrix completion based methods
- Compressed sensing based methods
- Data-driven methods
- Maximum likelihood estimation methods

Sun et al. [79] consider unknown-complete sensor malfunctions and propose an array diagnosis method in which the absolute sum of the rows/columns of covariance matrix is compared with a threshold calculated from the average of other rows/columns. Array element for which the absolute sum doesn't exceed the threshold is declared as faulty element. If the faulty element is essential and leads to hole(s) in difference coarray, then matrix completion approach is followed and singular value thresholding (SVT) is applied to fill the covariance matrix. Otherwise, the corresponding entries in the covariance matrix is filled by the average of the entries corresponding to the same lag in the difference coarray. Sun et al. [80] develop array diagnosis and matrix

completion methods for unknown-complete sensor failures. Array diagnosis method is the same as described for [79] and matrix completion is achieved by first extending the covariance matrix of faulty array into Toeplitz form of intact array. Then, matrix completion theory is applied by an optimization problem involving trace norm and mapping matrix which select non-faulty sensors. Then, reconstructed matrix is used with root-MUSIC to obtain DOA angle estimates. Setayesh et al. [81] study different matrix completion techniques such as SVT, low-rank matrix fitting (LMaFit) and OptSpace for the case of complete-known failures. MUSIC is employed for DOA angle estimation after matrix completion. Jalal et al. [82] consider unknown-partial sensor failures and propose a diagnosis method in which the absolute value of the received signals are compared with the weighted mean of remaining sensor signals at that snapshot and declared as faulty in case it is smaller than the threshold. Received signal recovery is applied by low rank matrix completion. The rank of the recovered signal matrix is minimized such that the recovered matrix is as similar as possible to the observed matrix. Since it is non-convex, it is reformulated as frobenius norm minimization and non-convex relaxation is used. Recursive least squares with nulling antenna array is applied for DOA estimation where nulls occur at the direction of received signals. Zhu et al. [83] focuses on unknown-complete sensor malfunctions and their proposed method relies on difference coarray. Array diagnosis is conducted by comparing frequency and amplitude of coarray elements. The impaired sensor data is replaced by the intact sensors' data which belong to the same lag in the difference coarray. After replacement, the resulting matrix might be of rank 1, therefore spatial smoothing is applied. Then, MUSIC is used for DOA estimation. Yerriswamy et al. [84] consider unkown-partial sensor malfunctions and propose methods regarding array diagnosis and matrix completion. For array diagnosis, the difference between successive data from a sensor is observed and in case it is larger than a threshold, this sensor is labeled as malfunctioning. SVT is used for matrix completion. Input is the received data matrix. The problem is formulated as rank minimization. Since rank minimization is NP-hard, convex relaxation is applied by which the problem is turned into nuclear norm minimization. Matrix pencil method is applied on completed matrix to obtain DOA estimates.

Chen et al. [85] focus on known-complete failures and propose joint missing data

recovery and DOA estimation framework. They apply singular value decomposition (SVD) on received signal matrix for dimension reduction and robustness improvement against noise. Missing data recovery is incorporated into sparse signal reconstruction with a reweighted $l_{2,1}$ norm penalty. After solving this optimization problem, sparse signal matrix is obtained where the angles corresponding to rows with minimum norm are declared as source angle estimates. Hamid et al. [86] propose modified version of OMP for unknown-complete sensor failures. Sensor failures are identified by comparing variance of each sensor's received energy with median variance. A modified stopping criteria is used in OMP which incorporate the effect of faulty sensor using raised sidelobe levels.

Vigneshwaran et al. [87] work on unknown-partial sensor malfunctions for which a data-driven method is employed. A radial basis function (RBF) based minimum resource allocation network is proposed for which the number of hidden neurons change based on input data. Off-diagonal entries of covariance matrix are used as input. Ahmed et al. [88] propose two data-driven methods for known-complete sensor failures. The first one compensate the missing entries in the covariance matrix obtained from fault array using MLP. MLP is applied after spatial smoothing on the vectorized covariance matrix. The second approach emulates spatial smoothing process as well and directly applied on vectorized covariance matrix of faulty array. Therefore, the output of the network is spatially smoothed and hole-filled vectorized covariance matrix. MUSIC is applied on this matrix to obtain angle estimates. Zheng et al. [35] propose a data-driven approach for known-complete failure case where only a single snapshot is available. A sparse augmentation model is proposed which is comprised of an augmentation layer where a random mask is applied on input to simulate different sparse array configurations. Augmentation layer masks randomly selected sensors where the number is limited by a parameter. A normalization layer is used for different number of sensor cases. Then, a fully connected layer (FCL) is applied. Sparse signal frequency embedding and active antenna position encoding is applied to obtain hand-crafted features. These features are concatenated with FCL's output. MLP is applied to obtain DOA estimates. He et al. [89] employs a MLP for known-complete sensor malfunctions for which meta-learning is utilized during training. The constructed network takes off-diagonal entries of sample covariance matrix

and trained with different sensor failure scenarios which are considered as distinct tasks. The proposed method can learn the inherent relationship between these tasks and finds a global optimum parameter vector. The output of the proposed network is on-grid angle estimates.

Wang et al. [90] consider known-partial sensor failures in which different sensors may fail when collecting a bunch of snapshots. They focus on sparse arrays and propose maximum likelihood estimation with which partial measurements from different sensors can be used for obtaining Toeplitz parametrization of ULA covariance matrix. Then, the obtained matrix is used by MUSIC to generate angle estimates. Larsson et al. [91] focus on known-partial sensor malfunctions and propose maximum likelihood estimation which utilizes Cholesky factorization of covariance matrix. Estimated covariance matrix is used by MUSIC or covariance matching techniques (COMET) for generating angle estimates.

In this thesis, sensor failures are restricted to known-complete failures and the robustness of the proposed method is evaluated for such kind of failures. The reason for such a restriction is that the proposed method doesn't apply array diagnosis and it processes sample covariance matrix from which it is not possible to utilize the data collected during non-failure moments.

CHAPTER 3

TRANSFORMER-BASED DIRECTION OF ARRIVAL ESTIMATION AND SOURCE ENUMERATION

In this chapter, the proposed method for DOA estimation and source enumeration is explained with details. It falls under the category of data-driven methods and utilizes deep learning models. Section 3.1 gives background information about deep learning related concepts and Section 3.2 describes the architectures for the proposed method.

3.1 Background Information

In Section 2.1, array signal model is derived as

$$\boldsymbol{Y} = \boldsymbol{A}(\boldsymbol{\theta})\boldsymbol{S} + \boldsymbol{E} \tag{3.1}$$

where $\boldsymbol{Y}$, $\boldsymbol{S}$ and $\boldsymbol{E}$ contain received signals, source signals and noise respectively. $\boldsymbol{A}(\boldsymbol{\theta})$ is the array steering matrix corresponding to source directions $\boldsymbol{\theta}$. DOA estimation aims to find $\boldsymbol{\theta}$ and this problem can be considered as an inverse problem. Equation (3.1) can be rewritten as

$$\boldsymbol{Y} = f(\boldsymbol{\theta}) + \boldsymbol{E} \tag{3.2}$$

where f is the function which represents the physical theory governing the phase changes in the received source signals depending on the source directions.

Deep learning methods can be interpreted as the solvers of such inverse problems since they operate on some kind of observed measurements and generate estimations

about a desired quantity, which is the unknown state of the system [92]. For DOA estimation problem, it generates

$$\hat{\boldsymbol{\theta}} = g(\boldsymbol{Y}; \boldsymbol{\Psi}) \tag{3.3}$$

where $\hat{\boldsymbol{\theta}}$ is the DOA angle estimate, g represents the operations performed in the network and $\boldsymbol{\Psi}$ contains the network parameters.

3.1.1 Model Architectures

There are several deep learning architectures such as MLP, CNN, RNN, LSTM [8], transformer [51], autoencoders, GAN and GNN. In the context of DOA estimation, the most commonly used deep learning models are MLP and CNN [6]. This section gives information about the topologies of MLP and transformer which are the base models of the proposed approach.

3.1.1.1 MLP

MLP is a neural network structure which contains input, hidden and output layers composed of artificial neurons. The neurons belonging to the same layer are not connected to each other while there are connections between all neurons at the adjacent layers. The structure of a three-layer MLP is depicted in Figure 3.1.

The connections between the neurons are feedforward and no feedback connection is allowed. Therefore, the information flow is only permitted from input to output direction [93]. These connections represent nonlinear functions which have the form of

$$\boldsymbol{X}^d = f(\boldsymbol{Z}^d) \tag{3.4}$$

where

$$\boldsymbol{Z}^d = \boldsymbol{W}^d \boldsymbol{X}^{d-1} + \boldsymbol{B}^d \tag{3.5}$$

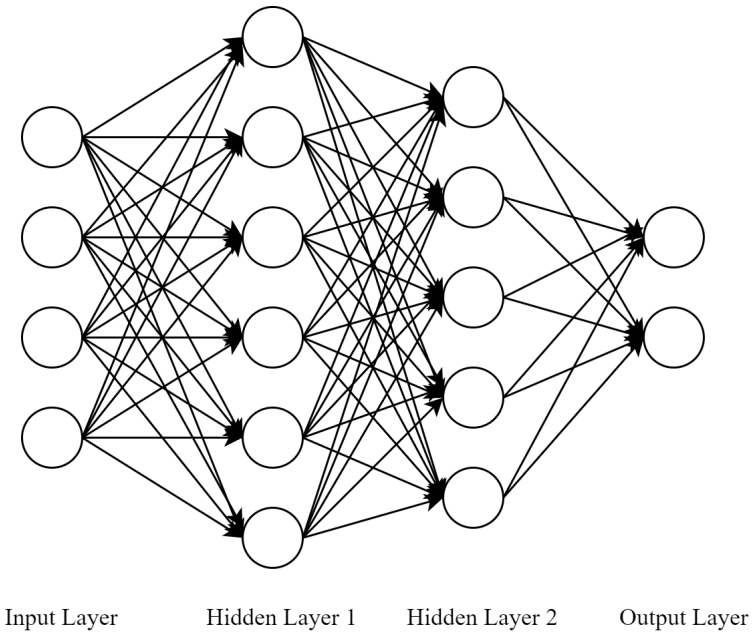

Figure 3.1: Architecture of a 3-layer MLP.

are pre-activation outputs for d^{th} layer, f is the nonlinear activation function, $\boldsymbol{X}^d$ represent the neuron outputs at d^{th} layer, $\boldsymbol{W}^d$ contain the weights between $(d-1)^{th}$ and d^{th} layer and $\boldsymbol{B}^d$ are the bias values for d^{th} layer.

3.1.1.2 The Transformer

The Transformer is composed of encoder and decoder structures in which encoder maps the input sequence into a sequence of representations and decoder uses these representations to generate output sequences. Encoder contains a stack of layers each of which is comprised of multi-head self attention mechanism and fully-connected feedforward network as sublayers. Similarly, decoder also consists of a stack of layers containing the sublayers of encoder with the addition of multi-head attention mechanism applied on the outputs of encoder stack [51]. Figure 3.2 illustrates the architecture of the Transformer model with N encoder and decoder layers.

An input sequence $\boldsymbol{x}_0, \boldsymbol{x}_1, ...\boldsymbol{x}_t$ is fed into encoder structure after applying input embedding and positional encoding. The resultant sequence of representations $\boldsymbol{z}_0, \boldsymbol{z}_1, ...\boldsymbol{z}_t$ are given to the decoder structure along with the output sequence $\boldsymbol{y}_0, \boldsymbol{y}_1, ...\boldsymbol{y}_{t-1}$ which

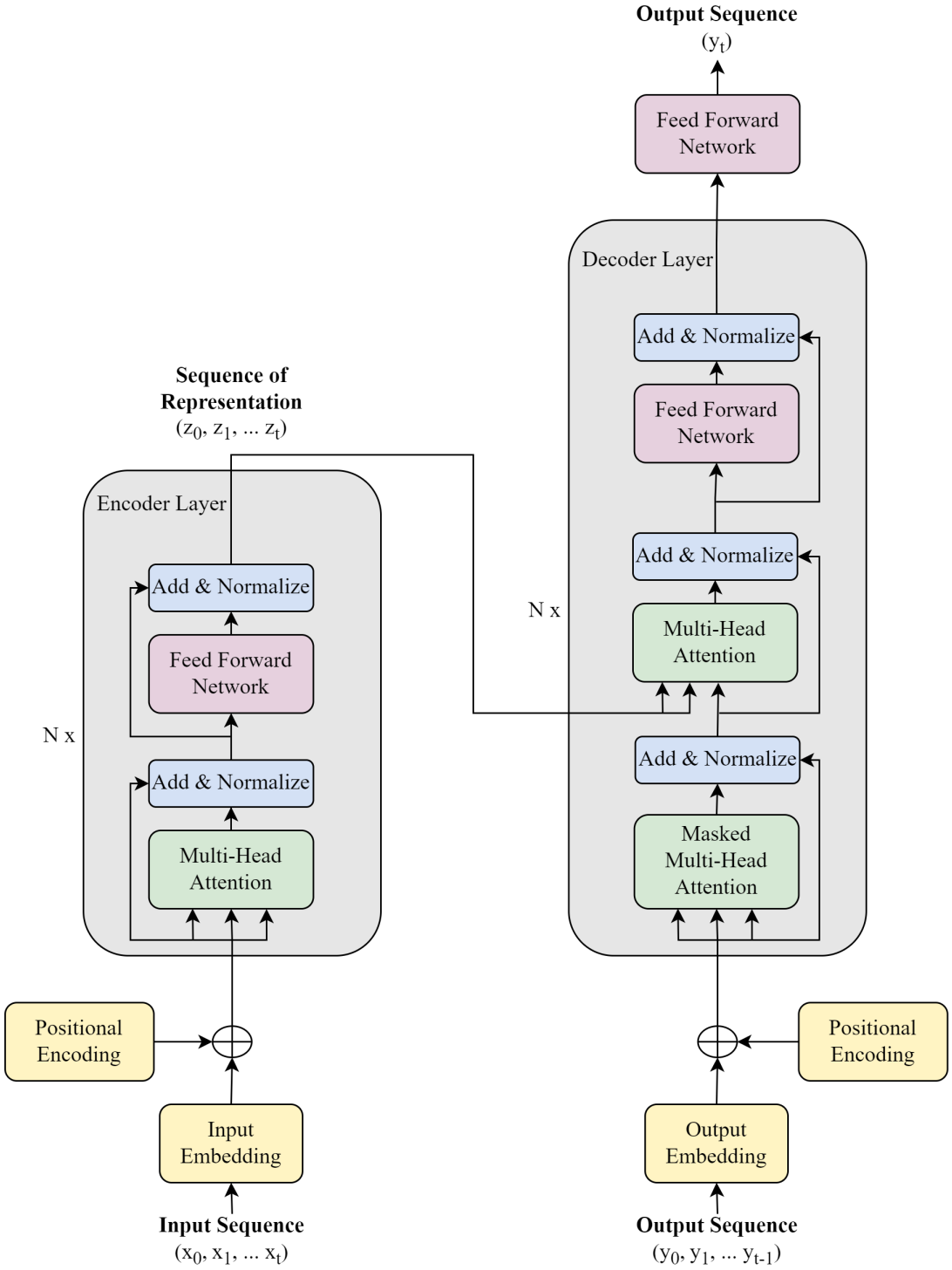

Figure 3.2: Architecture of the Transformer.

contains the outputs of decoder up to sequence index of $t-1$. In the end, decoder generates $\boldsymbol{y}_t$ which is the estimation of the Transformer model for t^{th} index of the output sequence [51].

Input/Output Embedding

The input of the encoder/decoder structure is a sequence which might be discrete or continuous valued. For discrete valued sequences, which are also called as tokens, learned embedding can be applied to convert them into vectors with desired dimension d_{embed} [51]. A similar approach can also be followed for continuous valued sequences for which a neural network is trained to obtain embedding vector with desired dimension d_{embed}. After applying embedding, input sequence element $\boldsymbol{x}_t \in \mathbb{R}^{d_{input}}$ is converted into $\hat{\boldsymbol{x}}_t \in \mathbb{R}^{d_{embed}}$.

Positional Encoding

Since the Transformer model has not a recurrent nature, input and output sequences are fed to the network in parallel. This parallelism reveals the need of information about the absolute or relative position of the sequence elements. For this purpose, positional encoding is applied on input/output sequences after input/output embedding. There are learned and fixed positional encoding techniques [51].

Learned positional encoding uses an embedding matrix $\boldsymbol{D} \in \mathbb{R}^{T_{max} \times d_{embed}}$ where T_{max} is the maximum sequence length. For a sequence element $\hat{\boldsymbol{x}}_t$, $\boldsymbol{w}_t \in \mathbb{R}^{d_{embed}}$ which is t^{th} column of $\boldsymbol{D}$ is selected and

$$\bar{\boldsymbol{x}}_t = \hat{\boldsymbol{x}}_t + \boldsymbol{w}_t \tag{3.6}$$

is obtained after applying positional encoding. Similar operation is done for decoder part as well. The columns in $\boldsymbol{D}$ are learned jointly during training of the network [94].

Fixed positional encoding uses functions which outputs fixed vectors for each absolute or relative positions in the sequence. An example application is the usage of cosine and sine functions of different frequencies and the encoding is achieved by

$$\bar{x}_{(t,2i)} = \hat{x}_{(t,2i)} + \sin\left(\frac{t}{10000^{\frac{2i}{d_{embed}}}}\right) \tag{3.7}$$

and

$$\bar{x}_{(t,2i+1)} = \hat{x}_{(t,2i+1)} + \sin\left(\frac{t}{10000^{\frac{2i+1}{d_{embed}}}}\right) \tag{3.8}$$

where i represents the dimension in $\bar{\boldsymbol{x}}_t$ and $\hat{\boldsymbol{x}}_t$ [51]. An illustration of embedding vectors used in fixed positional encoding with cosine and sine functions is provided in Figure 3.3.

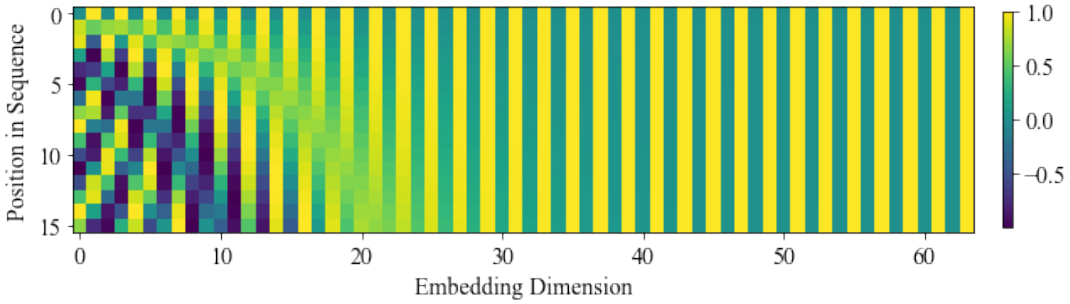

Figure 3.3: Illustration of fixed positional encoding with cosine and sine functions.

Attention Mechanism

Attention mechanism enables the Transformer to model relationships in input and output sequences without any dependency on the distance. It makes use of vectors called as query, key and value and the output of attention mechanism is the weighted sum of value vectors where weights are calculated using the compatibility function calculated for query and key vectors [51].

Additive attention [95] and dot-product attention are the most commonly used attention functions [51]. Additive attention calculates the weights by using a feed-forward network for which the inputs are query and key vectors. Query and key vectors are assigned as $\bar{\boldsymbol{x}}_t$ for each sequence step and calculation of weights can be formulated as

$$\beta_{ij} = b(\boldsymbol{q}_i, \boldsymbol{k}_j) \tag{3.9}$$

and

$$w_{ij} = \frac{e^{\beta_{ij}}}{\sum_{k=1}^{T} e^{\beta_{ik}}} \tag{3.10}$$

where $\boldsymbol{q}_i$ is the i^{th} query, $\boldsymbol{k}_j$ is the j^{th} key, w_{ij} is the weight for i^{th} query and j^{th} key pair, b represents the non-linear functions applied in the feed-forward network and T is the length of input sequence [95].

Scaled dot-product attention firstly finds the query, key and value vectors by

$$\boldsymbol{q}_t = \boldsymbol{W}^Q \bar{\boldsymbol{x}}_t \tag{3.11}$$

$$\boldsymbol{k}_t = \boldsymbol{W}^K \bar{\boldsymbol{x}}_t \tag{3.12}$$

$$\boldsymbol{v}_t = \boldsymbol{W}^V \bar{\boldsymbol{x}}_t \tag{3.13}$$

where $\boldsymbol{W}^Q \in \mathbb{R}^{d_{att} \times d_{embed}}$ is the weight matrix for query, $\boldsymbol{W}^K \in \mathbb{R}^{d_{att} \times d_{embed}}$ is the weight matrix for key, $\boldsymbol{W}^V \in \mathbb{R}^{d_{att} \times d_{embed}}$ is the weight matrix for value vectors and d_{att} is the dimension of each query, key and value vector. These matrices are optimized during training of the network.

After finding these vectors, the weights are calculated by

$$\beta_{ij} = \frac{\boldsymbol{q}_i{}^T \boldsymbol{k}_j}{\sqrt{d_{att}}} \tag{3.14}$$

and

$$w_{ij} = \frac{e^{\beta_{ij}}}{\sum_{k=1}^{T} e^{\beta_{ik}}} \tag{3.15}$$

where where $\boldsymbol{q}_i$ is the i^{th} query, $\boldsymbol{k}_j$ is the j^{th} key and w_{ij} is the weight for i^{th} query and j^{th} key pair [51].

After computing the weights, attention output is found by

$$\boldsymbol{a}_i = \sum_{j=1}^{T} w_{ij} \boldsymbol{v}_j \tag{3.16}$$

Multi-head Attention

Rather than using single attention mechanism, projecting the query, key and value vectors h times with linear projections into lower dimensions with size of $d_{att} = d_{embed}/h$ and applying attention functions in these projected subspaces in parallel allows model to jointly gather information from different representation subspaces [51]. Figure 3.4 shows multi-head attention structure with its components.

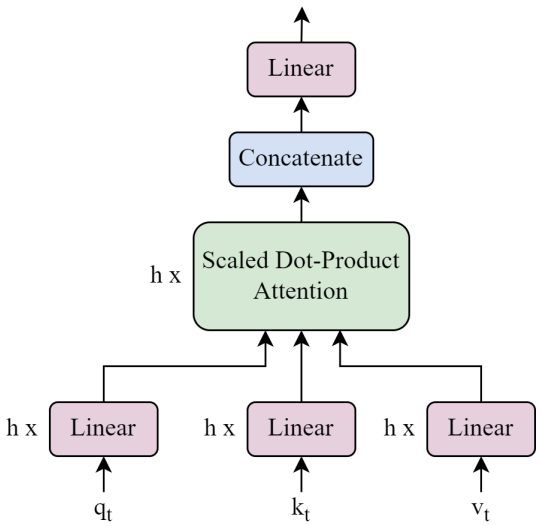

Figure 3.4: Structure of multi-head attention.

For multi-head attention, weight matrices $\boldsymbol{W}^Q$, $\boldsymbol{W}^K$ and $\boldsymbol{W}^V$ are different for each head. Furthermore, scaled dot-product attention is applied independently in parallel for each head. After applying attention, the outputs are concatenated and linearly projected by

$$\boldsymbol{a}_i = \boldsymbol{W}^O Concat(\boldsymbol{a}_i^1, \boldsymbol{a}_i^2, ...\boldsymbol{a}_i^h) \tag{3.17}$$

where $Concat()$ represents the concatenation function , $\boldsymbol{a}_i^k$ is the attention output for i^{th} sequence element and k^{th} head and $\boldsymbol{W}^O$ is the weight matrix for linear projection of the concatenated attention outputs [51].

Multi-head attention is applied in three different parts of transformer model:

- Encoder self-attention: The inputs of the multi-head attention are $\bar{\boldsymbol{x}}_0, \bar{\boldsymbol{x}}_1, ...\bar{\boldsymbol{x}}_t$ and query, key and value vectors are calculated using these inputs. There is no masking applied, hence all of the instances in the sequence attend each other's output [51].

- Decoder self-attention: The inputs of the multi-head attention are $\bar{\boldsymbol{y}}_0, \bar{\boldsymbol{y}}_1, ...\bar{\boldsymbol{y}}_{t-1}$ and query, key and value vectors are calculated using these inputs. Masking is applied such that the instance cannot attend to the output of an instance coming

before. For applying masking, the weight in the scaled dot-product attention is set to $-\infty$ for the corresponding query-key pair [51].

- Encoder-decoder attention: Keys and value vectors are calculated using the encoder output $\boldsymbol{z}_0, \boldsymbol{z}_1, ...\boldsymbol{z}_t$ while query vectors are obtained using $\bar{\boldsymbol{a}}_0, \bar{\boldsymbol{a}}_1, ...\bar{\boldsymbol{a}}_{t-1}$ which are the output of residual connection unit after decoder self-attention. Having attention between encoder and decoder enables all the positions in the decoder to attend all the positions in the sequence of representation [51].

Residual Connections

After every multi-head attention and feed-forward network sublayers, a residual connection with layer normalization is applied. Residual connections ease the optimization in deep networks and provides accuracy gains compared to plain connections [96]. An example application of this connection on the output of encoder's multi-head attention can be formulated as

$$\hat{\boldsymbol{a}}_t = \boldsymbol{a}_t + \bar{\boldsymbol{x}}_t \tag{3.18}$$

where $\boldsymbol{a}_t$ is the output of multi-head attention and $\bar{\boldsymbol{x}}_t$ is the output of positional encoding for t^{th} instance of the input sequence [51].

Layer normalization is applied following residual connections. It decreases training time compared to using batch normalization [97]. Mean and variance values along embedding dimension are calculated for each sequence step and normalization is applied. An example application of this normalization on the output of encoder's multi-head attention can be written as

$$\mu_t = \frac{1}{d_{embed}} \sum_{k=1}^{d_{embed}} \hat{\boldsymbol{a}}_{t,k} \tag{3.19}$$

$$\sigma_t = \sqrt{\frac{1}{d_{embed}} \sum_{k=1}^{d_{embed}} (\hat{\boldsymbol{a}}_{t,k} - \mu_t)^2} \tag{3.20}$$

$$\bar{\boldsymbol{a}}_t = \frac{\hat{\boldsymbol{a}}_t - \mu_t}{\sigma_t} \tag{3.21}$$

where μ_t is mean and σ_t is variance for t^{th} sequence step and $\bar{\boldsymbol{a}}_t$ is the output of layer normalization [98].

Feed-forward Networks

2-layered fully-connected feed-forward networks are used in encoder and decoder structures as well as a final network generating the output sequence instances. It is applied independently on the all sequence instances. The network parameters are shared for different sequence instances while feed-forward networks at different layers have different parameters. Input and output layer of feed-forward network has dimension of d_{embed} while hidden layer can have different dimension represented by d_f. An example application of this network on the output of the encoder's first layer normalization block can be formulated as

$$\boldsymbol{f}_t = \boldsymbol{W}_f^2 f(\boldsymbol{W}_f^1 \bar{\boldsymbol{a}}_t + \boldsymbol{b}_f^1) + \boldsymbol{b}_f^2 \tag{3.22}$$

where f is the nonlinear activation function, $\boldsymbol{W}_f^1$ and $\boldsymbol{W}_f^2$ are the weight matrix of hidden and output layers respectively, $\boldsymbol{b}_f^1$ and $\boldsymbol{b}_f^2$ are the bias vectors of hidden and output layers respectively and $\boldsymbol{f}_t$ is the output of feed-forward network for t^{th} sequence step [51].

3.1.2 Activation Functions

There are several activation functions such as sigmoid, softmax, tanh, ReLU [99] and ELU [100] which can be used to introduce non-linearity into artificial neurons. In DOA estimation domain; sigmoid, tanh and ReLU are the most commonly used activation functions in deep learning based methods [6]. Information about these functions is provided in the below list along with softmax which is utilized by the proposed method.

- Sigmoid: It produces an output in the interval of $[0, 1]$ using

$$f_{\text{sigmoid}}(x) = \frac{1}{1 + e^{-x}} \tag{3.23}$$

Sigmoid function suffers from vanishing gradient caused by small gradient multiplications, bias in the gradients introduced by non-zero centered output, gradient saturation for pre-activations near 0 and 1 and slow convergence due to exponential function [101].

- Tanh: The outputs of this function lie in $[-1, 1]$ and calculated using

$$f_{\text{tanh}}(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{3.24}$$

Tanh function solves the bias problem encountered in sigmoid function since outputs are zero-centered, however, it suffers from vanishing gradient and slow convergence problems [101].

- ReLU: It is a widely used activation function recently due to its ability to decrease the number of active neurons and reduce training time [102–104]. It sets all the negative pre-activation outputs to zero and passes through positive pre-activations as

$$f_{\text{ReLU}}(x) = \max(0, x) \tag{3.25}$$

There are variants of ReLU, such as leaky ReLU, which is proposed for preventing dead neuron problem encountered during training. It achieves this by setting negative pre-activation not to zero but a small positive value for introducing non-zero gradient [105].

- Softmax: It is used to calculate probability distribution over a vector. Therefore, the output of a neuron is affected by other neurons in the layer. Softmax is mostly used on the output neurons especially for classification tasks and it can be formulated as

$$f_{\text{softmax}}(x_i) = \frac{e^{x_i}}{\sum_{k=1}^{K} e^{x_k}} \tag{3.26}$$

where K is the number of neurons in the corresponding layer [93].

3.1.3 Learning Methodologies

Deep learning models are trained for solving specific tasks and these tasks require different types of learning methodologies. These learning methodologies can be broadly categorized as supervised and unsupervised learning. There is also another class of algorithms called as reinforcement learning [93].

- Supervised Learning: Algorithms experience a dataset which contains samples where each sample is associated with a target/label. It is aimed to learn prediction of the target/label by observing the sample. Some example tasks are classification and regression [93, 106].
- Unsupervised Learning: Algorithms experience a dataset containing samples by which probability distribution or important properties about this distribution is learned. There is no target/label assigned for samples in the data. Some examples tasks are synthesis, denoising and clustering [93].
- Reinforcement Learning: Algorithms do not experience a dataset but rather interact with an environment such that learning is performed utilizing the feedbacks generated by the environment [93]. It is applied for the tasks involving decision-making processes [107].

3.1.4 Optimization Algorithms

Deep learning models are trained such that the network parameters are optimized to minimize a specified loss function over training dataset. For this purpose, a process called backpropagation is applied to compute the gradients of the loss function, which is calculated between the network output and desired output, for each sample with respect to the network parameters [108]. The calculated gradients are used in optimization algorithms such as SGD, AdaGrad [109], RMSProp [110] and Adam [111] to optimize the network parameters for minimum loss.

- SGD: It is the most commonly used optimization algorithm in deep learning [93]. A minibatch of samples are selected from training set and parameter update is applied by

$$\boldsymbol{\Psi} \leftarrow \boldsymbol{\Psi} - \eta \boldsymbol{g} \tag{3.27}$$

$$\boldsymbol{g} \leftarrow \frac{1}{m} \boldsymbol{\nabla}_{\boldsymbol{\Psi}} \sum_{i=1}^{m} L(f(\boldsymbol{x}^{(i)}; \boldsymbol{\Psi}), \boldsymbol{y}^{(i)}) \tag{3.28}$$

where $\boldsymbol{\Psi}$ represents the network parameters, η is the learning rate, m is the size of minibatch, L is the loss function, f represents the network inference, $\boldsymbol{x}^{(i)}$ is the i^{th} sample in the minibatch and $\boldsymbol{y}^{(i)}$ is the corresponding label.

SGD can be used with a fixed learning rate however, there is a need for learning rate decay in practice due to noise generated by random sampling of minibatch which doesn't disappear when loss is decreased around minimum level [93].

- AdaGrad: Learning rate is scaled depending on the cumulative gradient calculated for each network parameter. The parameter updates can be formulated as

$$\boldsymbol{\Psi} \leftarrow \boldsymbol{\Psi} - \frac{\eta}{\delta + \sqrt{r}} \odot \boldsymbol{g} \tag{3.29}$$

where accumulated squared gradient is updated by

$$r \leftarrow r + \boldsymbol{g} \odot \boldsymbol{g} \tag{3.30}$$

and gradient is calculated by

$$\boldsymbol{g} \leftarrow \frac{1}{m} \boldsymbol{\nabla}_{\boldsymbol{\Psi}} \sum_{i=1}^{m} L(f(\boldsymbol{x}^{(i)}; \boldsymbol{\Psi}), \boldsymbol{y}^{(i)}) \tag{3.31}$$

In equation 3.29, η is the global learning rate and δ is the small constant for numerical stability [109].

For AdaGrad, it is observed that accumulating all the gradients starting from the beginning of training causes resultant learning rate to decrease excessively [93].

- RMSProp: It modifies AdaGrad algorithm by using a decaying average in the calculation of accumulated gradients. The modification is applied on

$$r \leftarrow \rho r + (1 - \rho) \boldsymbol{g} \odot \boldsymbol{g} \tag{3.32}$$

where ρ is the decay rate. Using weighted average enables discarding the gradients from initial stages of training and prevents excessive decrease in learning rate [110].

- Adam: It combines the advantages of AdaGrad and RMSProp by utilizing first and second order moment of gradients and keeping exponentially decaying average of these moments. In addition, it applies bias correction to the estimates of the moments for enabling their initialization at the origin [111]. It applies the updates using

$$\boldsymbol{\Psi} \leftarrow \boldsymbol{\Psi} - \eta \frac{\hat{\boldsymbol{s}}}{\delta + \sqrt{\hat{r}}} \tag{3.33}$$

where first moment estimate $\boldsymbol{s}$, second moment estimate $\boldsymbol{r}$ and their bias corrected versions $\hat{\boldsymbol{s}}$ and $\hat{\boldsymbol{r}}$ are found using

$$\boldsymbol{s} \leftarrow \beta_1 \boldsymbol{s} + (1 - \beta_1)\boldsymbol{g} \tag{3.34}$$

$$\boldsymbol{r} \leftarrow \beta_2 \boldsymbol{r} + (1 - \beta_2)\boldsymbol{g} \odot \boldsymbol{g} \tag{3.35}$$

$$\hat{\boldsymbol{s}} \leftarrow \frac{\boldsymbol{s}}{1 - \beta_1^t} \tag{3.36}$$

$$\hat{\boldsymbol{r}} \leftarrow \frac{\boldsymbol{r}}{1 - \beta_2^t} \tag{3.37}$$

where β_1 and β_2 are decay rates for first and second order moments respectively and t is the time step in the optimization process. Gradients are calculated similar to SGD, AdaGrad and RMSProp using

$$\boldsymbol{g} \leftarrow \frac{1}{m} \boldsymbol{\nabla}_{\boldsymbol{\Psi}} \sum_{i=1}^{m} L(f(\boldsymbol{x}^{(i)}; \boldsymbol{\Psi}), \boldsymbol{y}^{(i)}) \tag{3.38}$$

3.1.5 Regularization Techniques

Regularization techniques are used for deep learning models to prevent overfitting and reduce the generalization error [93]. Dropout [112] and parameter norm penalties such as L1 regularization [40] and L2 regularization [113] are some examples of these techniques.

- Dropout: It is applied by disabling neurons and their incoming/outgoing connections randomly in a network. It provides a way to combine different network structures efficiently since at each iteration different neurons can be removed. The probability of removal for each neuron is p. Removal process is only applied during training while the output of each neuron is scaled by p during testing phase [112].

- Parameter Norm Penalties: Penalty terms are added into loss function such that the regularized loss function is

$$\bar{L}(\boldsymbol{X}, \boldsymbol{Y}; \boldsymbol{\Psi}) = L(\boldsymbol{X}, \boldsymbol{Y}; \boldsymbol{\Psi}) + \lambda \Omega(\boldsymbol{\Psi}) \tag{3.39}$$

  where L is the loss function without regularization, $\boldsymbol{X}$ and $\boldsymbol{Y}$ represents inputs and desired outputs, $\boldsymbol{\Psi}$ represents the network parameters, λ is the regularization coefficient and Ω is the penalty term.

  L1 and L2 regularization are two types of norm penalties. For L1 regularization, penalty term is

$$\Omega(\boldsymbol{\Psi}) = \|\boldsymbol{\Psi}\|_1 = \sum_i |\Psi_i| \tag{3.40}$$

  while L2 regularization applies

$$\Omega(\boldsymbol{\Psi}) = \|\boldsymbol{\Psi}\|_2^2 = \sum_i \Psi_i^2 \tag{3.41}$$

  as the penalty term [40, 113].

3.2 Proposed Architecture

A data-driven method is proposed for the tasks of DOA estimation and source enumeration. DOA estimation problem is formulated as multi-label classification task where on-grid approach is followed. Therefore, the proposed method outputs on-grid DOA angle estimates for source signals.

The input for the proposed method is selected to be sample covariance matrix for achieving generalization since training data-driven methods using raw sensor signals might lead to performance degradation in case of unseen types of source signals. Sample covariance matrix carries correlation of received sensor signals and it is independent of the source signal types.

The proposed method is primarily composed of three stages:

1. Covariance Reconstruction Network
2. DOA Estimation Network
3. Source Enumeration Network

The networks that form these stages are based on deep learning models. Covariance reconstruction network and DOA estimation network are built upon transformer while source enumeration network consists of MLP. An overview of the proposed method is depicted in Figure 3.5. As an initial step, input conversion is applied on the sample covariance matrix. Covariance reconstruction network and DOA estimation network operate sequentially while there is a feedback connection between DOA estimation network and source enumeration network.

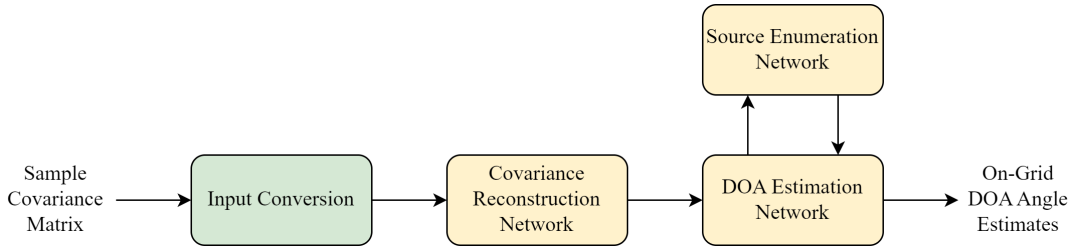

Figure 3.5: Overview of the proposed method.

3.2.1 Input Conversion

Sample covariance matrix, which is the input of the proposed method, is not directly fed into covariance reconstruction network. A conversion is applied so that it can be considered as a sequential data. In this conversion, coarray structure of the array

is exploited and the mean of covariance matrix entries corresponding to the same lag in the coarray is calculated. This conversion process is explained with details in Algorithm 1.

Algorithm 1 Input (sample covariance matrix) conversion algorithm

Require: Aperture of the array A

Require: The number of sensors in the array M

Require: Sensor position coefficients (in terms of 0.5λ) $\boldsymbol{p} \in \mathbb{R}^M$

Require: Sample covariance matrix $\hat{\boldsymbol{R}} \in \mathbb{C}^{M \times M}$

Initialize sample covariance coarray $\hat{\boldsymbol{c}} = 0, \hat{\boldsymbol{c}} \in \mathbb{C}^A$

Initialize counter for sample covariance coarray $\boldsymbol{n} = 0, \boldsymbol{n} \in \mathbb{R}^A$

for $i \leftarrow 1$ to M **do**

for $j \leftarrow 1$ to M **do**

Find the lag between sensors: $k \leftarrow \boldsymbol{p}(i) - \boldsymbol{p}(j)$

if $k > 0$ **then**

Add the entry to covariance coarray: $\hat{\boldsymbol{c}}(k) \leftarrow \frac{\boldsymbol{n}(k)\hat{\boldsymbol{c}}(k)+\hat{\boldsymbol{R}}(i,j)}{\boldsymbol{n}(k)+1}$

else

Add conj. of the entry to covariance coarray: $\hat{\boldsymbol{c}}(|k|) \leftarrow \frac{\boldsymbol{n}(|k|)\hat{\boldsymbol{c}}(|k|)+\hat{\boldsymbol{R}}^*(i,j)}{\boldsymbol{n}(|k|)+1}$

end if

Increment counter: $\boldsymbol{n}(|k|) \leftarrow \boldsymbol{n}(|k|) + 1$

end for

end for

The reasons of applying such a conversion are as follows:

- Covariance matrix entries corresponding to the same coarray lag are ideally identical since the same phase difference is observed in the received signals of sensors which have the same location difference. Calculating mean of these entries reduces the approximation error that is present in the sample covariance matrix.
- The resultant covariance coarray $\hat{\boldsymbol{c}}$ can be considered as a sequential data which exhibits unique sequential patterns depending on the DOA angle of the source. Example illustrations for these patterns are given in Appendix A.
- Formulating the input of subsequent networks as sequential data bring the pro-

posed method the ability of functioning without any need of retraining in case of sensor malfunctions. This ability is clarified in Section 3.2.2.

3.2.2 Stage 1: Covariance Reconstruction Network

Sample covariance matrix is unbiased estimate of true covariance matrix and approximation error depends on the number of samples and the level of noise in the data with which sample covariance matrix is calculated. Applying input conversion algorithm, see Algorithm 1, decreases the approximation error for a certain level however, there is still a need for improvement especially for low SNR levels. The primary function of covariance reconstruction network is to reduce the approximation error further.

The input of the covariance reconstruction network is

- real components of sample covariance coarray $\hat{\boldsymbol{c}}$
- imaginary components of sample covariance coarray $\hat{\boldsymbol{c}}$

and the output is its reconstructed version $\hat{\boldsymbol{y}}$. Transformer model is employed in the network. The reason for selecting transformer rather than other models such as MLP, CNN and LSTM is its ability of functioning without retraining in case of sensor malfunctions. Sensor malfunctions in essential elements introduces holes in the coarray [13] and the performance of recurrent models severely degrades since there will be no data for the lags corresponding to new holes. Moreover, MLP and CNN cannot function anymore since the input size becomes different from what they are trained with. Therefore, they need to be retrained for re-functioning and obtaining their usual performance. However, transformer model can apply masking in its multi-head attention unit for the lags corresponding to the holes as described in Section 3.1.1.2. Also, it can embed the lag positions into the inputs owing to positional encoding and learn the spatial arrangements of the hole-free lags. This helps the network to function without any need of retraining.

The architecture of the network is shown in Figure 3.6. It consists of transformer encoder followed by a 2-layer feedforward network. The reason of using only encoder

structure in transformer is to reduce the computational complexity. This is a viable approach since the input and output sequences have the same length. Feedforward network is applied on the instances of encoder output sequence separately.

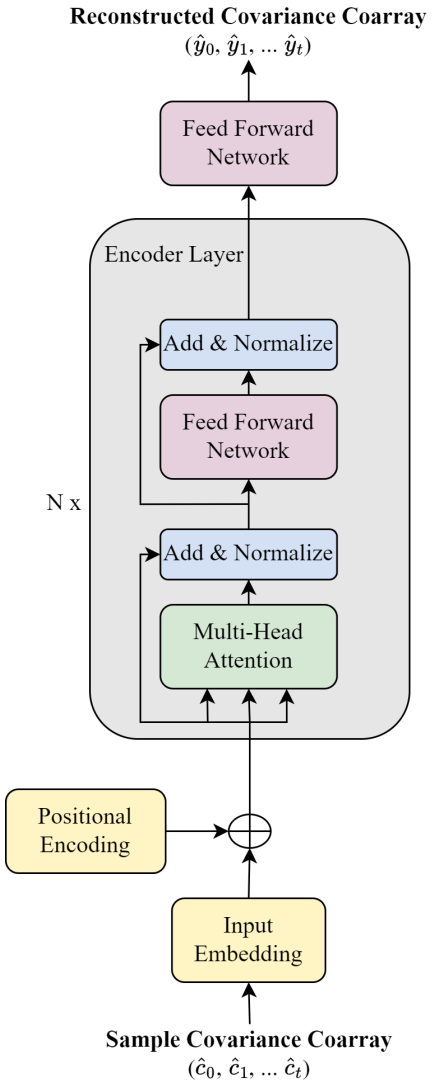

Figure 3.6: Architecture of covariance reconstruction network.

Covariance reconstruction network can be mathematically represented by

$$(\hat{\boldsymbol{r}}_0, \hat{\boldsymbol{r}}_1, ...\hat{\boldsymbol{r}}_t) = f_{\text{enc}}(\hat{\boldsymbol{c}}_0, \hat{\boldsymbol{c}}_1, ...\hat{\boldsymbol{c}}_t; \boldsymbol{\Psi}_{\text{CRN}}) \tag{3.42}$$

and

$$\hat{\boldsymbol{y}}_i = f_{\text{ffn}}(\hat{\boldsymbol{r}}_i; \boldsymbol{\Psi}_{\text{CRN}}) \qquad \forall i \in [0, ..., t] \tag{3.43}$$

where f_{enc} and f_{ffn} represent the operations done by transformer encoder and feedfor-

ward network as described in Section 3.1.1.2, $\mathbf{\Psi}_{\text{CRN}}$ represents the parameters of the network, $(\hat{\boldsymbol{r}}_0, \hat{\boldsymbol{r}}_1, ...\hat{\boldsymbol{r}}_t)$ is the sequence of representation produced by encoder, $\hat{\boldsymbol{y}}_i$ is i^{th} instance of the reconstructed covariance coarray.

Table 3.1 provides additional information about the architecture of the network. The output dimension of last feedforward network is 2 which represents the real and imaginary components of a single instance of reconstructed covariance coarray. As positional encoding, fixed embedding with cosine and sine functions is preferred to have lower computational complexity since fixed and learned embedding produce similar results [51]. Scaled dot-product attention is selected for attention units because dot-product attention is faster and space-efficient compared to additive attention [51]. ReLU is used as activation function both in encoder and last feedforward network. The reason for selecting ReLU is that it doesn't experience slow convergence and vanishing gradient problems [101].

Table 3.1: Information about architecture of covariance reconstruction network

Transformer Encoder	Embedding Type	Learned embedding using 2-layer NN
	Embedding Dimension	128
	Positional Encoding Type	Fixed embedding using cosine and sine functions
	Number of Layers (N)	12
	Number of Heads (h)	8
	Attention Type	Scaled dot-product attention
	Feedforward Network Hidden Dimension	128
	Feedforward Network Activation Function	ReLU
	Regularization	Dropout with $p = 0.1$
Feedforward Network	Hidden Dimension	64
	Hidden Activation Function	ReLU
	Output Dimension	2
	Output Activation Function	Linear

3.2.3 Stage 2: Direction of Arrival Estimation Network

Covariance coarrays exhibit different patterns for different DOA angles. For attaining the mapping between coarrays and angles, DOA estimation network is proposed.

The input of DOA estimation network is

- real components of the reconstructed covariance coarray $\hat{\boldsymbol{y}}$
- imaginary components of the reconstructed covariance coarray $\hat{\boldsymbol{y}}$
- sine of the phase components of the reconstructed covariance coarray $\hat{\boldsymbol{y}}$
- cosine of the phase components of the reconstructed covariance coarray $\hat{\boldsymbol{y}}$

The reason for using sine and cosine of the phase component rather than direct usage of it is that the phase component presents discontinuities and these discontinuities might introduce additional bias into the learning process. By mapping the phase into two dimensions using sine and cosine functions, these continuities are prevented. The output of the network is the estimated spatial spectrum over a grid. The network is constructed to be based on transformer model due to the same reasons stated for covariance reconstruction network in Section 3.2.2.

The architecture of the network is depicted in Figure 3.7. It contains transformer encoder, global average pooling and a 2-layer feedforward network. Since the problem is formulated as multi-label classification task, there is no need for the usage of decoder of transformer. Global average pooling is performed on the output sequence of encoder over sequence length dimension and the resultant vector is fed into feedforward network to obtain spatial spectrum estimate.

DOA estimation network can be mathematically expressed by

$$(\hat{\boldsymbol{r}}_0, \hat{\boldsymbol{r}}_1, ...\hat{\boldsymbol{r}}_t) = f_{\text{enc}}(\hat{\boldsymbol{y}}_0, \hat{\boldsymbol{y}}_1, ...\hat{\boldsymbol{y}}_t; \boldsymbol{\Psi}_{\text{DAEN}}) \tag{3.44}$$

$$\hat{\boldsymbol{r}} = \frac{1}{t+1}\sum_{i=0}^{t}\hat{\boldsymbol{r}}_i \tag{3.45}$$

$$\hat{\boldsymbol{z}} = f_{\text{ffn}}(\hat{\boldsymbol{r}}; \boldsymbol{\Psi}_{\text{DAEN}}) \tag{3.46}$$

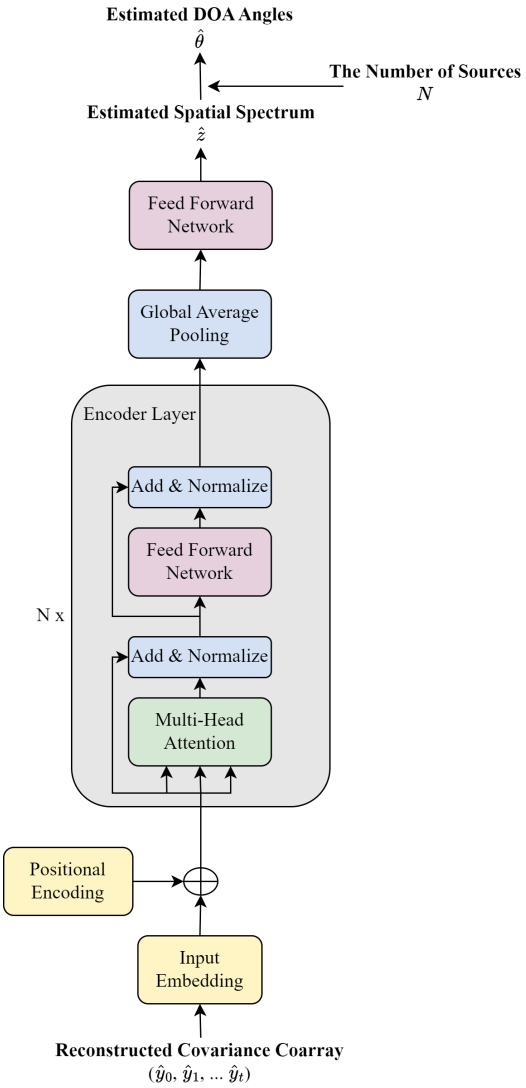

Figure 3.7: Architecture of DOA estimation network.

where f_{enc} and f_{ffn} represent the operations done by transformer encoder and feedforward network as described in Section 3.1.1.2, $\boldsymbol{\Psi}_{\text{DAEN}}$ represents the parameters of the network, $(\hat{\boldsymbol{r}}_0, \hat{\boldsymbol{r}}_1, ...\hat{\boldsymbol{r}}_t)$ is the sequence of representation produced by encoder and $\hat{\boldsymbol{z}}$ is the estimated spatial spectrum.

Table 3.2 contains further information about the network architecture. Embedding type, positional encoding type, attention type and activation function are selected due to the same reasons as covariance reconstruction network that are mentioned in Section 3.2.2. At the last feedforward network, output activation function is chosen to be sigmoid since the problem is formulated as multi-label classification task.

Table 3.2: Information about architecture of DOA estimation network

Transformer Encoder	Embedding Type	Learned embedding using 2-layer NN
	Embedding Dimension	32
	Positional Encoding Type	Fixed embedding using cosine and sine functions
	Number of Layers (N)	6
	Number of Heads (h)	4
	Attention Type	Scaled dot-product attention
	Feedforward Network Hidden Dimension	64
	Feedforward Network Activation Function	ReLU
	Regularization	Dropout with $p = 0.1$
Feedforward Network	Hidden Dimension	256
	Hidden Activation Function	ReLU
	Output Dimension (G)	121
	Output Activation Function	Sigmoid

DOA estimation network generates DOA angle estimates over the estimated spatial spectrum over two scenarios:

- Known Number of Sources: The number of sources N is known beforehand. The highest N local maxima points are found from the spatial spectrum and corresponding grid angle(s) are declared as source DOA angle estimates.

- Unknown Number of Sources: The number of sources N is unknown. Source enumeration network produces source number estimate $\hat{N}$. The highest $\hat{N}$ local maxima points are found from the spatial spectrum and corresponding grid angle(s) are declared as source DOA angle estimates.

3.2.4 Stage 3: Source Enumeration Network

In practice, the number of sources in an environment is mostly unknown and there arises a need for estimating the number of sources. For this purpose, source enumeration network is constructed in the proposed method.

The architecture of the network is given along with DOA estimation network in Figure 3.8. The input of the network is the output of global average pooling block in DOA estimation network. DOA estimation network uses this feature vector for obtaining spatial spectrum estimate which indicates that this vector contains information about the number of sources as well. Therefore, it can be processed by a feedforward network for obtaining source number estimates as output.

Source enumeration network can be mathematically modeled by

$$\hat{\boldsymbol{o}} = f_{\text{ffn}}(\boldsymbol{p}; \boldsymbol{\Psi}_{\text{SEN}}) \tag{3.47}$$

and

$$\hat{N} = \arg\max \hat{\boldsymbol{o}} \tag{3.48}$$

where f_{ffn} represent the operations done by feedforward network as described in Section 3.1.1.1, $\boldsymbol{\Psi}_{\text{SEN}}$ represents the parameters of the network, $\boldsymbol{p}$ is the output of global average pooling block in DOA estimation network, $\hat{\boldsymbol{o}}$ carries the output probabilities for each number of sources and $\hat{N}$ is the source number estimate.

Table 3.3 shows more information about the architecture of the network. ReLU is used as activation function of hidden layer due to the reasons explained for previous networks in Section 3.2.2. Output activation function is softmax since a single estimation is to be made for the number of sources.

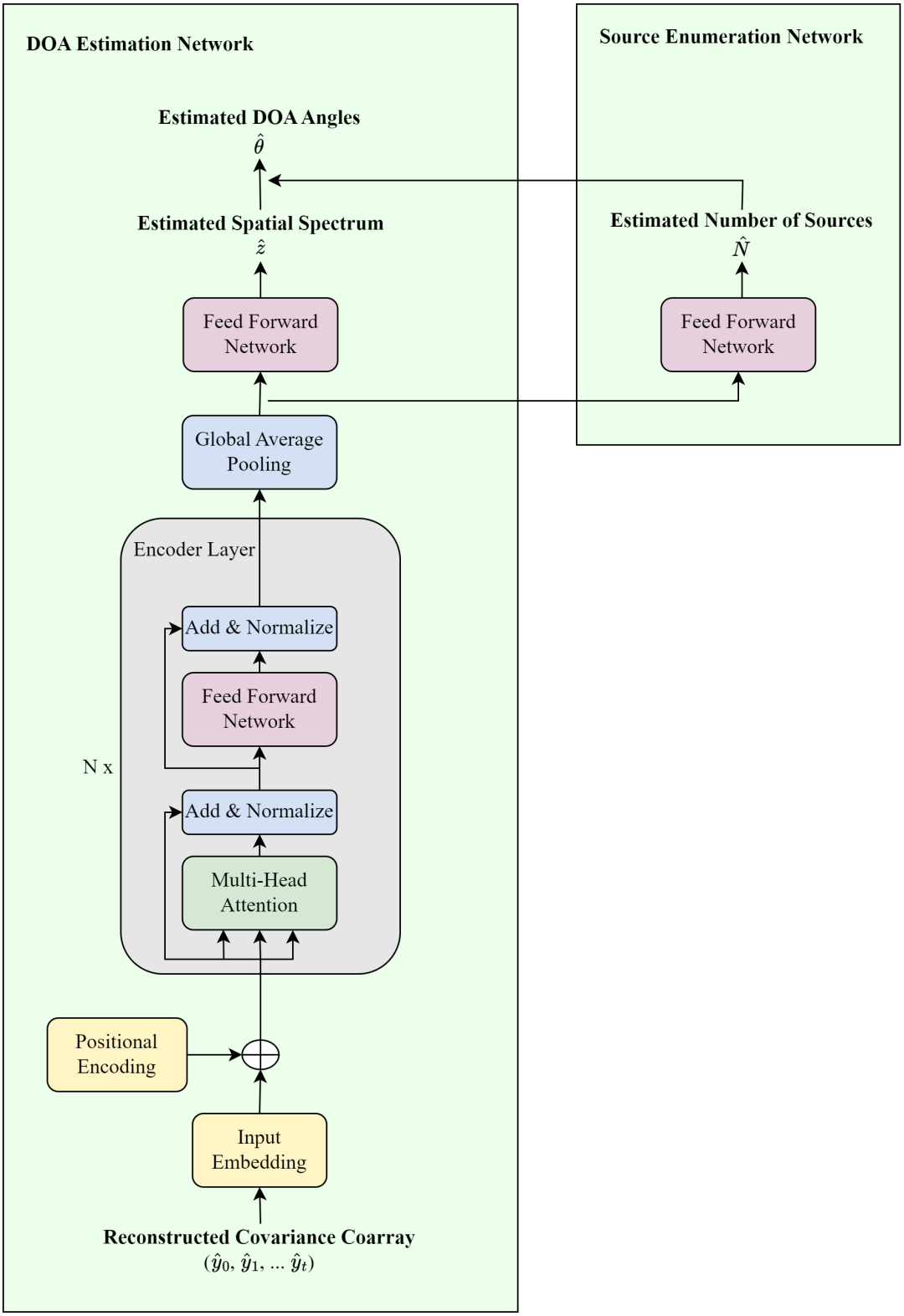

Figure 3.8: Architecture of source enumeration network.

Table 3.3: Information about architecture of source enumeration network

Feedforward Network	Hidden Dimension	256
	Hidden Activation Function	ReLU
	Output Dimension (N_{max})	4
	Output Activation Function	Softmax

CHAPTER 4

PERFORMANCE ANALYSIS: SPARSE ARRAYS

In this chapter, extensive simulation results are provided for transformer-based DOA estimation method where its performance is evaluated for different types of sparse arrays such as minimum redundant array, nested array, and coprime array. The performance comparison is made with the following methods:

1. MUSIC [22]
2. Capon Beamformer [27]
3. L1-SVD [37]
4. CNN [28]
5. MLP [4]
6. DAE + SS + MUSIC [46]
7. CNN + Root-MUSIC [66]

MUSIC [22] is selected as a representative of subspace-based methods since it is a widely used method with super-resolution ability. ESPRIT [29], another successful example of subspace-based methods, is not used in the comparison since the constraint that it imposes on the structure of the array cannot always be established for the aforementioned sparse array types. Capon Beamformer [27] is used for representing beamforming methods since it is a well-known method with its capability of providing good performance in the presence of noise. As an example for sparsity-inducing methods, L1-SVD [37] is chosen for its effectiveness in case of limited data.

As recent data-driven approaches, CNN [28] and MLP [4] are used in the comparison. These methods are the mostly used deep learning architectures in DOA estimation task and this explains the usage of them in the comparison [6]. Lastly, DAE + SS + MUSIC [46] and CNN + Root-MUSIC [66] are used as the representatives of the hybrid methods. DAE + SS + MUSIC [46] utilizes DAE for the purpose of covariance matrix reconstruction which is also a stage in the proposed method and this makes it a potential candidate to make comparison with. CNN + Root-MUSIC [66] utilizes one of the most commonly used deep learning networks along with a variant of MUSIC [22] with gridless output. Since the proposed method is on-grid method, including CNN + Root-MUSIC [66] in the comparison provides an opportunity to observe the differences between on-grid and gridless methods.

For performing the comparisons, a single instance is selected for each of the above-stated sparse array classes. Figures 4.1, 4.2, and 4.3 illustrate the array configurations and corresponding difference coarrays for the selected instances of these sparse array types. It can be observed that the minimum redundant array and nested array are hole-free while coprime array has holes which degrades the degree of freedom. However, coprime array possesses larger aperture compared to others.

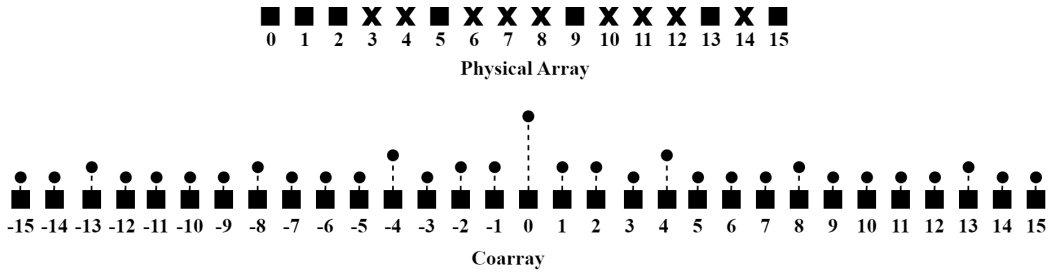

Figure 4.1: Minimum redundant array configuration that is used in the performance comparisons (top) and its coarray (bottom). Rectangle represents presence of the element while cross indicates its absence. The number of dashes in the dashed lines represent the weights for each lag.

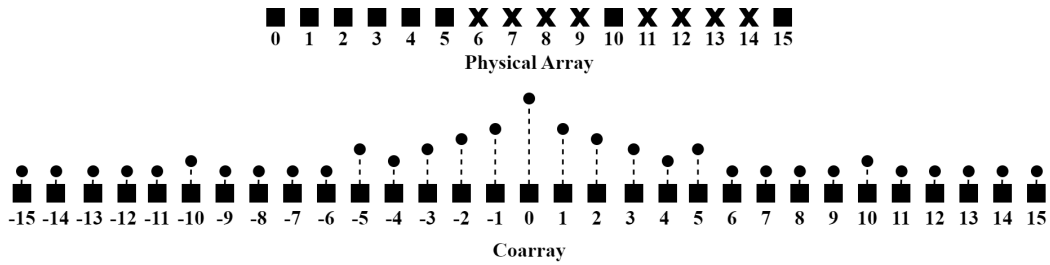

Figure 4.2: Nested array configuration that is used in the performance comparisons (top) and its coarray (bottom). Rectangle represents presence of the element while cross indicates its absence. The number of dashes in the dashed lines represent the weights for each lag.

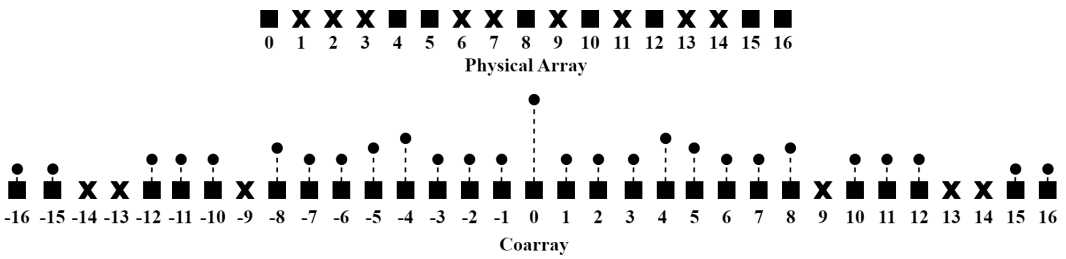

Figure 4.3: Coprime array configuration that is used in the performance comparisons (top) and its coarray (bottom). Rectangle represents presence of the element while cross indicates its absence. The number of dashes in the dashed lines represent the weights for each lag.

4.1 Simulation Details

For each of the provided array configurations, data-driven methods are trained with the synthetic data that is generated with the specifications given in Table 4.1. The number of samples is selected such that there are sufficient number of samples to represent the distribution which relates sample covariance matrices with corresponding DOA angles. Multiple sources are generated since it is more possible to experience in the practice compared to single source case. Source separations are selected to be not too close since the main focus of this study is not resolution ability of the proposed method. Separations are varying for obtaining a data distribution closer to the true distribution experienced in DOA estimation task. Azimuth direction range

is selected considering the standards that are followed in the existing research for obtaining consistent results [4, 28, 46, 66]. Elevation is chosen as 0° since linear arrays allow direction finding only in a single dimension. SNR and snapshot levels are determined to have low values since the network would generalize to less noisy and data-sparse conditions after training. Noise signals are chosen to be complex random Gaussian as indicated in the assumptions specified in Section 2.1.

Table 4.1: The specifications of training set

The number of samples (D)	2000000
The number of sources at each sample (N)	4
Source separations	$>10^\circ$
Source direction (azimuth, θ)	30° - 150°*
Source direction (elevation, ϕ)	0°
The number of snapshots (T)	100 - 1000*
SNR	-20 dB - 0 dB*
Source signal type	Complex random Gaussian
Noise signal type	Complex random Gaussian
Sampling rate	10 kHz

* Uniformly selected between the given interval

The proposed transformer-based network is trained in two steps. In both steps, the network is implemented in PyTorch framework and training is performed using Tesla T4 GPU. The first step consists of training the covariance reconstruction network which forms the first stage of the overall network. During training, true covariance coarray is used as ground truth such that denoising can be achieved after training. Hold-out cross-validation is applied with a validation ratio of 0.1. Training is applied offline in a supervised manner such that mean square error between the reconstructed and true covariance coarray over validation dataset is minimized. The trainable parameters $\mathbf{\Psi}_{\text{CRN}}$ of the network are optimized by the updates conducted by back-propagation and this optimization process can be formulized as

$$\mathbf{\Psi}^*_{\text{CRN}} = \underset{\mathbf{\Psi}_{\text{CRN}}}{\arg\min} \frac{1}{D} \sum_{d=1}^{D} \|\boldsymbol{y_d} - \hat{\boldsymbol{y}}_{\boldsymbol{d}}\|^2 \tag{4.1}$$

where $\boldsymbol{y}_d$ represents the ground truth and $\hat{\boldsymbol{y}}_d$ indicates the network output for d^{th}

sample in the validation set. Other training hyperparameters for the covariance reconstruction network are summarized in Table 4.2. Adam [111] is chosen as optimizer due to the advantages it provides over other optimization algorithms as described in Section 3.1.4. The hyperparameters of Adam [111], learning rate, learning rate decay and batch size are selected by hyperparameter tuning.

Table 4.2: Training hyperparameters

Optimizer	Adam [111] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$
Learning rate	Initially 0.001
Learning rate decay	0.7 once validation loss reaches plateau with a patience of 10 epochs
Batch size	512

The second step of the training process comprises of training DOA estimation network. Since the problem is formulated as multi-label classification task, ground truths in training set are formed of binary vectors in which the entries corresponding to a source direction grid are 1 and others are 0. The output layer of the network is constructed to have $G = 121$ neurons such that 1° of resolution is obtained because DOA angle range is set as [30, 150]$^\circ$ in training set. Similar to covariance reconstruction network, offline and supervised learning are applied. The parameters $\mathbf{\Psi}_{\text{DAEN}}$ of the network is optimized by minimizing the error

$$\mathbf{\Psi}^*_{\text{DAEN}} = \underset{\mathbf{\Psi}_{\text{DAEN}}}{\arg\min} \frac{1}{D} \sum_{d=1}^{D} L(\boldsymbol{z}_d, \hat{\boldsymbol{z}}_d) \tag{4.2}$$

where

$$L(\boldsymbol{z}_d, \hat{\boldsymbol{z}}_d) = -\frac{1}{G} \sum_{g=1}^{G} [\boldsymbol{z}_d(g) \log(\hat{\boldsymbol{z}}_d(g)) + (1 - \boldsymbol{z}_d(g)) \log(1 - \hat{\boldsymbol{z}}_d(g))] \tag{4.3}$$

represents binary cross-entropy loss, $\boldsymbol{z}_d$ is the ground truth and $\hat{\boldsymbol{z}}_d$ shows the network output for d^{th} sample in the validation set. Other training hyperparameters are the same as in the case of covariance reconstruction network which are summarized in Table 4.2.

For training the data-driven and hybrid methods that are used in the comparison, the

same training set is used for a fair evaluation. Training processes of these methods are applied by following the procedures and using the hyperparameters that are provided in the corresponding articles [4, 28, 46, 66].

For performance evaluation, various test sets are generated which exhibit different SNR levels and snapshot numbers. Signal types, sampling rate, source elevation and the number of sources are the same as in training set. Other details about these sets can be found in Table 4.3.

Table 4.3: The specifications of test sets

Test set #	Number of samples (D)	Source separations	Source direction (θ)	SNR	Number of snapshots (T)
1	10000	$>10^\circ$	30° - 150°*	-20 dB	1000
2	10000	$>10^\circ$	30° - 150°*	-15 dB	1000
3	10000	$>10^\circ$	30° - 150°*	-10 dB	1000
4	10000	$>10^\circ$	30° - 150°*	-5 dB	1000
5	10000	$>10^\circ$	30° - 150°*	0 dB	1000
6	10000	$>10^\circ$	30° - 150°*	5 dB	1000
7	10000	$>10^\circ$	30° - 150°*	10 dB	1000
8	10000	$>10^\circ$	30° - 150°*	15 dB	1000
9	10000	$>10^\circ$	30° - 150°*	20 dB	1000
10	10000	$>10^\circ$	30° - 150°*	25 dB	1000
11	10000	$>10^\circ$	30° - 150°*	30 dB	1000
12	10000	$>10^\circ$	30° - 150°*	-10 dB	100
13	10000	$>10^\circ$	30° - 150°*	-10 dB	200
14	10000	$>10^\circ$	30° - 150°*	-10 dB	500
15	10000	$>10^\circ$	30° - 150°*	-10 dB	1000
16	10000	$>10^\circ$	30° - 150°*	-10 dB	2000
17	10000	$>10^\circ$	30° - 150°*	-10 dB	5000
18	10000	$>10^\circ$	30° - 150°*	-10 dB	10000

* Uniformly selected between the given interval

For evaluating and comparing the results on the test sets, root mean square error (RMSE) metric is utilized which can be formulized as

$$\text{RMSE} = \sqrt{\frac{1}{D}\sum_{d=1}^{D}\|\boldsymbol{\theta}_d - \hat{\boldsymbol{\theta}}_d\|^2} \tag{4.4}$$

where $\boldsymbol{\theta}_d$ is the ground truth DOA angles and $\hat{\boldsymbol{\theta}}_d$ is the estimated DOA angles.

Regularization parameter that is used for L1-SVD [37] is set to 0.1 during experiment.

4.2 Results and Discussion

Simulation results for each sparse array type are provided in the following subsections.

4.2.1 Minimum Redundant Array

The loss curves obtained for covariance reconstruction network are shown in Figure 4.4. Training lasted for 53 epochs where validation loss converges during the last epochs. It can be observed that validation loss is lower throughout training which stems from the dropout layer in the network that is activated during training. A similar case happens during training CNN as illustrated in the corresponding article [28].

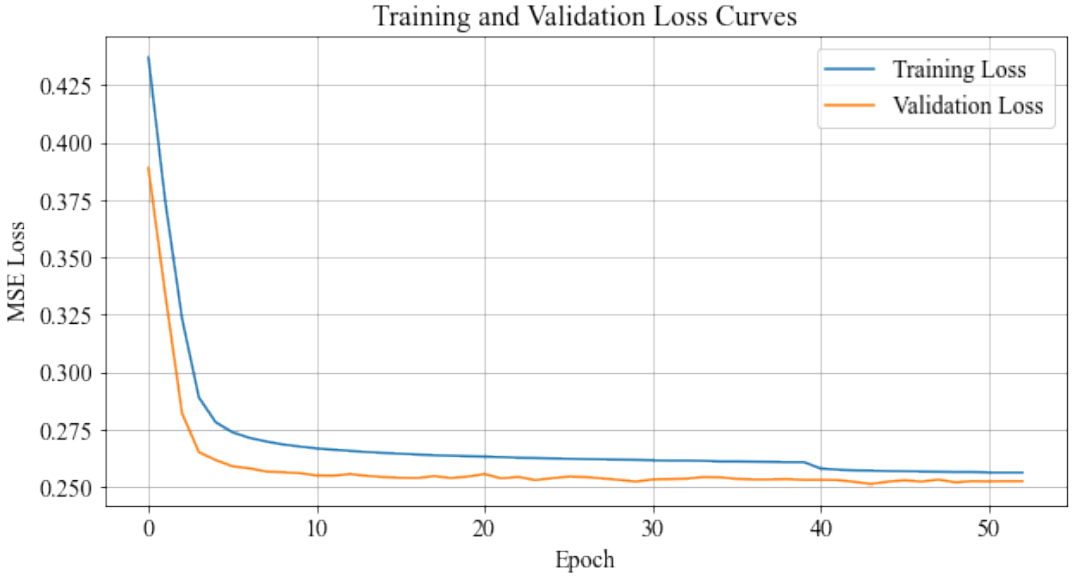

Figure 4.4: Loss curves for covariance reconstruction network.

For illustrating the performance of covariance reconstruction network, random samples are selected from the test sets. Figure 4.5 illustrates the phase component of the measured covariance matrices belonging to these samples, true versions which do not contain any noise and reconstructed versions after applying covariance reconstruc-

tion network. It can be observed that the reconstructed covariance matrices are more similar to true covariance matrices compared to the measured ones especially for low SNR and small number of snapshots. This observation is validated by calculating MSE between the phase components of the true covariance matrices and measured and reconstructed covariance matrices as given in Figures 4.6 and 4.7. Having lower difference with the true covariance matrix demonstrates the denoising capability of the proposed reconstruction network. In addition, being able to reconstruct for SNR levels between 0 and 30 dB proves the generalization ability of the network since these SNR levels are not present in training set.

For direction of arrival estimation network, the loss curves that are presented in Figure 4.8 are obtained during training process. Training is applied for 52 epochs and the parameters Ψ_{DAEN} are selected for which the lowest validation loss is obtained during this process.

After training the first two stages of the proposed method, performance comparisons are made on test sets with other methods. Figure 4.9 and 4.10 show the RMSE values for different SNR levels and snapshot numbers. Results indicate that the proposed method leads to lower RMSE for low SNR regime and small number of snapshots. This demonstrates the resilience of the proposed method to the presence of noise and relatively small amount of data collected from the sensors. As SNR and snapshot number increases, RMSE values converge to a level due to grid mismatch. This flooring effect is less for CNN + Root-MUSIC [66] since it provides gridless estimates. Therefore, the best performance in the high SNR and snapshot number region is achieved by CNN + Root-MUSIC [66]. Among on-grid methods, data-driven and hybrid methods (the proposed method, CNN [28], MLP [4], DAE + SS + MUSIC [46]) provide similar performance in the bound of 0.05° of RMSE. MUSIC [22] and L1-SVD [37] show relatively poor performance compared to other methods. We observe an increase in RMSE for SNR values larger than 10 dB for Capon Beamformer [27]. The reason of this observation might be that it produces spatial spectrum with sharper peaks for larger SNR values even though angle grid is too coarse to capture these sharp peaks accurately.

Figures 4.11 and 4.12 depict the difference between RMSE obtained by the proposed

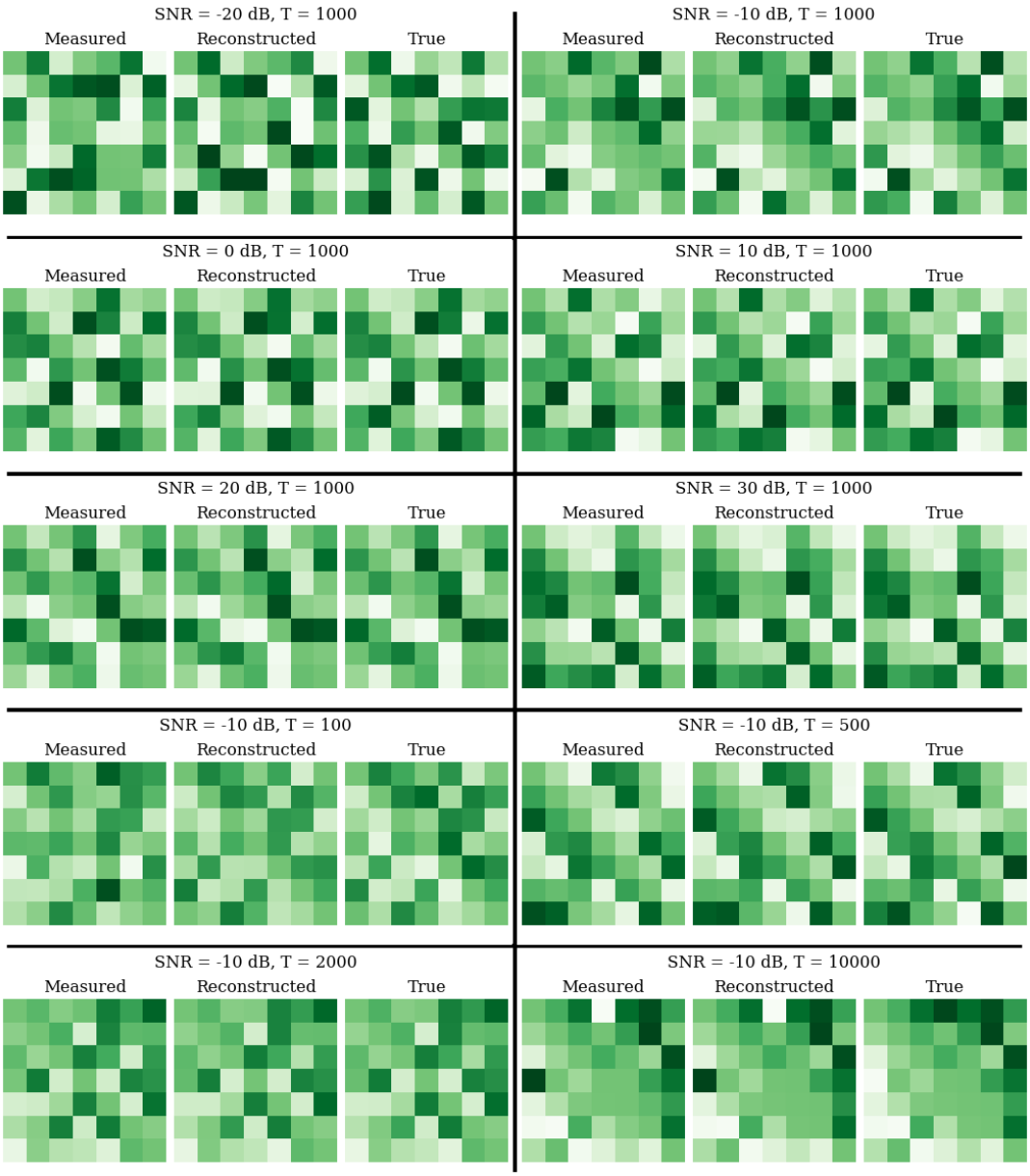

Figure 4.5: The effect of covariance reconstruction network on the covariance matrices.

method and the lowest RMSE among other on-grid methods. It can be observed that the proposed method provide significant improvement for low SNR levels and small snapshot numbers while having similar performance with the best performing methods for high SNR and snapshot regions.

The proposed method is also compared with other methods in terms of processing time. Table 4.4 contains the average processing time of each method which is calcu-

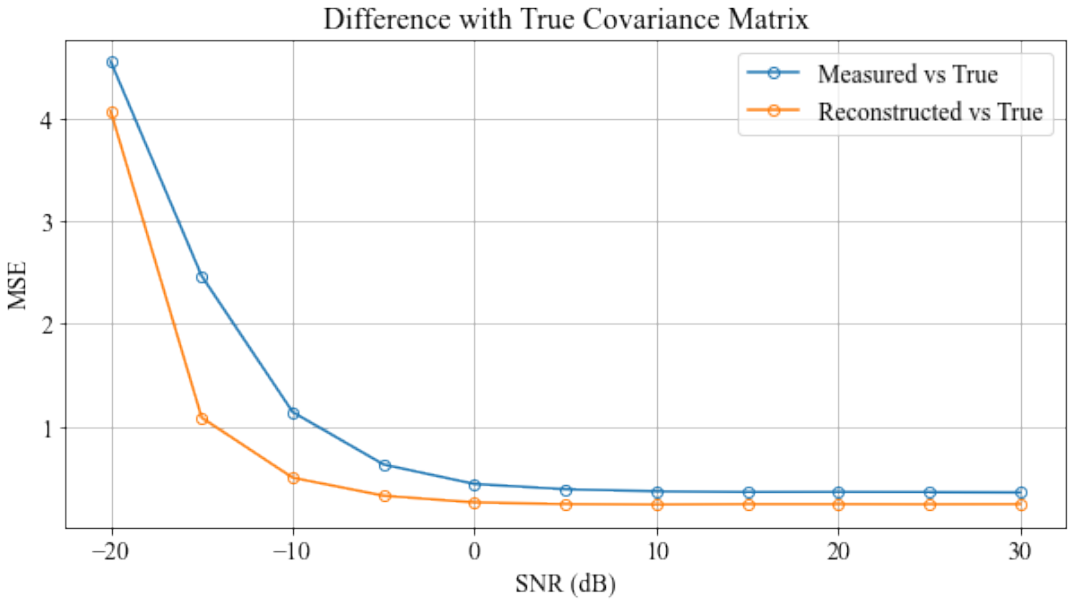

Figure 4.6: Difference of measured and reconstructed covariance matrices with true version for different SNR levels.

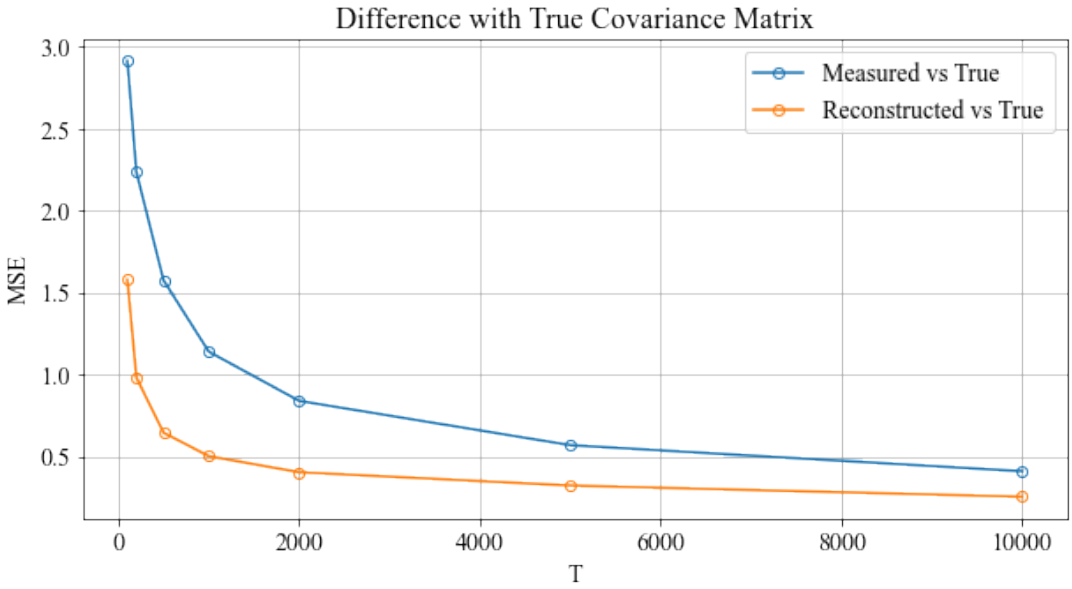

Figure 4.7: Difference of measured and reconstructed covariance matrices with true version for different snapshot numbers.

lated using 1000 Monte Carlo runs on Intel Xeon CPU. The lowest processing time is achieved by Capon Beamformer [27] which is followed by MUSIC [22] and the proposed method. Therefore, the proposed method provides performance improvement

Figure 4.8: Loss curves for direction of arrival estimation network.

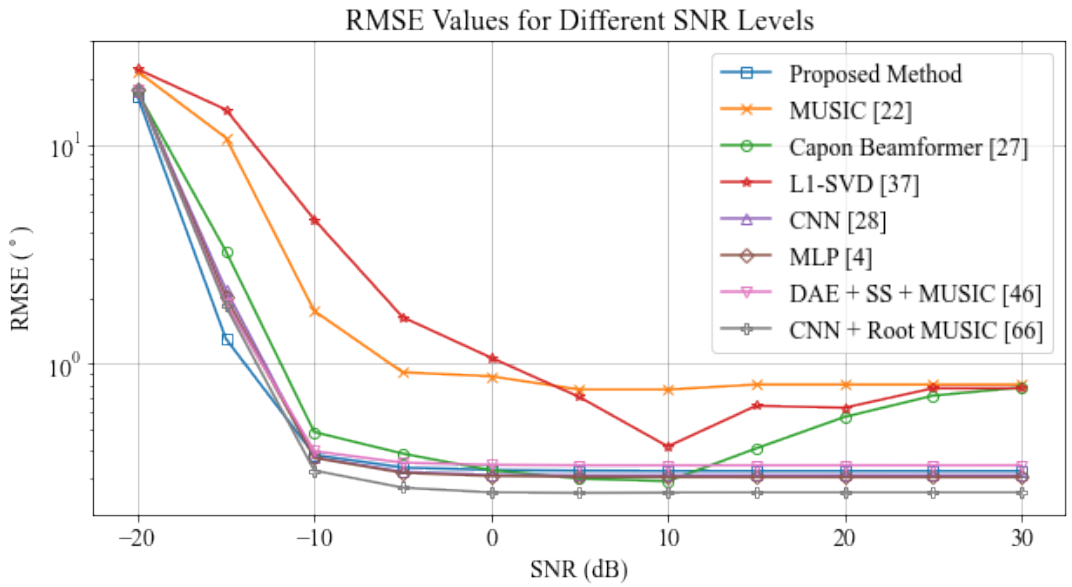

Figure 4.9: RMSE comparison for different SNR levels.

without too much computational burden. It has the lowest processing time among data-driven and hybrid methods.

The errors made by each method are illustrated in Appendix B for different SNR and snapshot levels.

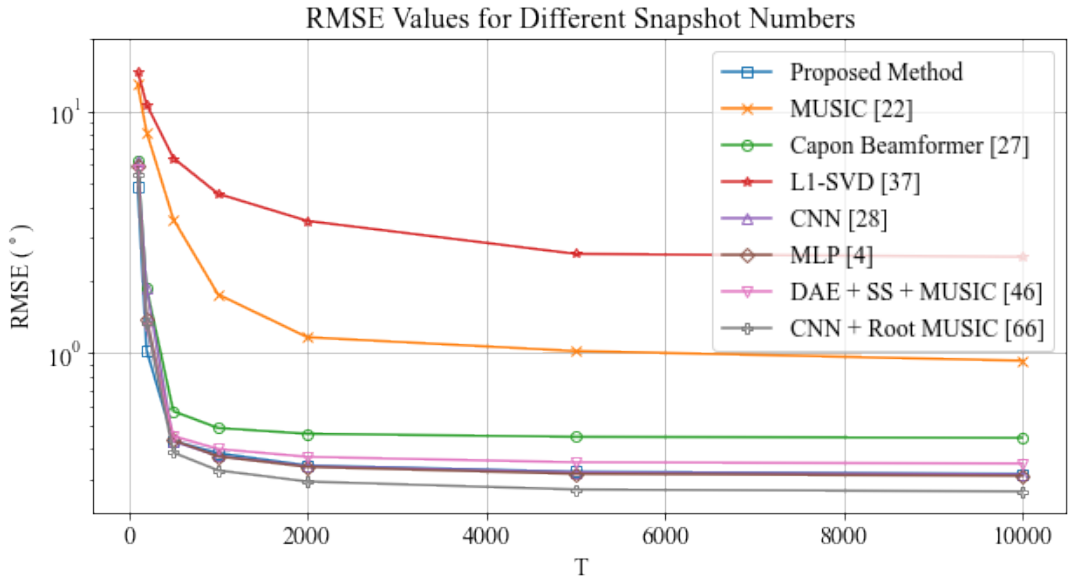

Figure 4.10: RMSE comparison for different snapshot numbers.

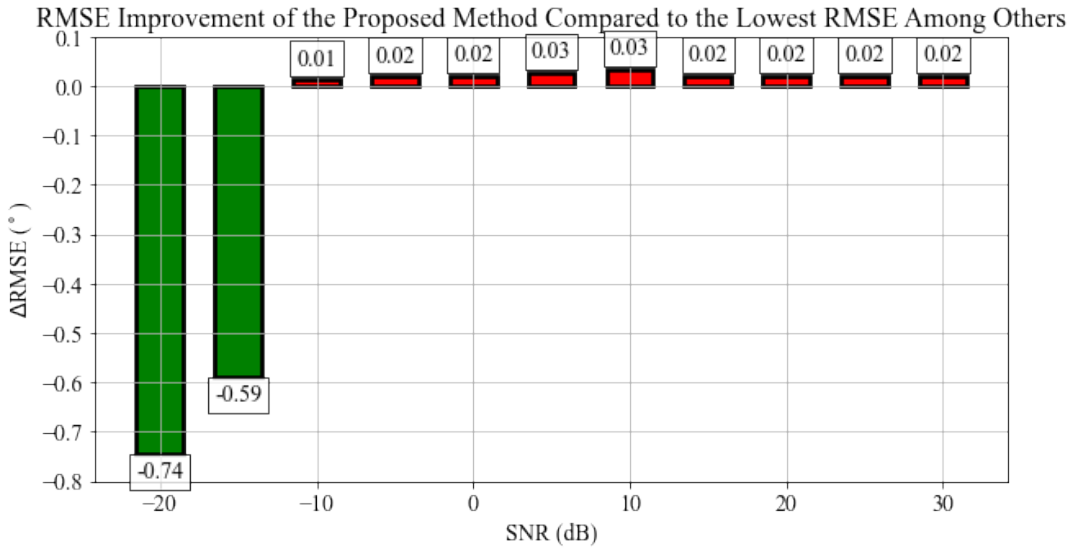

Figure 4.11: RMSE improvement by the proposed method for different SNR levels.

The proposed method is also evaluated in terms of its spatial spectrum recovery performance. Since the last layer of direction of arrival estimation network has sigmoid activation function, each output node produces a probability of source existence in the corresponding grid. Therefore, the probability values produced by each output node can be considered as a spatial spectrum. The other data-driven methods also use sigmoid activation function at their output layer, therefore the same spectrum generation

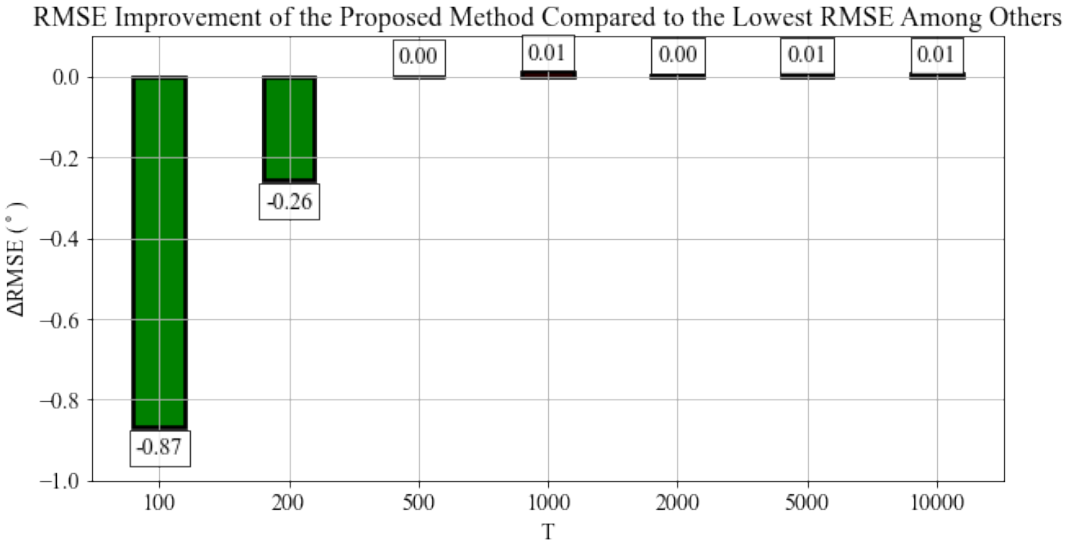

Figure 4.12: RMSE improvement by the proposed method for different snapshot numbers.

Table 4.4: Processing time for each method

Method	Processing Time (ms)
Proposed Method	14.803
MUSIC [22]	10.473
Capon Beamformer [27]	4.811
L1-SVD [37]	3070.515
CNN [28]	1455.647
MLP [4]	81.060
DAE + SS + MUSIC [46]	17.500
CNN + Root-MUSIC [66]	44.652

process is valid for them as well. Figure 4.13 shows the comparison of the spatial spectrums generated by each method for a randomly selected sample in a test set. In this graph, spectrum for only CNN + Root-MUSIC [66] is not given since Root-MUSIC utilizes polynomial root-finding instead of peak search from a spectrum [33]. From the graph, it can be observed that the proposed method has higher peaks for the true source angles compared to subspace-based methods, beamforming methods, and sparsity-inducing methods. However, the peaks for other data-driven methods are higher than the proposed method which describes a shortage of the proposed method compared to other data-driven methods.

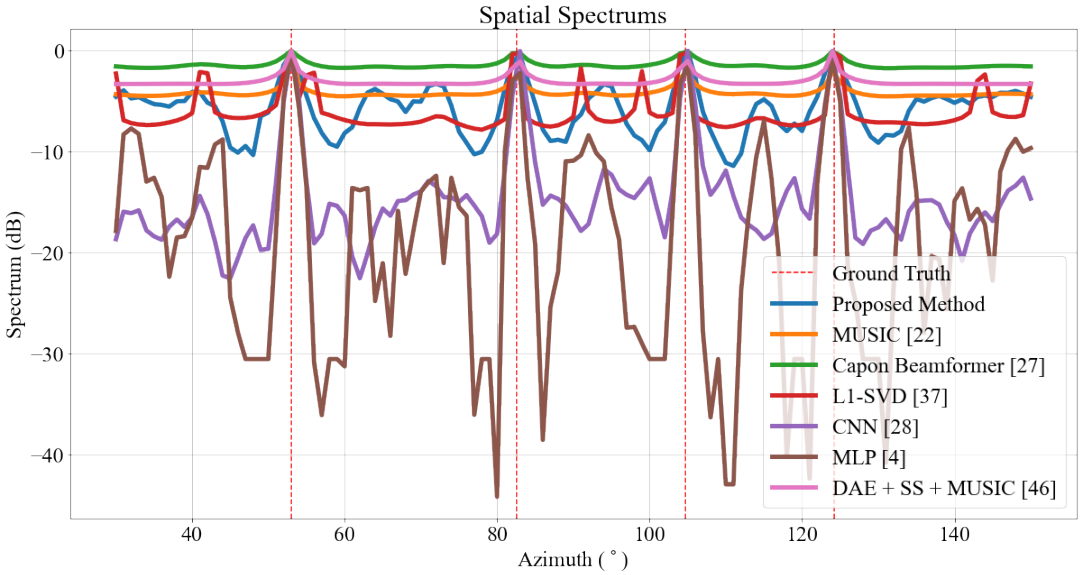

Figure 4.13: Spatial spectrum generated by each method.

4.2.2 Nested Array

The loss curves obtained for covariance reconstruction network are given in Figure 4.14. Training is performed for 80 epochs and the network parameters Ψ_{CRN} are set with the values obtained for the lowest validation loss.

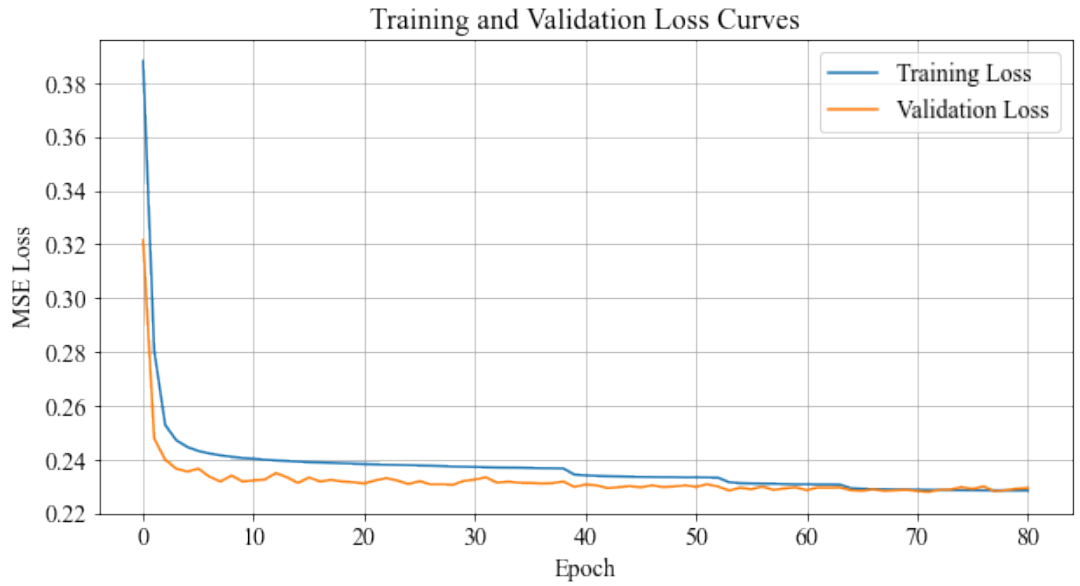

Figure 4.14: Loss curves for covariance reconstruction network.

Reconstruction performance of the trained network is illustrated by examples in Figure 4.15. As it is the case for minimum redundant array, reconstruction network enhances the measured covariance matrix such that it is more similar to true version especially in low SNR and small number of snapshots. Figures 4.16 and 4.17 show the difference between true covariance matrix and reconstructed and measured covariance matrices where reconstructed version has lower MSE compared to the measured one. It can also be observed that reconstruction process is still applicable for unseen SNR level and snapshot numbers which demonstrates the generalization ability of the trained network.

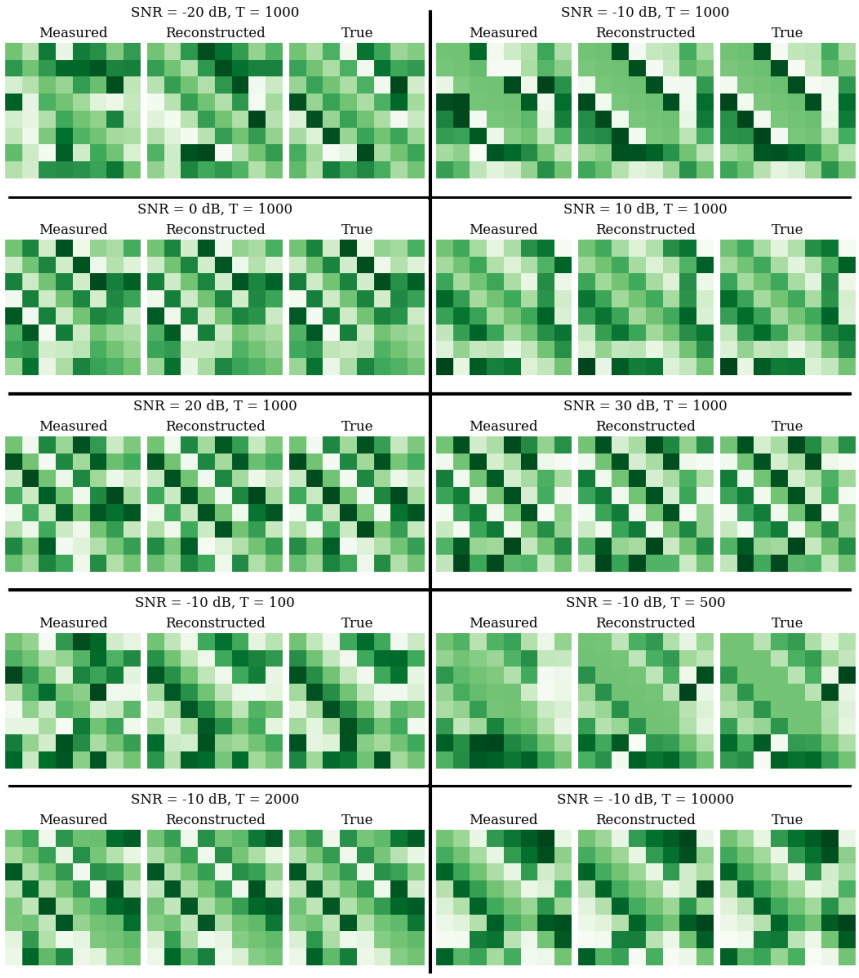

Figure 4.15: The effect of covariance reconstruction network on the covariance matrices.

For direction of arrival estimation network, the loss curves that are shown in Figure 4.18. are obtained during training process. Training is performed for 39 epochs and

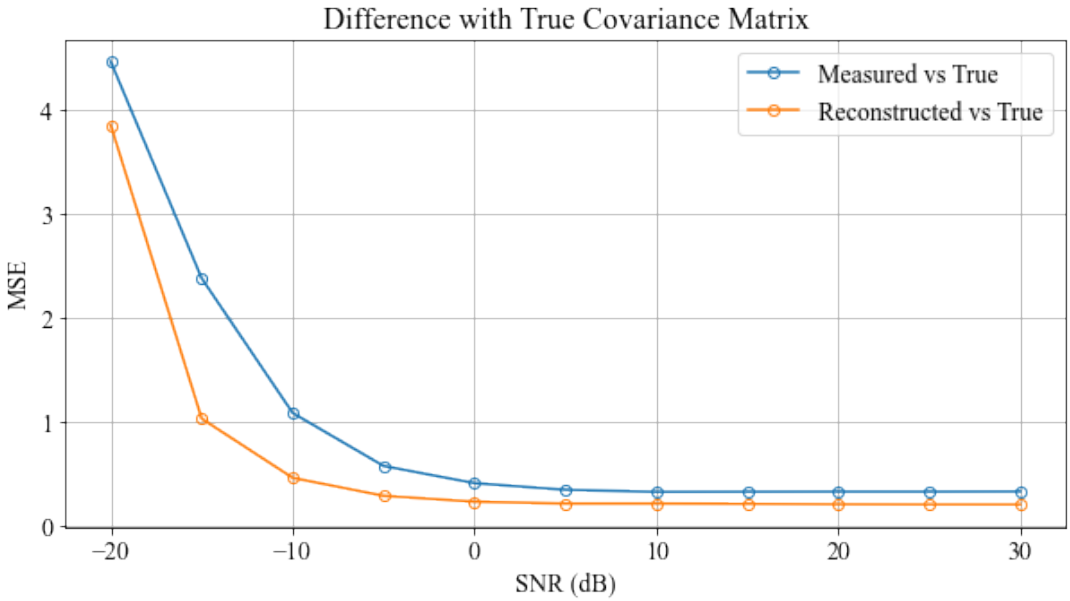

Figure 4.16: Difference of measured and reconstructed covariance matrices with true version for different SNR levels.

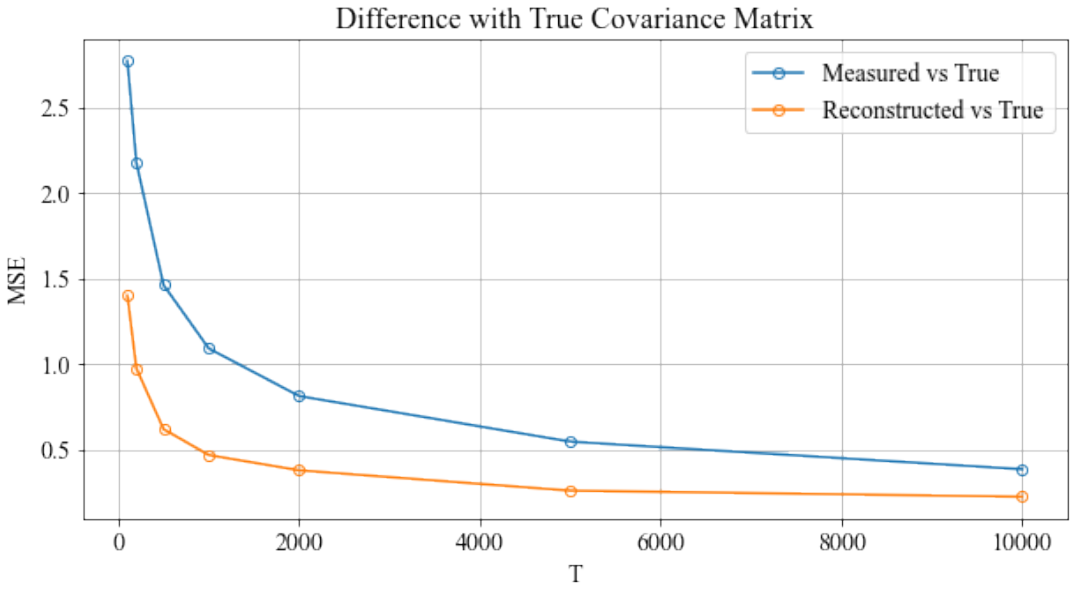

Figure 4.17: Difference of measured and reconstructed covariance matrices with true version for different snapshot numbers.

the network parameters Ψ_{DAEN} are set to the values obtained for lowest validation loss.

Figure 4.18: Loss curves for direction of arrival estimation network.

Performance evaluation is done on test sets for which RMSE values are illustrated in Figures 4.19 and 4.20 for different SNR levels and snapshot numbers respectively. Similar to the case in minimum redundant array, the best performance for low SNR and small number of snapshot is achieved by the proposed method. For high SNR and snapshot number, CNN + Root-MUSIC [66] has lower RMSE due to its gridless estimates. Data-driven and hybrid methods which make on-grid estimations have similar performance for this region where their RMSE values lie in the interval of 0.05° of RMSE. Unlike the case for minimum redundant array, the performance of MUSIC [22] is better for nested array. However, relatively poor performance of L1-SVD [37] is applicable for nested array as well.

Figures 4.21 and 4.22 shows the difference between RMSE obtained by the proposed method and the lowest RMSE among other on-grid methods. Similar to the case in minimum redundant array, the proposed method provide performance improvement especially in low SNR and snapshot regime while having similar RMSE with the best performing methods for higher SNR and snapshot levels.

The processing times for this part is not given for brevity since the same algorithms are applied.

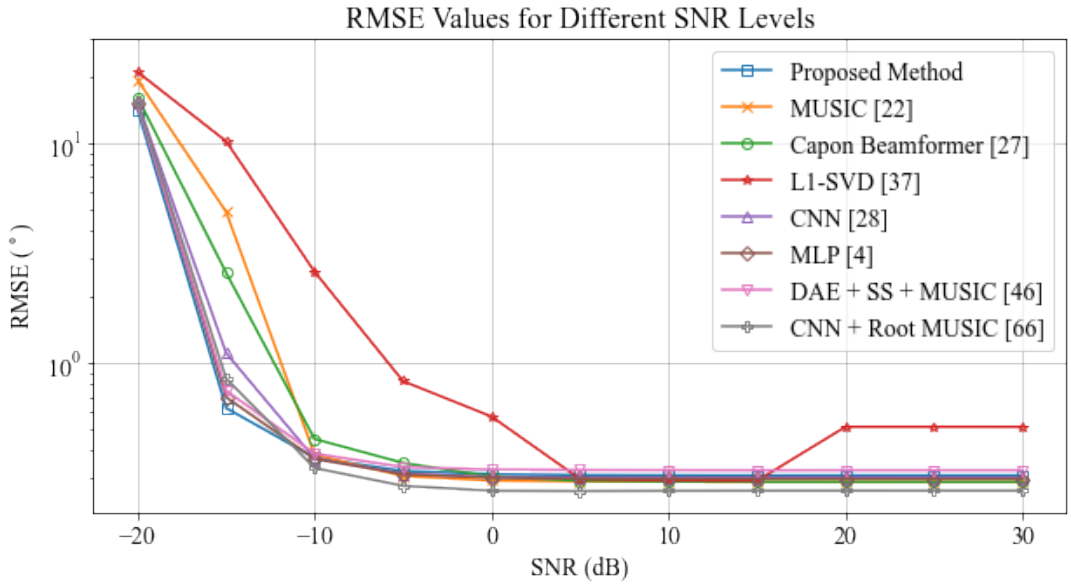

Figure 4.19: RMSE comparison for different SNR levels.

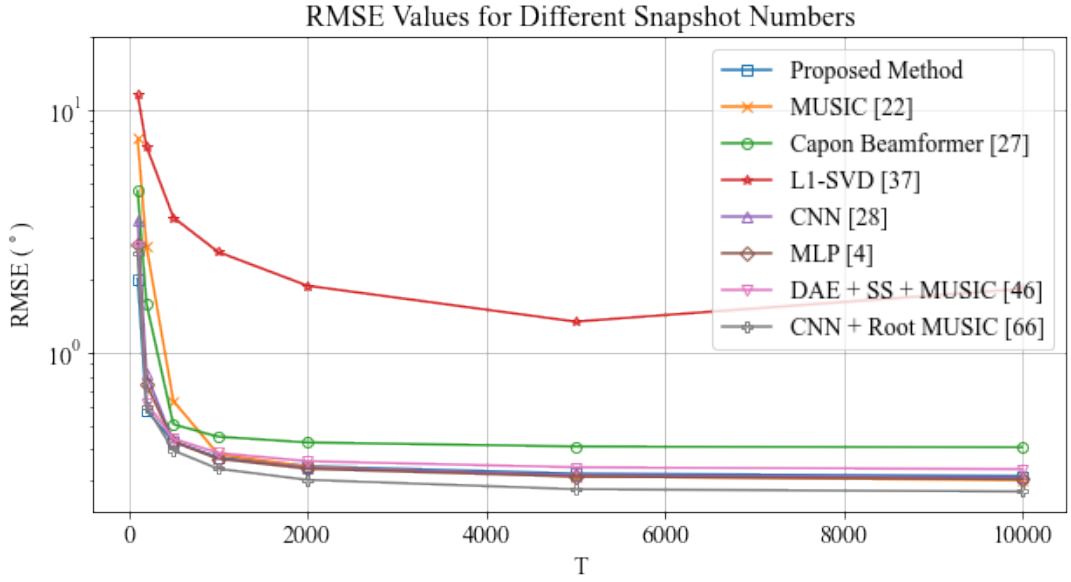

Figure 4.20: RMSE comparison for different snapshot numbers.

In Appendix C, the errors made by each method are illustrated for different SNR and snapshot levels.

Figure 4.23 shows the comparison of the spatial spectrums generated by each method for a randomly selected sample in a test set. As in the case of minimum redundant array, the proposed method has higher peaks for the true source angles compared to

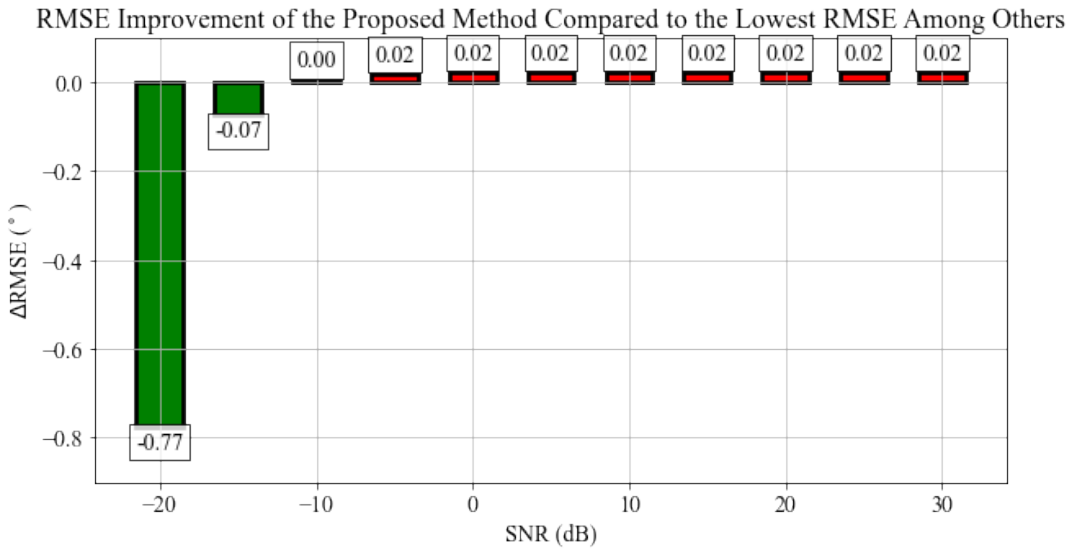

Figure 4.21: RMSE improvement by the proposed method for different SNR levels.

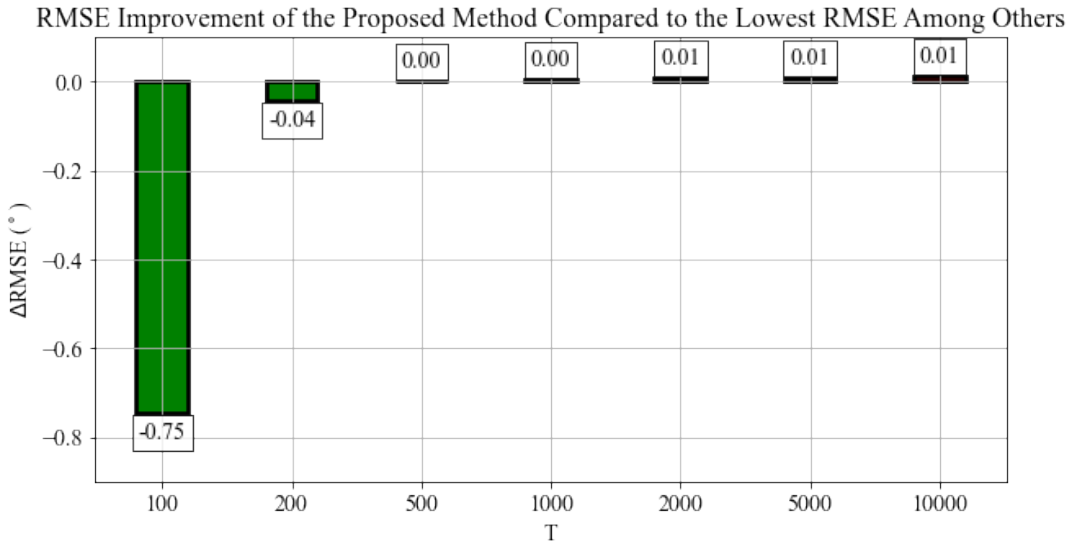

Figure 4.22: RMSE improvement by the proposed method for different snapshot numbers.

conventional methods. However, the peaks for other data-driven methods are slightly higher than the proposed method. The difference between the peaks of the proposed method and other data-driven methods is lower compared to the case for minimum redundant array.

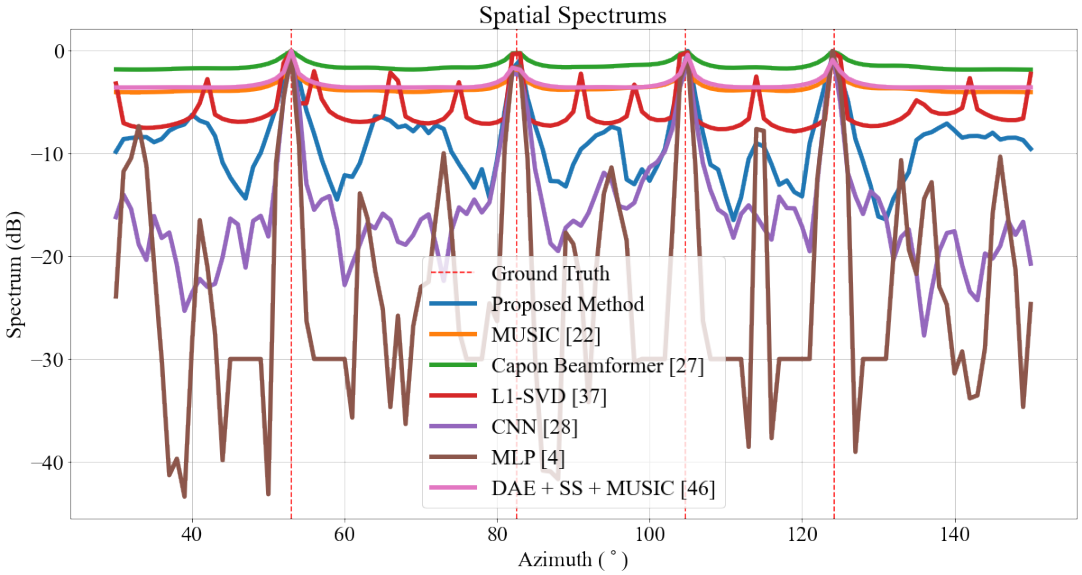

Figure 4.23: Spatial spectrum generated by each method.

4.2.3 Coprime Array

During training of covariance reconstruction network, the loss curves given in Figure 4.24 are obtained. The network is trained for 90 epochs and parameters $\mathbf{\Psi}_{\text{CRN}}$ are set with the values obtained for the lowest validation loss.

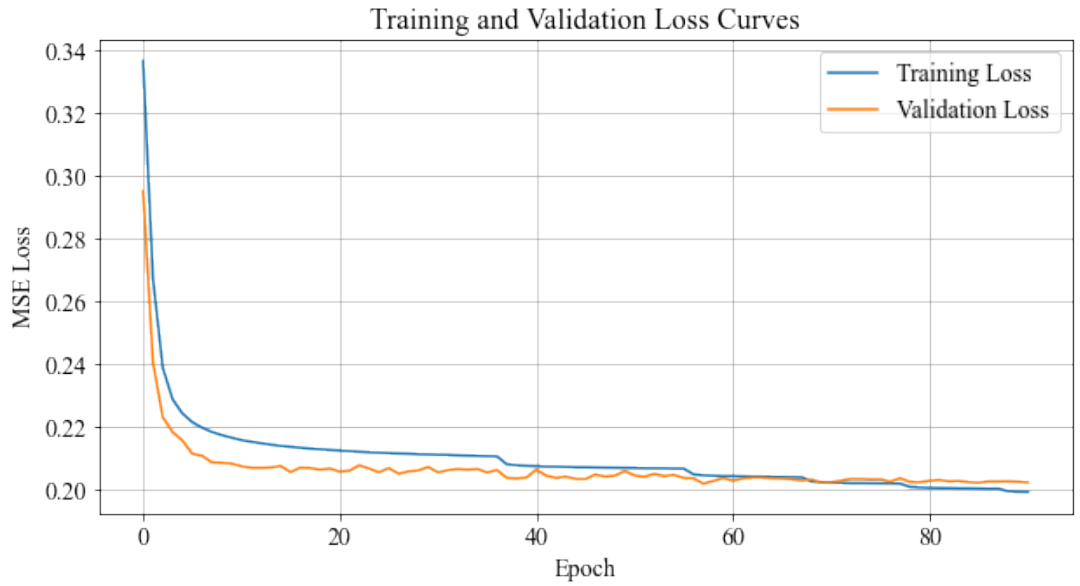

Figure 4.24: Loss curves for covariance reconstruction network.

Examples of the reconstructions applied by the trained network are illustrated in Figure 4.25. Similar to the cases for minimum redundant array and nested array, covariance reconstruction network presents denoising effect on the measured covariance matrix. The difference between reconstructed and measured covariance matrices with true covariance matrix is shown in Figures 4.26 and 4.27 for different SNR levels and snapshot numbers. It can be observed that the reconstructed version has lower MSE compared to measured one. Further, it doesn't cause any distortion for unseen SNR levels and snapshot numbers which indicates the generalization ability of the network.

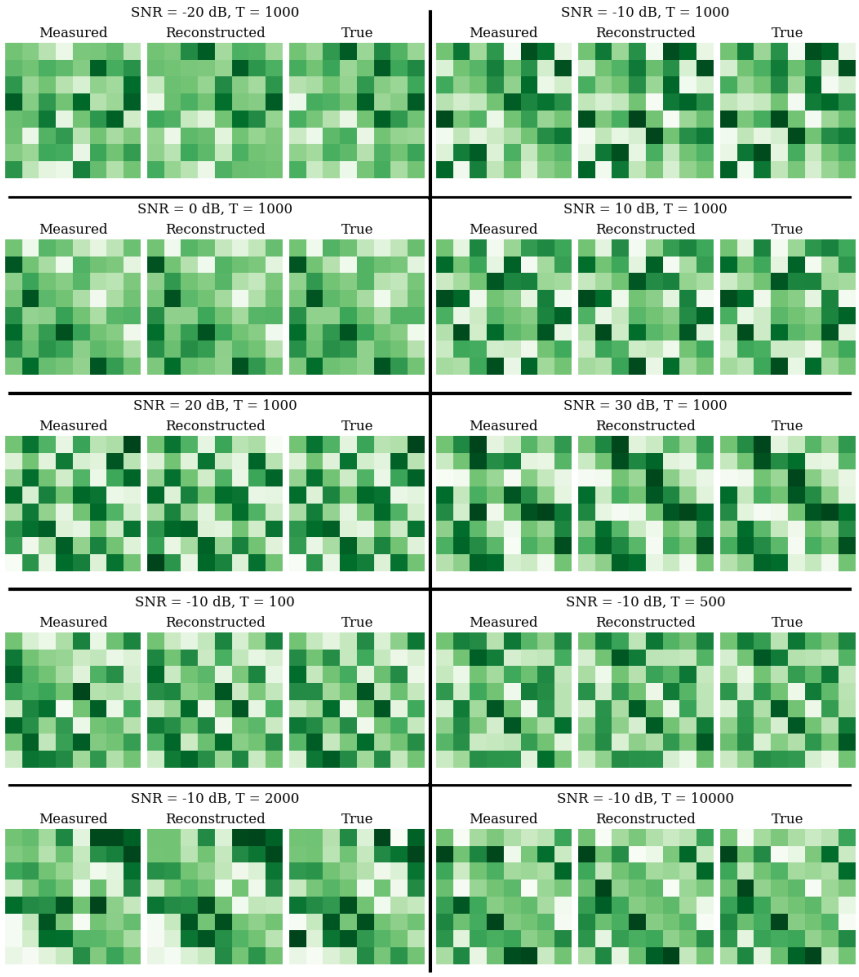

Figure 4.25: The effect of covariance reconstruction network on the covariance matrices.

Figure 4.28 shows the loss curves for direction of arrival estimation network obtained during training process. Training lasted 74 epochs and the network parameters Ψ_{DAEN} are set to the values obtained for lowest validation loss.

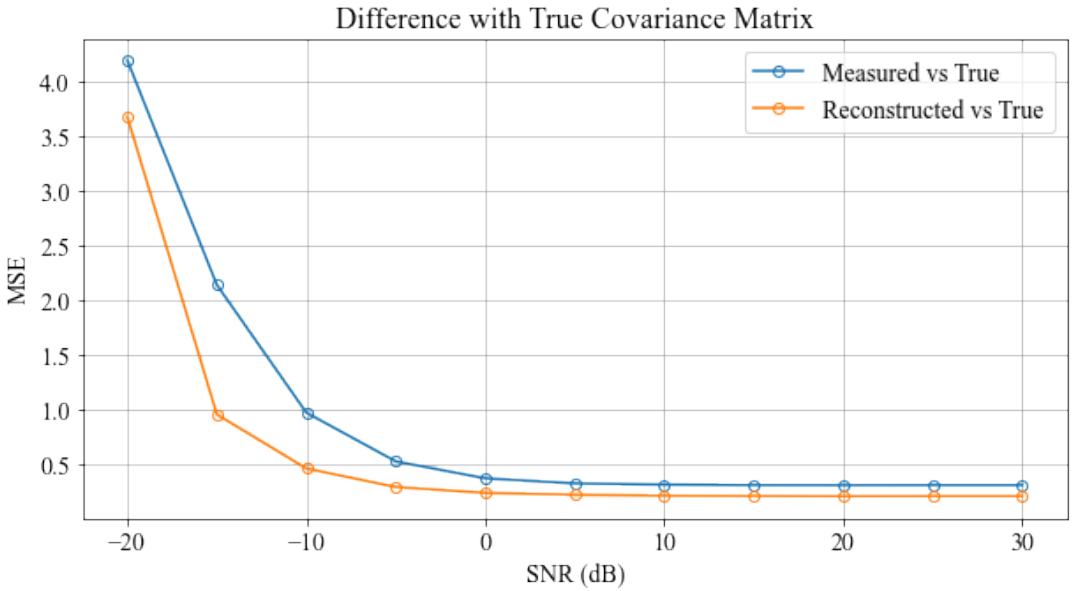

Figure 4.26: Difference of measured and reconstructed covariance matrices with true version for different SNR levels.

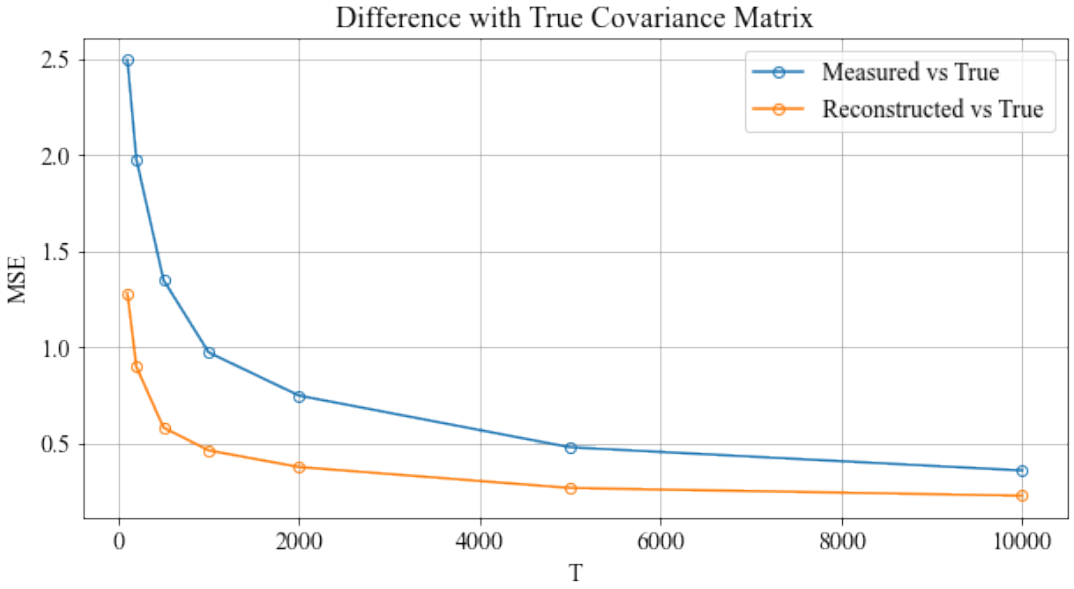

Figure 4.27: Difference of measured and reconstructed covariance matrices with true version for different snapshot numbers.

For performing the comparisons with other methods, trained networks are applied on test sets. Figures 4.29 and 4.30 show RMSE values for different SNR levels and snapshot numbers. As in the case of minimum redundant array and nested array, the

Figure 4.28: Loss curves for direction of arrival estimation network.

proposed method leads to the lowest RMSE for low SNR levels and snapshot numbers in the coprime array as well. This indicates that the proposed method is more robust against noise and small number of sensor measurements independent of the configuration and type of sparse array. For high SNR levels and snapshot numbers, the best performance is achieved by CNN + Root-MUSIC [66] which differs from the proposed and other methods in that it is a gridless method. RMSE values for the proposed method and other data-driven methods lie in 0.05° for this region. DAE + SS + MUSIC [46] shows the worst performance for coprime array which is not the case for minimum redundant array and nested array. Furthermore, the conventional methods like MUSIC [22] and Capon Beamformer [27] results in higher RMSE compared to data-driven methods which is observed for other sparse array types as well. This demonstrates the effectiveness of learning the nonlinear relationships in the sensor measurements over utilizing only signal model for direction of arrival estimation. The proposed method improves the performance of data-driven methods especially in low SNR and snapshot numbers without considerable performance loss in high SNR and snapshot numbers.

Figures 4.31 and 4.32 depict the difference between RMSE obtained by the proposed method and the lowest RMSE among other on-grid methods. Similar to the case

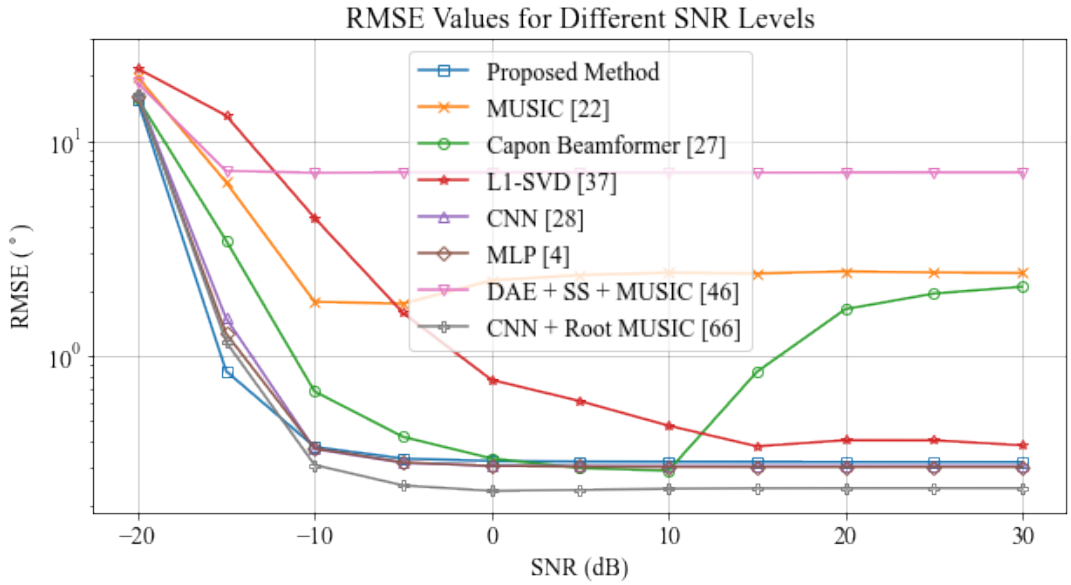

Figure 4.29: RMSE comparison for different SNR levels.

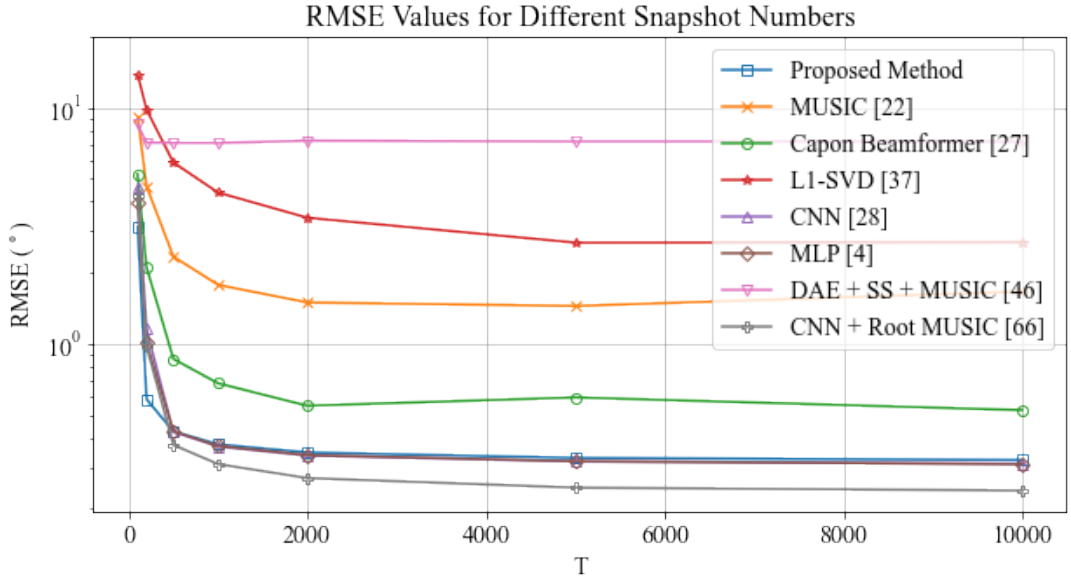

Figure 4.30: RMSE comparison for different snapshot numbers.

in minimum redundant and nested array, the proposed method improves the performance in low SNR and snapshot regime while having similar error rate with the best performing methods for higher SNR and snapshot levels.

The processing times are not given for brevity since there is no change in the methods that are applied.

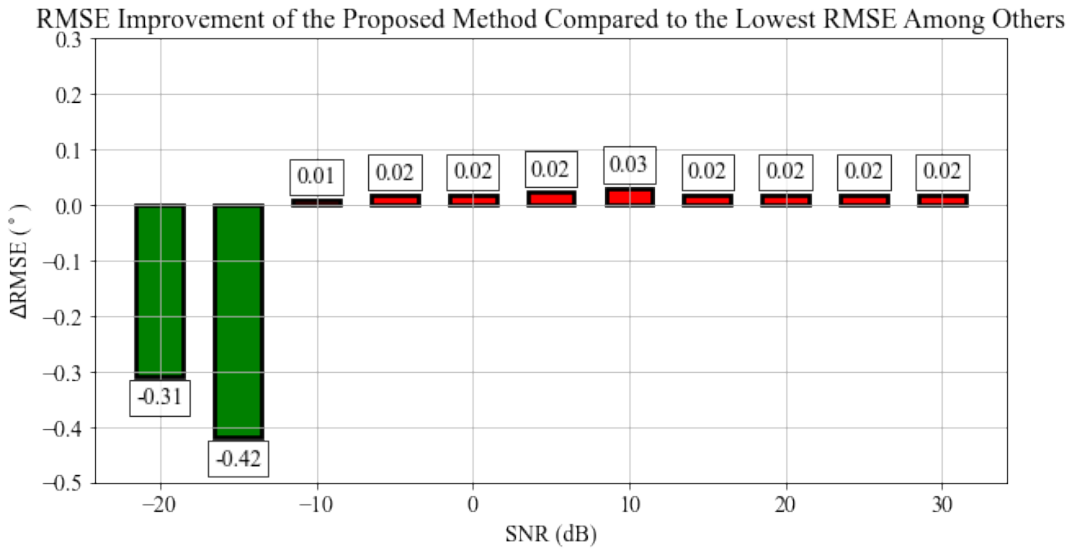

Figure 4.31: RMSE improvement by the proposed method for different SNR levels.

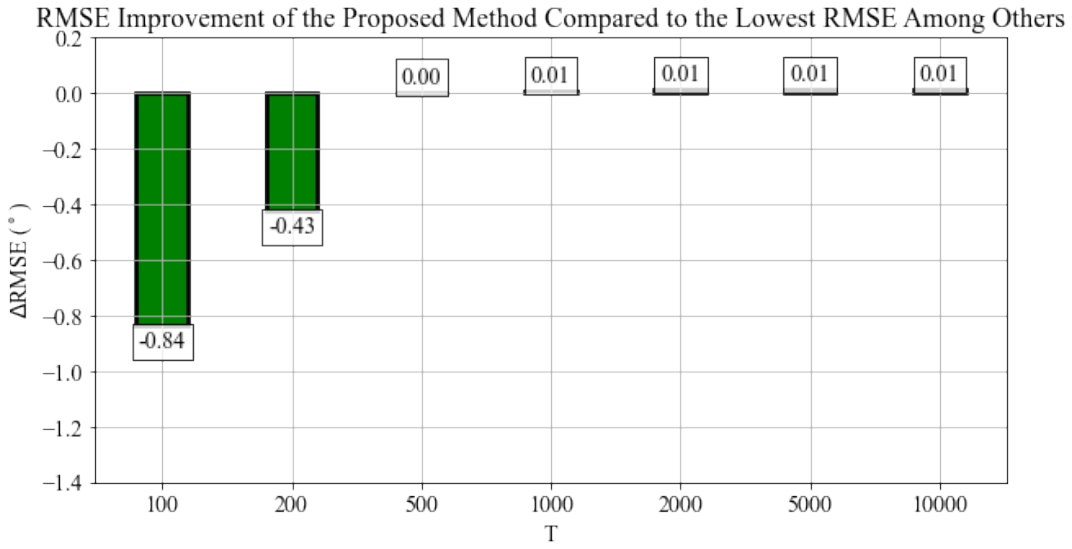

Figure 4.32: RMSE improvement by the proposed method for different snapshot numbers.

Appendix D shows the errors made by each method for different SNR and snapshot levels.

For a randomly selected sample from test sets, spatial spectrums are generated for each method and illustrated in Figure 4.33. Similar to the case in minimum redundant array and nested array, the proposed method leads to higher peaks for true source

angles compared to conventional methods, however, other data-driven methods have higher peaks than the proposed one which was the case for other sparse array types as well.

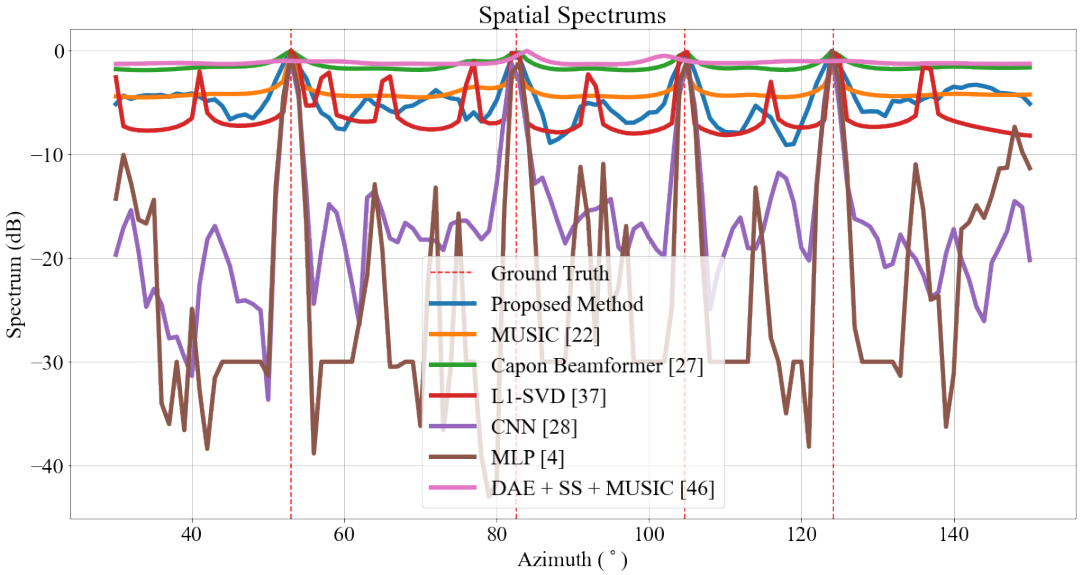

Figure 4.33: Spatial spectrum generated by each method.

CHAPTER 5

PERFORMANCE ANALYSIS: ROBUSTNESS TO SENSOR MALFUNCTIONS

In this chapter, the performance of transformer-based DOA estimation method is evaluated for the cases of sensor malfunctions. Similar to Chapter 4, sparse arrays are the main focus and the effects of possible failures in these arrays are to be investigated. For this purpose, a sparse array configuration is selected which is illustrated with its difference coarray in Figure 5.1.

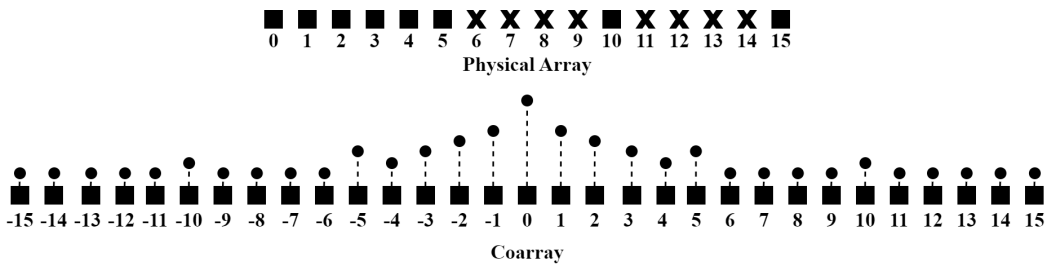

Figure 5.1: Intact sparse array configuration (top) and its coarray (bottom). Rectangle represents presence of the element while cross indicates its absence. The number of dashes in the dashed lines represent the weights for each lag.

As described in Section 2.5, the scope of this thesis is restricted with complete failures at the known sensors. Therefore, it is assumed that there is no source signal received by the faulty sensor(s) and other noise-related signals observed in these sensors are ignored. Hence, the experiments are done by only utilizing the signals from fully-functional sensors.

The performance of the proposed method is compared with the following methods:

1. MUSIC [22]

2. Capon Beamformer [27]

3. L1-SVD [37]

4. SVT + MUSIC [79]

5. MLP + MUSIC [88]

6. OMP [86]

MUSIC [22], Capon Beamformer [27] and L1-SVD [37] are the representatives of subspace-based methods, beamforming methods and sparsity-inducing methods respectively as in Chapter 4. SVT + MUSIC [79] is an example of matrix completion methods proposed for sensor malfunctions. SVT is known for its ability to recover low-rank matrices from incomplete measurements and applying it with MUSIC [22] which is a super-resolution method provides a strong baseline to compare with. MLP + MUSIC [88] is used for representing the data-driven methods. MLP enables compensating the missing information by learning from data and since the proposed method is also data-driven, MLP + MUSIC [88] is a good candidate to use in the evaluation. As a representative of compressed sensing-based methods which are introduced for sensor malfunctions, OMP [86] is chosen since it is a well-known method in sparse signal recovery which can also be used for handling incomplete data.

Application of MUSIC [22], Capon Beamformer [27] and L1-SVD [37] are not dependent on sensor number/configuration in the array, so they can be used in case of sensor malfunctions as well. However, data-driven and hybrid methods in Chapter 4 couldn't be applied in this chapter since they can only be used on the array configuration with which they are trained. CNN [28] and CNN + Root-MUSIC [66] take the sample covariance matrix as input whose dimension is dependent on the number of sensors. Therefore, the trained network cannot be used in case of sensor malfunctions since the input dimension changes in such a case. Similarly, MLP [4] and DAE + Spatial Smoothing + MUSIC [46] takes the vectorized version of the sample covariance matrix and hence, they cannot be applied when the number of sensors changes. There arises a need of retraining for these methods when sensor failure occurs while

the proposed method does not have such a requirement owing to its input formulation and positional encoding ability.

For performing the experiments, sensor malfunctions are simulated on the selected array and performance comparisons are made with the mentioned methods. More details about these experiments are given in Section 5.1.

5.1 Simulation Details

Sensor failures can be encountered in any of the sensor(s) of the array, therefore it is essential for data-driven methods to be able to make estimations for arrays which possess the subset of the intact array sensors. For this purpose, a training set which contains samples from the constructed sparse array and its subarrays is generated. The details of training set is given in Table 5.1. For each sample, randomly selected sensors are considered as faulty and the received signals are generated for the other intact sensors. The number of faulty sensors is limited considering the number of sources since subspace-based methods cannot function for undetermined case. Other parameters are selected for the same reasons that are explained in Section 4.1.

Table 5.1: The specifications of training set

The number of samples (D)	2000000
The number of sources at each sample (N)	4
The number of sensors at each sample (M)	4 - 8*
Source separations	$>10^\circ$
Source direction (azimuth, θ)	30° - 150°*
Source direction (elevation, ϕ)	0°
The number of snapshots (T)	100 - 1000*
SNR	-20 dB - 0 dB*
Source signal type	Complex random Gaussian
Noise signal type	Complex random Gaussian
Sampling rate	10 kHz

* Uniformly selected between the given interval

As in Chapter 4, the proposed transformer-based network is trained in two steps.

In both steps, the network is implemented using PyTorch framework and training is performed on Tesla T4 GPU. Firstly, covariance reconstruction network is trained for learning the mapping between measured covariance coarray and the corresponding true versions. Both measured and true covariance matrices have the size of fully-functional sensor number in the array. Hold-out cross-validation is applied where the ratio of the validation set is 0.1. Supervised learning is applied offline where mean square error between the reconstructed and true covariance coarray is minimized over validation set. The optimization process in which the trainable parameters $\mathbf{\Psi}_{\text{CRN}}$ of the network are updated can be formulized as

$$\mathbf{\Psi}^{*}_{\text{CRN}} = \underset{\mathbf{\Psi}_{\text{CRN}}}{\arg\min} \frac{1}{D} \sum_{d=1}^{D} \|\boldsymbol{y}_d - \hat{\boldsymbol{y}}_d\|^2 \tag{5.1}$$

where $\boldsymbol{y}_d$ represents the ground truth and $\hat{\boldsymbol{y}}_d$ indicates the network output for d^{th} sample in the validation set. Other training hyperparameters for the covariance reconstruction network are given in Table 5.2 which are the same as that are used in Chapter 4. These hyperparameters are selected by hyperparameter tuning process.

Table 5.2: Training hyperparameters

Optimizer	Adam [111] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$
Learning rate	Initially 0.001
Learning rate decay	0.7 once validation loss reaches plateau with a patience of 10 epochs
Batch size	512

In the second step, direction of arrival estimation network is trained for learning the relationship between the reconstructed covariance coarray and DOA angle. As in Chapter 4, the problem is formulated as multi-label classification task and output layer of the network is constructed to have $G = 121$ neurons such that 1° of resolution is achieved. As ground truths in training set, binary vectors are used for each sample where true source angles are labeled as 1 and others as 0. Supervised learning is applied offline where the parameters $\mathbf{\Psi}_{\text{DAEN}}$ of the network is optimized by minimizing the error

$$\boldsymbol{\Psi}^{*}_{\text{DAEN}} = \underset{\boldsymbol{\Psi}_{\text{DAEN}}}{\arg\min} \frac{1}{D} \sum_{d=1}^{D} L(\boldsymbol{z}_d, \hat{\boldsymbol{z}}_d) \tag{5.2}$$

where

$$L(\boldsymbol{z}_d, \hat{\boldsymbol{z}}_d) = -\frac{1}{G} \sum_{g=1}^{G} [\boldsymbol{z}_d(g) \log(\hat{\boldsymbol{z}}_d(g)) + (1 - \boldsymbol{z}_d(g)) \log(1 - \hat{\boldsymbol{z}}_d(g))] \tag{5.3}$$

is binary cross-entropy loss, $\boldsymbol{z}_d$ represents the ground truth and $\hat{\boldsymbol{z}}_d$ shows the network output for d^{th} sample in validation set. Other training hyperparameters are the same as that are used for covariance reconstruction network and they are summarized in Table 5.2.

For training data-driven method that is used in the comparison, the same training set is used for a fair comparison. Training process of this method is applied by following the procedures and using the hyperparameters that are provided in the corresponding article [88].

For evaluating the performance of the proposed method and compare with others, two different scenarios are built. In the first scenario, a test set is generated which contains samples from the intact array specified in Figure 5.1. In the second scenario, it is assumed that sensor malfunctions occur on randomly selected elements of the array. Test sets for these two scenarios are generated for different SNR levels and snapshot numbers whose details are summarized in Tables 5.3 and 5.4. Signal types, sampling rate and source elevation are the same as in training set.

As in Chapter 4, RMSE is used as the evaluation metric which can be formulized as

$$\text{RMSE} = \sqrt{\frac{1}{D} \sum_{d=1}^{D} \|\boldsymbol{\theta}_d - \hat{\boldsymbol{\theta}}_d\|^2} \tag{5.4}$$

where $\boldsymbol{\theta}_d$ is the ground truth DOA angles and $\hat{\boldsymbol{\theta}}_d$ is the estimated DOA angles.

Regularization parameter that is used for L1-SVD [37] is set to 0.1 during experiment.

Table 5.3: The specifications of test sets for intact array

Test set #	Number of samples (D)	Number of sensors (M)	Number of sources (N)	Source separations	Source direction (θ)	SNR	Number of snapshots (T)
1	10000	8	4	$>10°$	30° - 150°*	-20 dB	1000
2	10000	8	4	$>10°$	30° - 150°*	-15 dB	1000
3	10000	8	4	$>10°$	30° - 150°*	-10 dB	1000
4	10000	8	4	$>10°$	30° - 150°*	-5 dB	1000
5	10000	8	4	$>10°$	30° - 150°*	0 dB	1000
6	10000	8	4	$>10°$	30° - 150°*	5 dB	1000
7	10000	8	4	$>10°$	30° - 150°*	10 dB	1000
8	10000	8	4	$>10°$	30° - 150°*	15 dB	1000
9	10000	8	4	$>10°$	30° - 150°*	20 dB	1000
10	10000	8	4	$>10°$	30° - 150°*	25 dB	1000
11	10000	8	4	$>10°$	30° - 150°*	30 dB	1000
12	10000	8	4	$>10°$	30° - 150°*	-10 dB	100
13	10000	8	4	$>10°$	30° - 150°*	-10 dB	200
14	10000	8	4	$>10°$	30° - 150°*	-10 dB	500
15	10000	8	4	$>10°$	30° - 150°*	-10 dB	1000
16	10000	8	4	$>10°$	30° - 150°*	-10 dB	2000
17	10000	8	4	$>10°$	30° - 150°*	-10 dB	5000
18	10000	8	4	$>10°$	30° - 150°*	-10 dB	10000

* Uniformly selected between the given interval

5.2 Results and Discussion

The loss curves obtained during training of covariance reconstruction network are shown in Figure 5.2. Training is performed for 106 epochs and the network parameters $\mathbf{\Psi}_{\text{CRN}}$ are determined by considering the lowest validation loss.

Example reconstructions done by the trained network on test set are illustrated in Figure 5.3. As it can be observed, the network is able to operate on data collected from arrays with different number/configuration of sensors and apply denoising effect especially for low SNR levels. This effect is numerically more observable in Figures 5.4 and 5.5 where MSE of the difference between true covariance matrix and reconstructed and measured covariance matrices are plotted for different SNR and snapshot values.

Table 5.4: The specifications of test sets for faulty array

Test set #	Number of samples (D)	Number of intact sensors (M)	Number of sources (N)	Source separations	Source direction (θ)	SNR	Number of snapshots (T)
1	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	-20 dB	1000
2	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	-15 dB	1000
3	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	-10 dB	1000
4	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	-5 dB	1000
5	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	0 dB	1000
6	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	5 dB	1000
7	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	10 dB	1000
8	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	15 dB	1000
9	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	20 dB	1000
10	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	25 dB	1000
11	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	30 dB	1000
12	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	-10 dB	100
14	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	-10 dB	500
15	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	-10 dB	1000
16	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	-10 dB	2000
17	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	-10 dB	5000
18	10000	4 - 7*	4	$>10^\circ$	30° - 150°*	-10 dB	10000

* Uniformly selected between the given interval

During training of direction of arrival estimation network, the loss curves in Figure 5.6 are obtained. Training process lasted 129 epochs and the network parameters $\mathbf{\Psi}_{\text{DAEN}}$ are set with values obtained when the the validation loss is minimum.

After training process is completed, the proposed method and others are compared based on two scenarios for which the results are given in Sections 5.2.1 and 5.2.2.

5.2.1 Intact Array

In this section, test set containing samples from intact array is used make performance evaluation. Figures 5.7 and 5.8 provide the comparison of RMSE values for different SNR levels and snapshot numbers. Although it may seem that the proposed method doesn't provide performance improvement over other methods, it is

Figure 5.2: Loss curves for covariance reconstruction network.

important to consider the performance of individual methods for the whole SNR and snapshot regions. For instance, OMP [86], L1-SVD [37], MUSIC [22] and Capon Beamformer [27] cannot perform as successful as the proposed method for low SNR levels and snapshot numbers. The error difference between MLP + MUSIC [88] and the proposed method can be observed for high SNR and snapshot levels in which the proposed method has slightly better performance. SVT + MUSIC [79] shows mostly similar performance to the proposed method except snapshot number of 100 for which RMSE of the proposed method is 14.723° lower than it. Therefore, the proposed method provides a consistent performance for all SNR and snapshot regime while other methods may experience deterioration.

Figures 5.9 and 5.10 show the difference between RMSE obtained by the proposed method and the lowest RMSE among other methods. It can be observed that the proposed method provides similar performance to the best performing methods for all SNR levels and large number of snapshots while achieving a considerable performance improvement for small snapshot number.

A comparison is conducted in terms of processing times as well. Table 5.5 contains the average processing time of each method which is calculated over 1000 Monte Carlo runs on Intel Xeon CPU. The lowest processing time is achieved by MLP +

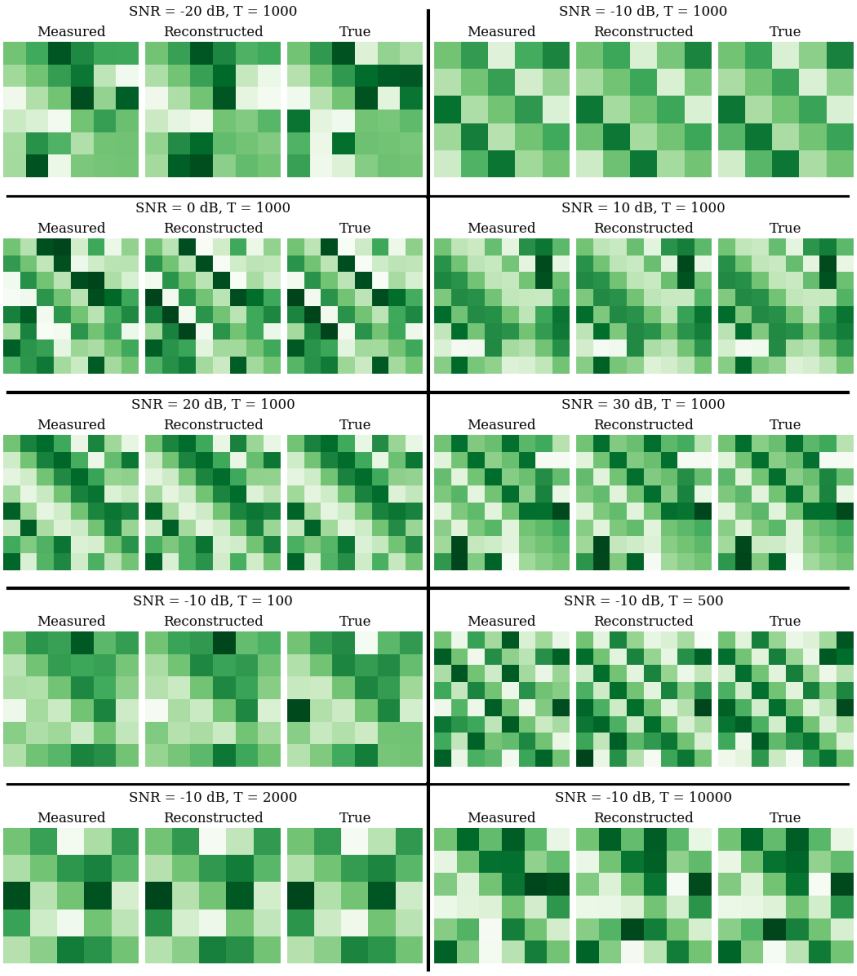

Figure 5.3: The effect of covariance reconstruction network on the covariance matrices.

MUSIC [88] which is followed by the proposed method. Therefore, the proposed method doesn't introduce too much computational burden. It has lower processing time than most of the other methods.

The errors made by each method are illustrated in Appendix E for different SNR and snapshot levels.

Figure 5.11 shows the spatial spectrums generated by each method for a a randomly selected sample in test set. It can be observed that the highest peaks for the source angles are obtained by the proposed method. In Chapter 4, there were other data-driven methods which have higher peaks compared to the proposed method however, these methods couldn't be applied for sensor malfunction case due to the limitation

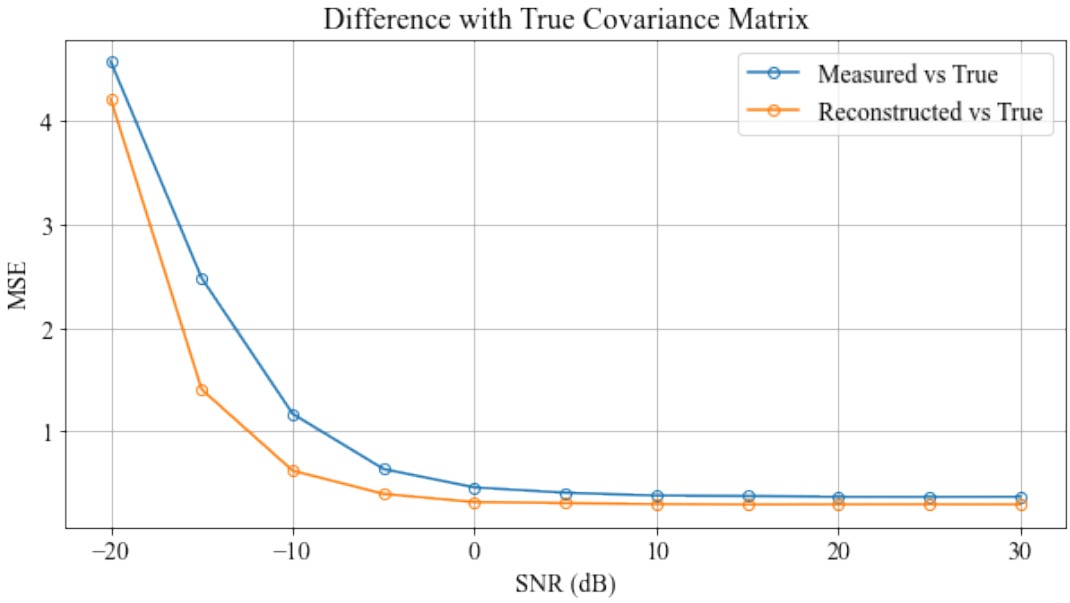

Figure 5.4: Difference of measured and reconstructed covariance matrices with true version for different SNR levels.

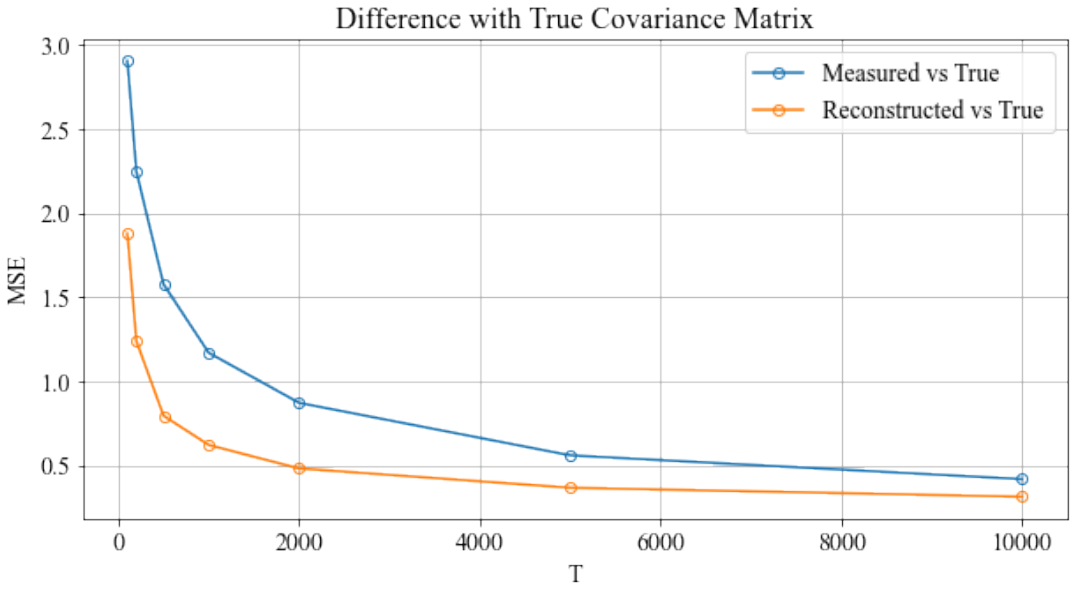

Figure 5.5: Difference of measured and reconstructed covariance matrices with true version for different snapshot numbers.

of their input formulation.

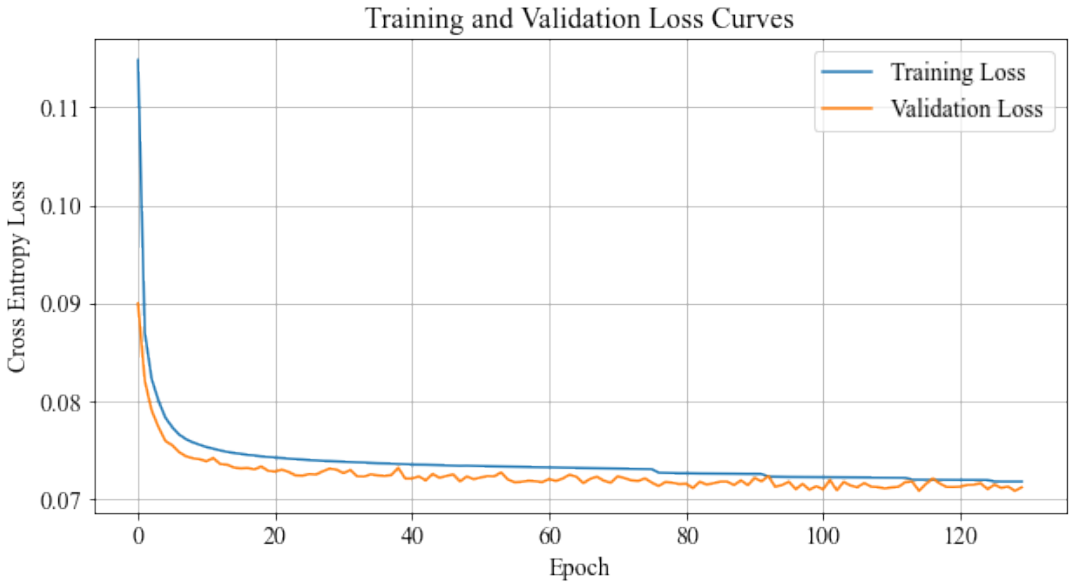

Figure 5.6: Loss curves for direction of arrival estimation network.

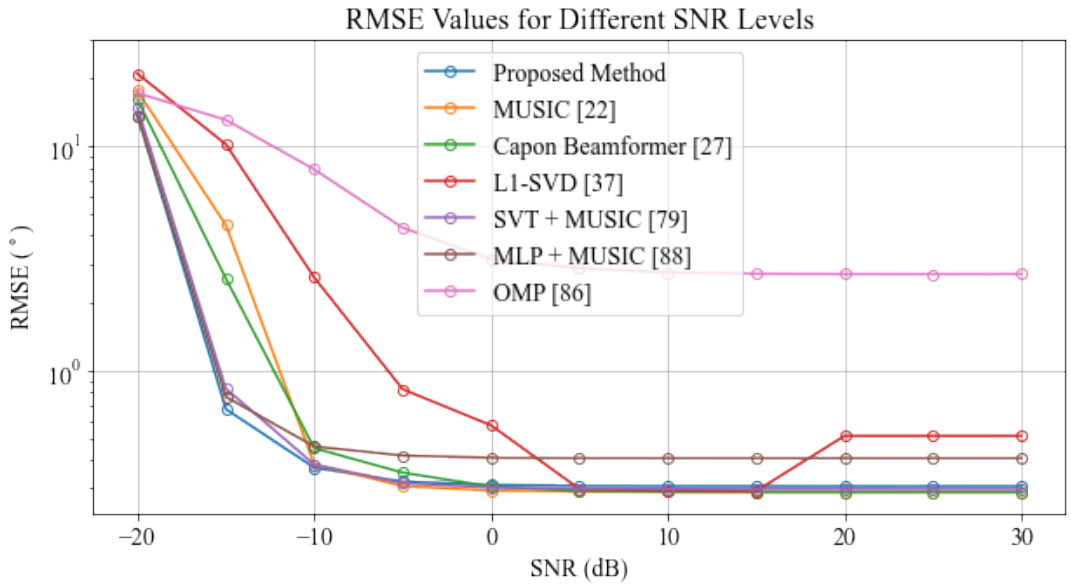

Figure 5.7: RMSE comparison for different SNR levels.

5.2.2 Faulty Array

This section covers performance evaluation for test set comprising of samples from faulty array. For different SNR levels and snapshot numbers, RMSE values obtained by each method are given in Figures 5.12 and 5.13. For SNR of -20 dB, the best

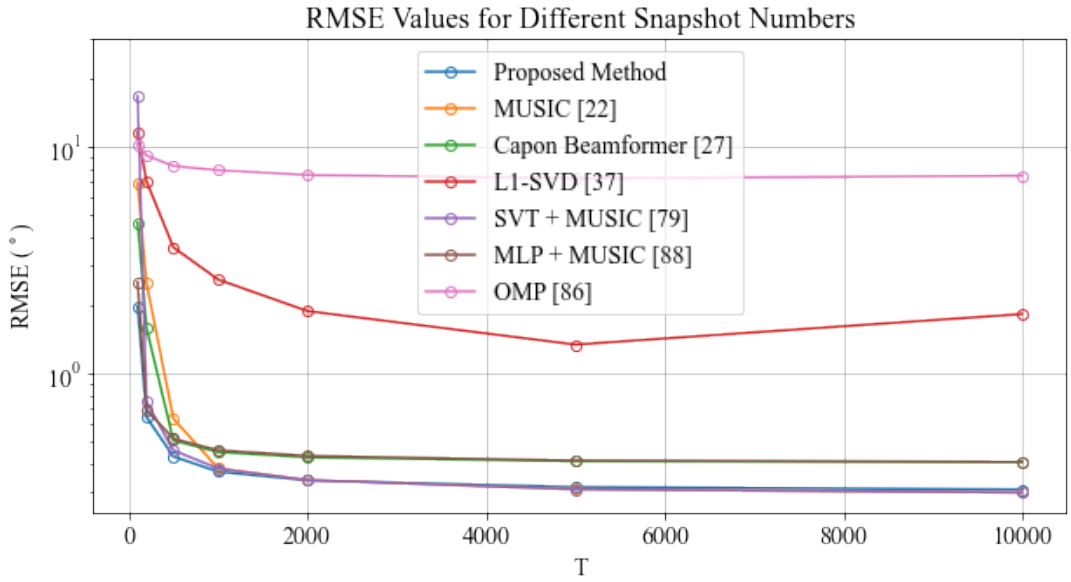

Figure 5.8: RMSE comparison for different snapshot numbers.

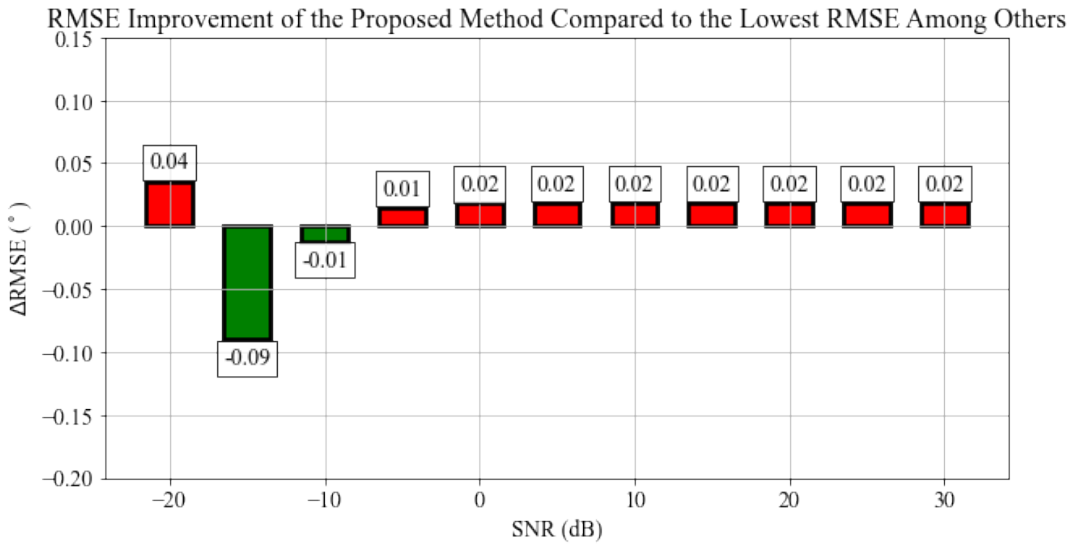

Figure 5.9: RMSE improvement by the proposed method for different SNR levels.

performance is achieved by MLP + MUSIC [88] which is followed by the proposed method. For SNR values between -15 and -5 dB, the proposed method leads to the lowest error. SVT + MUSIC [79] achieves the best performance for SNR levels higher than 0 dB. For these values, the proposed method has the second lowest RMSE except for between 10 and 30 dB where L1-SVD [37] has slightly lower error. Therefore, SVT + MUSIC [79] and L1-SVD [37] have better performance than the proposed

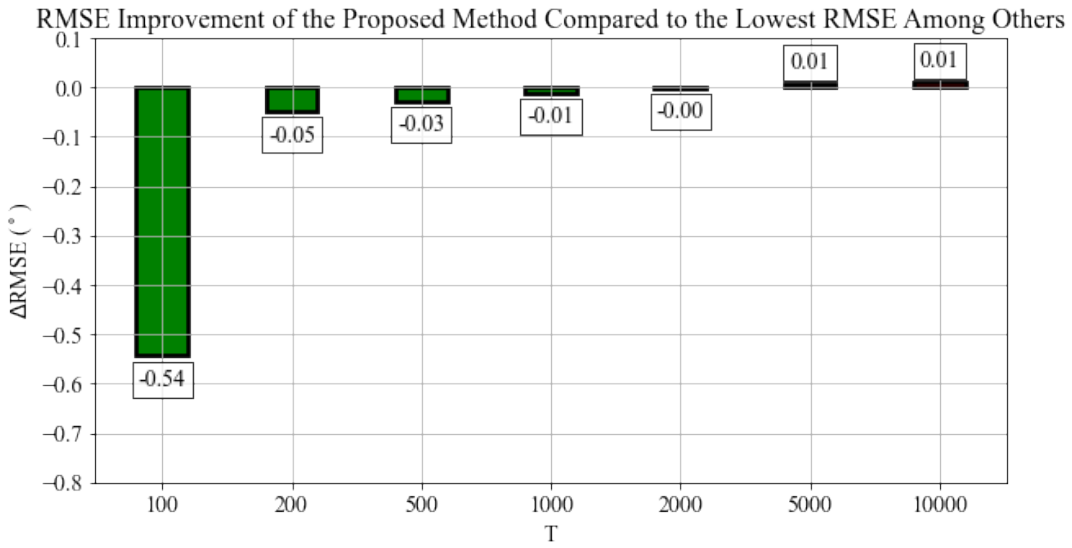

Figure 5.10: RMSE improvement by the proposed method for different snapshot numbers.

Table 5.5: Processing time for each method

Method	Processing Time (ms)
Proposed Method	4.082
MUSIC [22]	5.087
Capon Beamformer [27]	10.565
L1-SVD [37]	788.223
SVT + MUSIC [79]	642.672
MLP + MUSIC [88]	1.398
OMP [86]	4.814

method for high SNR region. However, it should be noted that SVT + MUSIC [79] and L1-SVD [37] are computationally much more demanding compared to the proposed method. Processing time of the proposed method is 157.44 times lower than SVT + MUSIC [79] and 193.09 times lower than L1-SVD [37] as given in Table 5.5. SVT + MUSIC [79] and L1-SVD [37] are methods which have iterative nature and this leads to massive processing times. Therefore, the proposed method is much more practical compared to them and it achieves the lowest RMSE among other practical methods which have feasible processing times. Furthermore, the proposed method outperforms other methods for all the snapshot numbers.

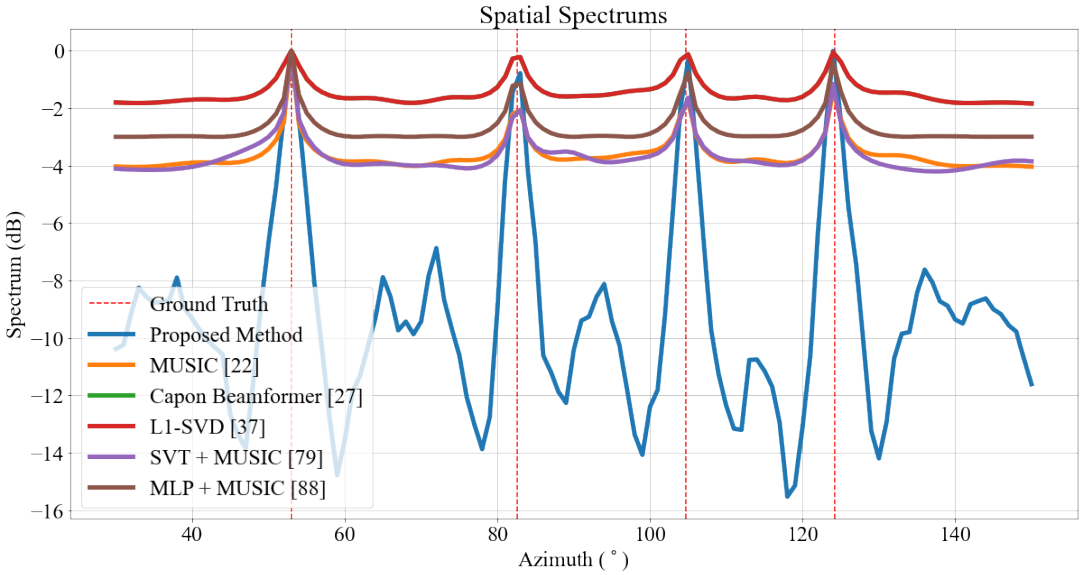

Figure 5.11: Spatial spectrum generated by each method.

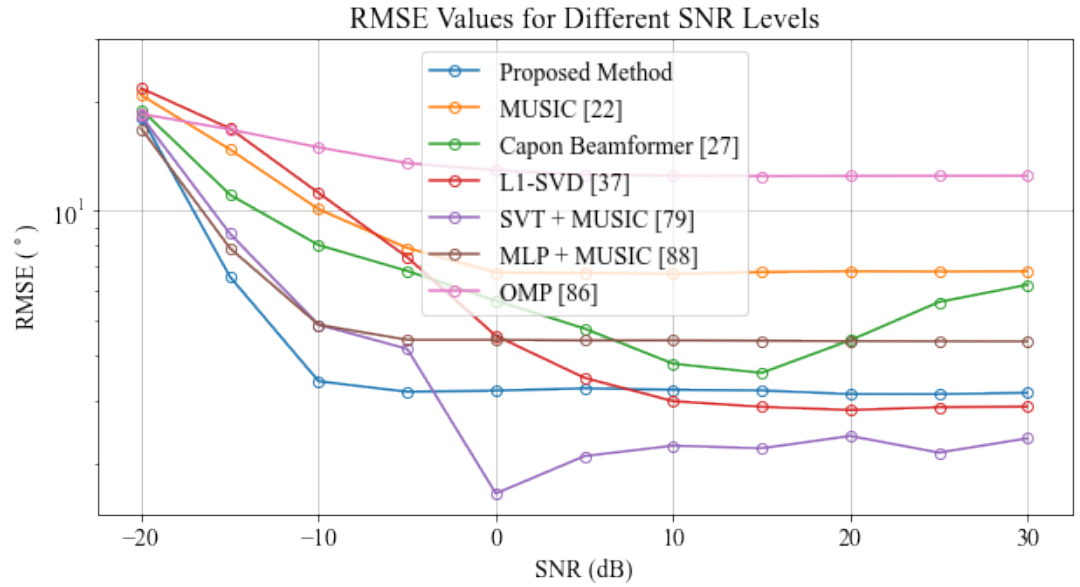

Figure 5.12: RMSE comparison for different SNR levels.

Figures 5.14 and 5.15 illustrate the difference between RMSE obtained by the proposed method and the lowest RMSE among other methods. Since MLP + MUSIC [88] has lower RMSE for SNR of -20 dB, the proposed method doesn't provide improvement. For SNR levels higher than 0 dB, there is not any improvement due to SVT + MUSIC [79] and L1-SVD [37] which are considered impracticle. For other SNR regime and all of the snapshot numbers, the proposed method provides error

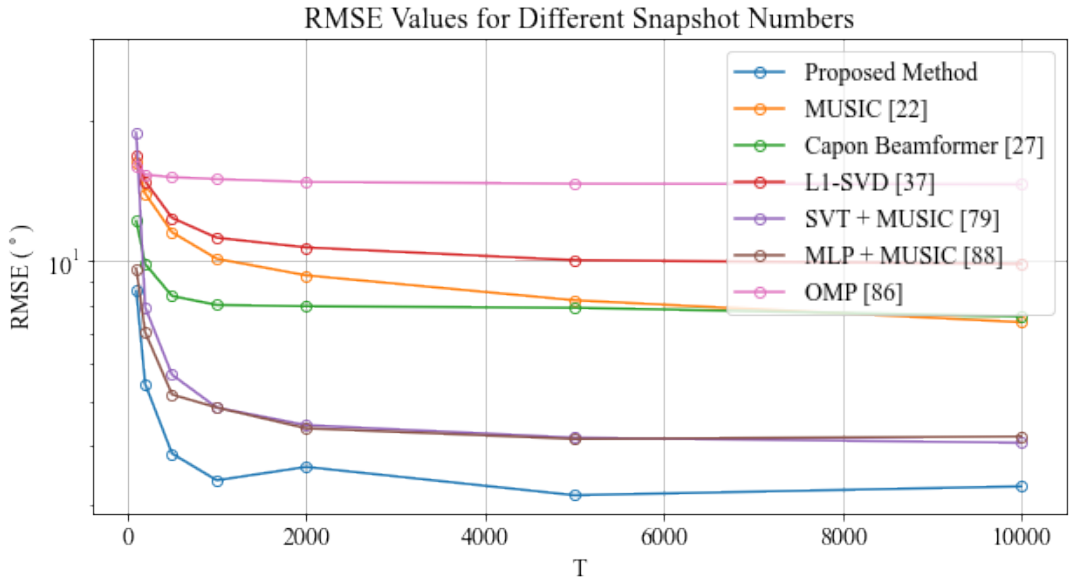

Figure 5.13: RMSE comparison for different snapshot numbers.

improvement.

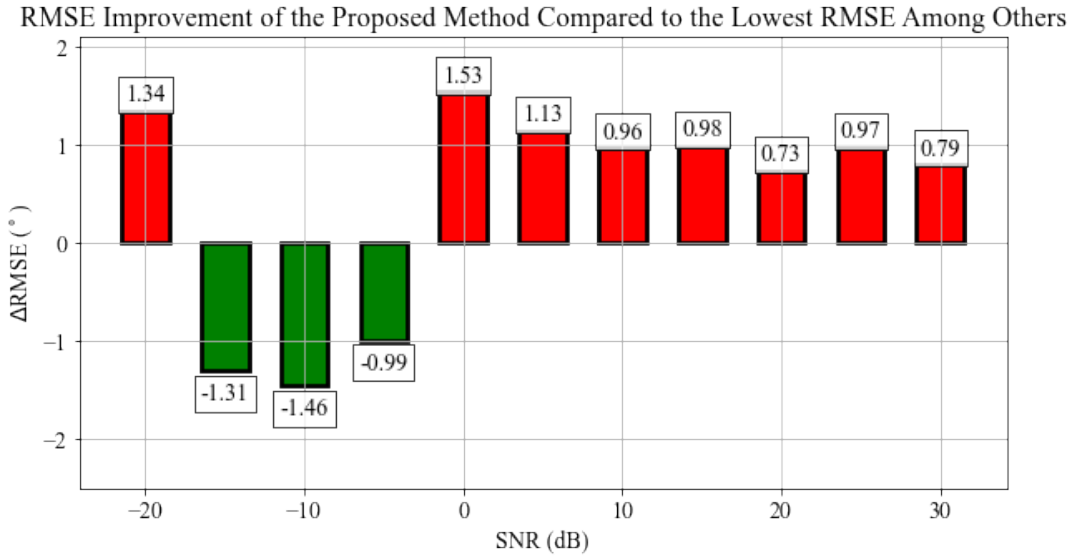

Figure 5.14: RMSE improvement by the proposed method for different SNR levels.

The errors made by each method are illustrated in Appendix F for different SNR and snapshot levels.

Spatial spectrums generated by the methods are shown in Figure 5.16. As in the case of intact array, the highest peaks are achieved by the proposed method. Therefore,

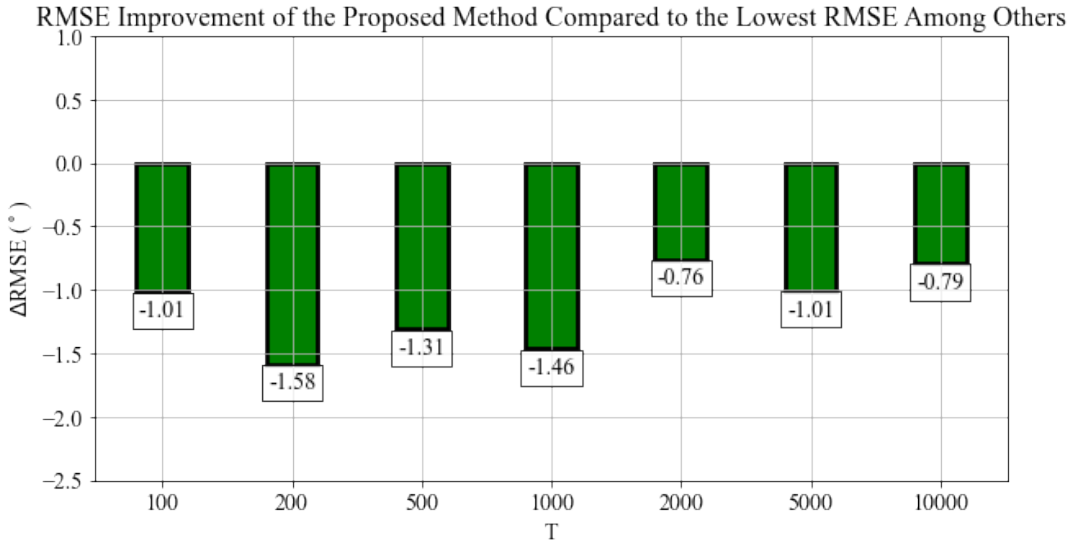

Figure 5.15: RMSE improvement by the proposed method for different snapshot numbers.

encountering sensor failure doesn't affect the spectrum of the proposed method and this shows the robustness of it to sensor malfunctions in terms of spectrum generation.

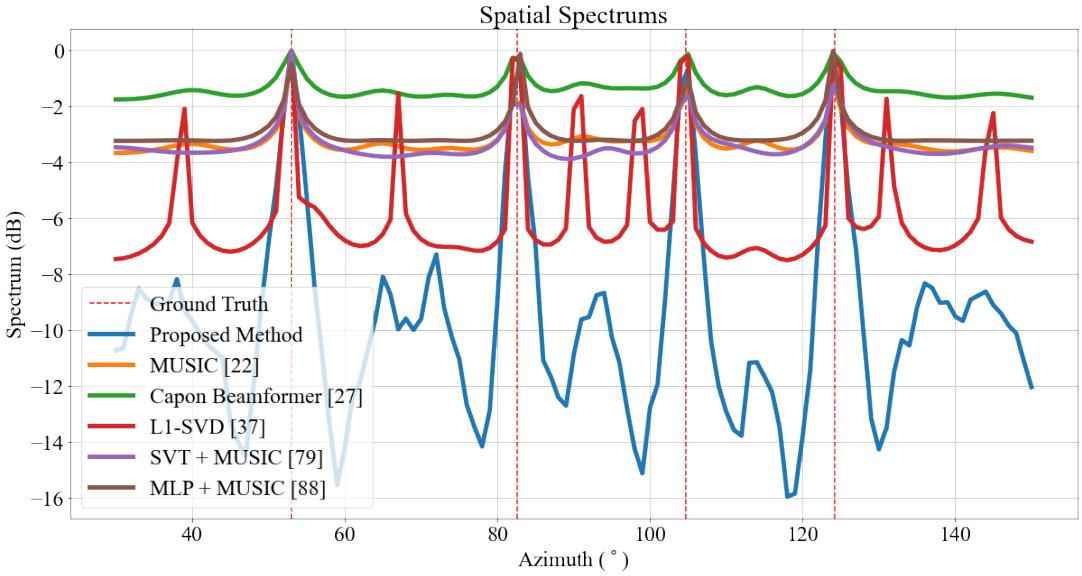

Figure 5.16: Spatial spectrum generated by each method.

For evaluating the robustness of these methods against sensor malfunctions, RMSE values obtained for the cases of intact and faulty array in different SNR levels are plotted on the same graph. The methods are separated into two groups for better visu-

alization as shown in Figures 5.17 and 5.18. For low SNR levels, it can be observed that the lowest RMSE values are obtained by the proposed method for both intact and faulty array cases. Also, it provides the lowest RMSE increase in low SNR region and this indicates the robustness of the proposed method. SVT + MUSIC [79] achieves the lowest RMSE values for SNR levels larger than 0 dB. The proposed method follows SVT + MUSIC [79] in the RMSE values except for SNR values between 10 and 15 dB where L1-SVD [37] seems to be slightly better than it. However, it should be recalled that SVT + MUSIC [79] and L1-SVD [37] requires much more processing time compared to the proposed method and this is an important limitation in practice.

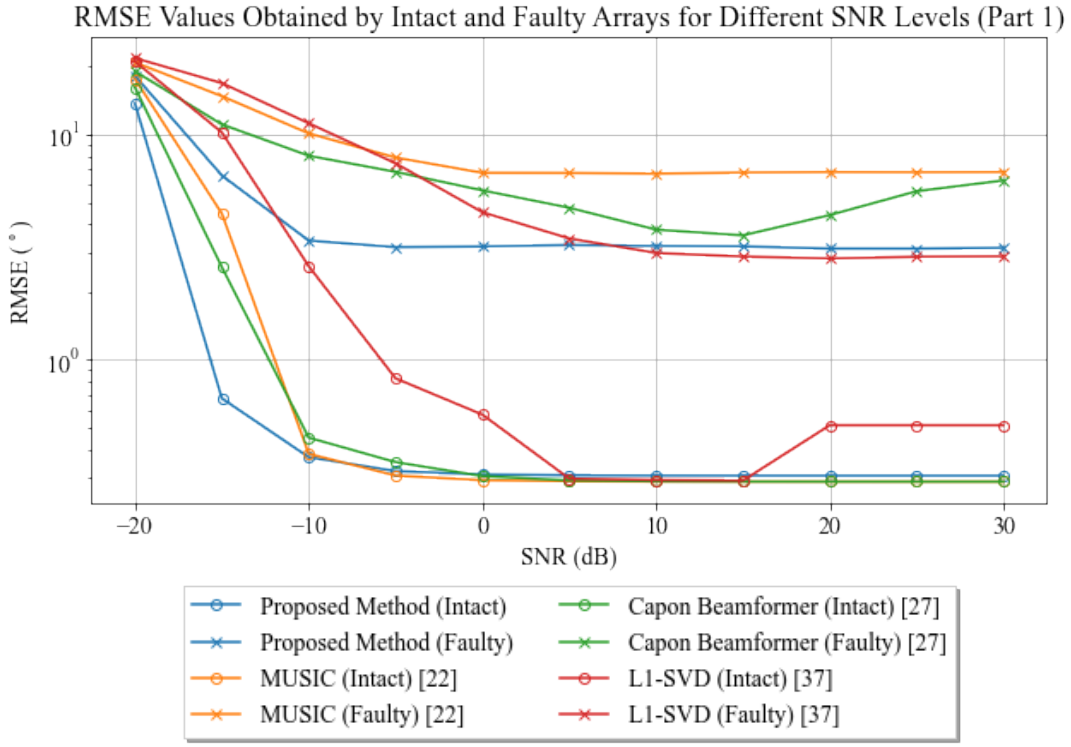

Figure 5.17: RMSE values obtained by intact and faulty arrays for different SNR levels (Part 1).

A similar comparison is conducted for different snapshot numbers as well. Figures 5.19 and 5.20 show RMSE values of the methods for both intact and faulty cases. For all number of snapshots, the proposed method leads to the lowest RMSE change between intact and faulty arrays while providing low error for both cases.

Overall, among the methods which have reasonable processing time, the results in-

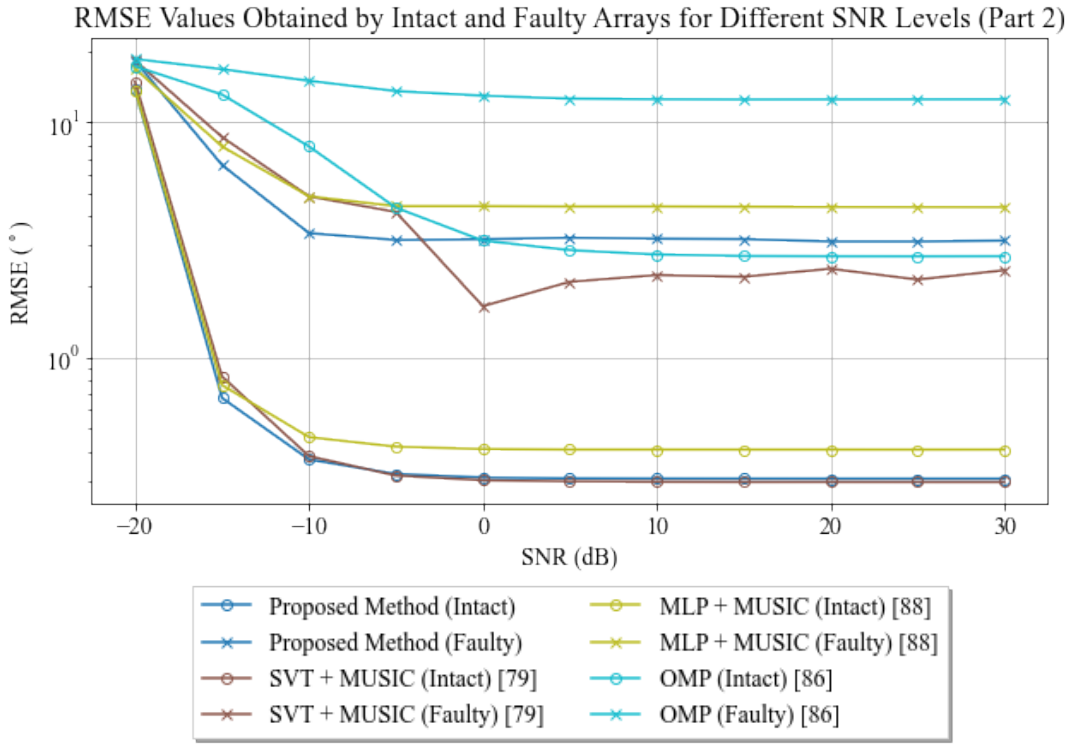

Figure 5.18: RMSE values obtained by intact and faulty arrays for different SNR levels (Part 2).

dicate that the proposed method is the most robust one against sensor malfunctions while providing low error for both intact and faulty arrays.

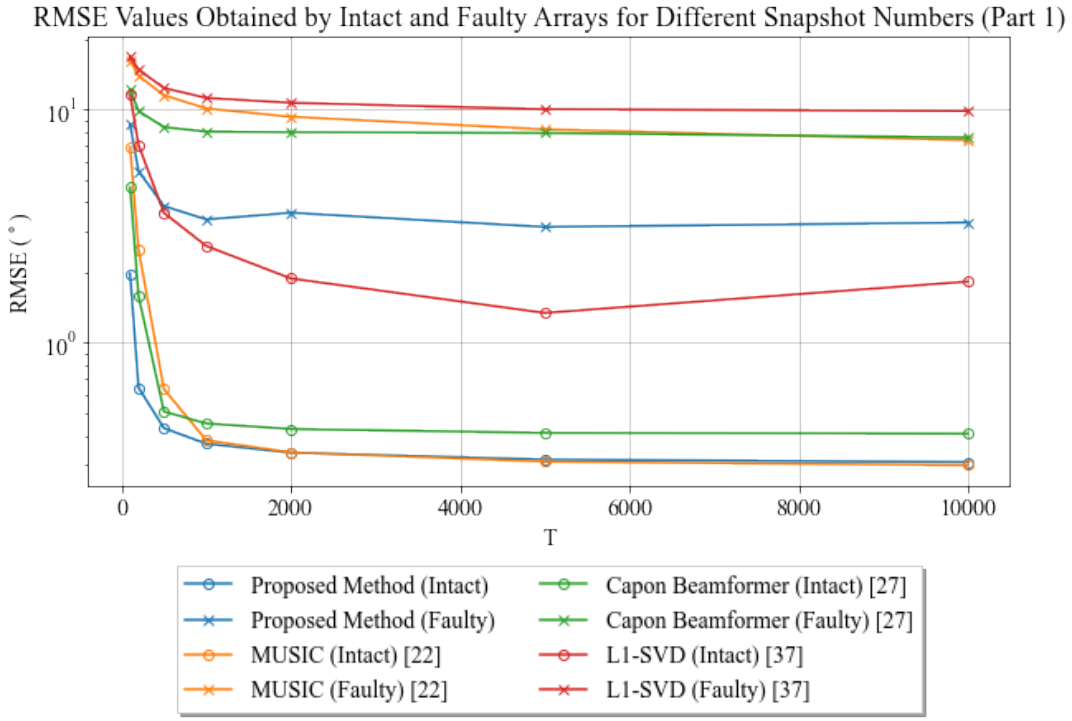

Figure 5.19: RMSE values obtained by intact and faulty arrays for different snapshot levels (Part 1).

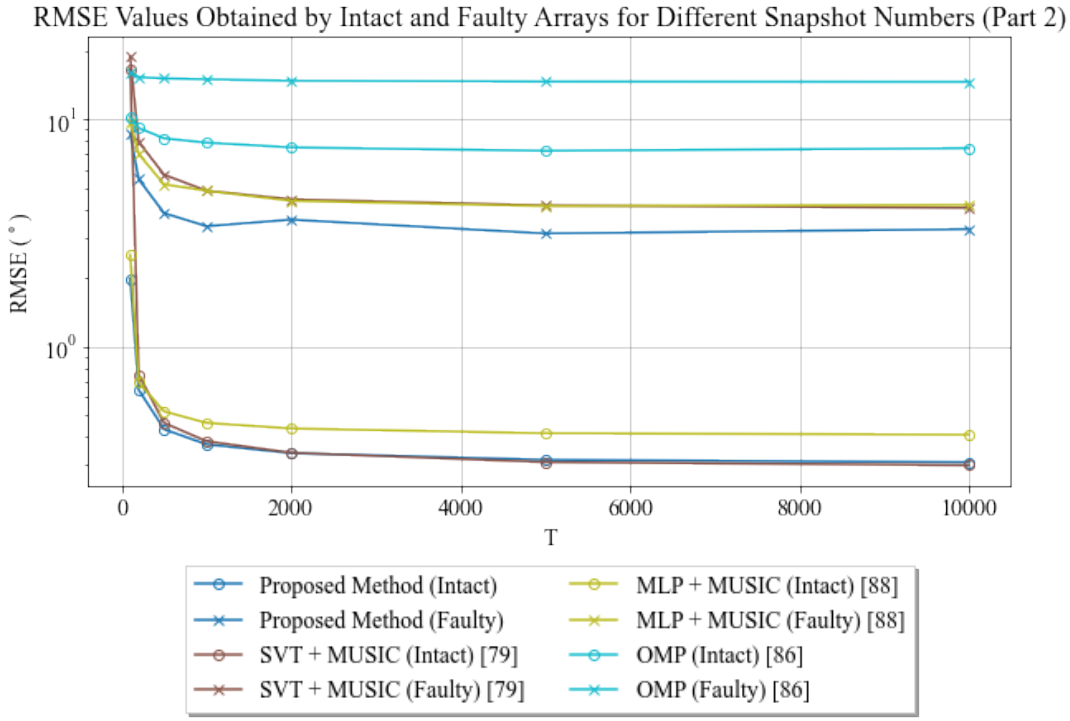

Figure 5.20: RMSE values obtained by intact and faulty arrays for different snapshot levels (Part 2).

CHAPTER 6

PERFORMANCE ANALYSIS: UNKNOWN NUMBER OF SOURCES

In this chapter, performance evaluation of transformer-based DOA estimation method is made for the joint presence of sensor malfunctions and unknown number of sources. In Chapters 4 and 5, it was assumed that the number of sources is known beforehand, therefore there was no need to use the source enumeration network which is the third stage of the proposed method. This chapter is mainly devoted to the performance of source enumeration network and comparison with other enumeration methods. As in Chapters 4 and 5, sparse arrays are the main interest and the same array configuration in Chapter 5 is chosen for the experiments. The physical positions and corresponding difference coarray of this sparse array is illustrated in Figure 6.1.

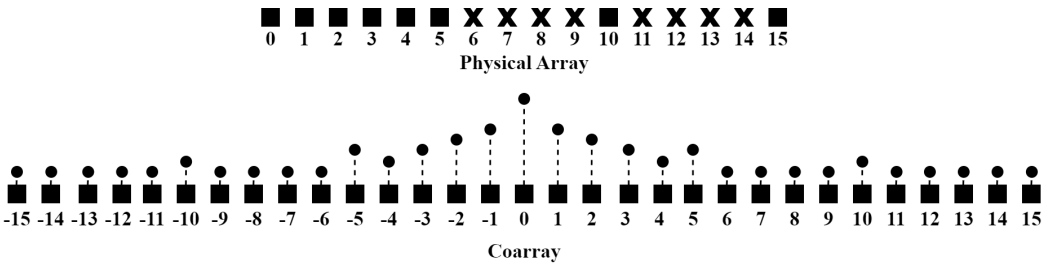

Figure 6.1: Intact sparse array configuration (top) and its coarray (bottom). Rectangle represents presence of the element while cross indicates its absence. The number of dashes in the dashed lines represent the weights for each lag.

The performance comparisons are made with the following methods:

1. AIC [71]
2. MDL [72]
3. SORTE [75]

4. Predicted Eigen-Threshold [76]

5. Eigen-Increment Threshold [77]

6. AREG [74]

7. T-GANE [74]

AIC [71] and MDL [72] are representatives of the information-theoretic based approaches and chosen since they are well-known methods used for source enumeration. As examples of eigenvalue based methods, SORTE [75], AREG [74], Predicted Eigen-Threshold [76] and Eigen-Increment Threshold [77] are selected. T-GANE [74] is the method chosen from the class of data-driven methods. All these methods utilize eigenvalues of the sample covariance matrix, therefore MUSIC [22] is applied with these methods for obtaining DOA estimation results in the experiments. The other methods which are applied in Chapters 4 and 5 couldn't be applied in this part since they cannot generalize in case of sensor malfunctions or they are not able to estimate the number of sources.

6.1 Simulation Details

For training the proposed method, a training set is generated for which the details are given in Table 6.1. It contains samples from aforementioned intact array and its subarrays. The faulty sensor elements are assumed to be known and received signals are generated for remaining sensors. The number of sources for each sample is varied such that source enumeration network can be trained for different number of sources. The number of sources is limited by the minimum number of intact sensors since subspace-based methods cannot function for undetermined case. Other parameters are selected for the same reasons that are explained in Section 4.1.

Covariance reconstruction network and direction of arrival estimation network, which comprise the first two stages of the proposed method, are trained following the same procedure that is described in Chapter 5. Therefore, training details of these networks are not repeated in this chapter for the sake of brevity.

Table 6.1: The specifications of training set

The number of samples (D)	2000000
The number of sources at each sample (N)	1 - 4*
The number of sensors at each sample (M)	4 - 8*
Source separations	$>10^{\circ}$
Source direction (azimuth, θ)	30° - 150°*
Source direction (elevation, ϕ)	0°
The number of snapshots (T)	100 - 1000*
SNR	-20 dB - 0 dB*
Source signal type	Complex random Gaussian
Noise signal type	Complex random Gaussian
Sampling rate	10 kHz

* Uniformly selected between the given interval

The third stage of the proposed method, which is introduced as source enumeration network, is trained for learning the mapping between number of sources and the feature vector that is extracted from the global average pooling block of direction of arrival estimation network. Training is performed on Tesla T4 GPU and the network is implemented using PyTorch framework. Hold-out cross-validation is applied with a validation ratio of 0.1. The output layer of source enumeration network is constructed to have maximum number of sources (N_{max}) that is present in training set which is 4. The ground truths in training set are binary vectors in which the entry corresponding to the number of sources is 1 and others are 0. Supervised learning is applied offline where cross-entropy loss is minimized between the ground truth and network output by optimizing the parameters $\mathbf{\Psi}_{\text{SEN}}$ of the network. Cross-entropy loss can be formulated as

$$\mathbf{\Psi}^{*}_{\text{SEN}} = \underset{\mathbf{\Psi}_{\text{SEN}}}{\arg\min} \frac{1}{D} \sum_{d=1}^{D} L(\boldsymbol{o}_d, \hat{\boldsymbol{o}}_d) \tag{6.1}$$

where

$$L(\boldsymbol{o}_d, \hat{\boldsymbol{o}}_d) = -\frac{1}{N_{max}} \sum_{n=1}^{N_{max}} [\boldsymbol{o}_d(n) \log(\hat{\boldsymbol{o}}_d(n))] \tag{6.2}$$

is cross-entropy loss, $\boldsymbol{o}_d$ is the ground truth and $\hat{\boldsymbol{o}}_d$ represents the network output for

d^{th} sample in validation set. Other training hyperparameters are shown in Table 6.2. Hyperparameter tuning is applied in the selection of these parameters.

Table 6.2: Training hyperparameters

Optimizer	Adam [111] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$
Learning rate	Initially 0.001
Learning rate decay	0.7 once validation loss reaches plateau with a patience of 10 epochs
Batch size	512

For a fair comparison, the data-driven method that is used in the comparison is trained with the same training set. Training process of this method is applied by following the procedures and using the hyperparameters that are given in the corresponding article [74].

For evaluating the trained network and make comparisons with other methods, different test sets are generated containing samples for various SNR levels, snapshot numbers, sensor numbers/configurations and the number of sources. The details of these test sets are described in Table 6.3. Signal types, sampling rate and source elevation are the same as in training set.

As the evaluation metric, average Hausdorff distance is used since RMSE is not an appropriate metric in case the number of sources that is predicted by the network is not equal to the true number of sources. Average Hausdorff distance can be formulated as

$$d_{h,avg} = \frac{1}{D} \sum_{d=1}^{D} d_h(\boldsymbol{Z}_d, \hat{\boldsymbol{Z}}_d) \tag{6.3}$$

where

$$d_h(\boldsymbol{P}_d, \hat{\boldsymbol{P}}_d) = \max \{d(\boldsymbol{P}_d, \hat{\boldsymbol{P}}_d), d(\hat{\boldsymbol{P}}_d, \boldsymbol{P}_d)\} \tag{6.4}$$

is the Hausdorff distance between the predicted angle set $\hat{\boldsymbol{P}}_d$ and true angle set $\boldsymbol{P}_d$ and

$$d(\boldsymbol{P}_d, \hat{\boldsymbol{P}}_d) = \sup \{d(p_d, \hat{\boldsymbol{P}}_d) \mid p_d \in \boldsymbol{P}_d\} \tag{6.5}$$

Table 6.3: The specifications of test sets

Test set #	Number of samples (D)	Number of intact sensors (M)	Number of sources (N)	Source separations	Source direction (θ)	SNR	Number of snapshots (T)
1	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	-20 dB	1000
2	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	-15 dB	1000
3	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	-10 dB	1000
4	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	-5 dB	1000
5	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	0 dB	1000
6	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	5 dB	1000
7	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	10 dB	1000
8	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	15 dB	1000
9	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	20 dB	1000
10	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	25 dB	1000
11	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	30 dB	1000
12	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	-10 dB	100
14	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	-10 dB	500
15	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	-10 dB	1000
16	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	-10 dB	2000
17	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	-10 dB	5000
18	10000	4 - 8*	1 - 4*	$>10^\circ$	30° - 150°*	-10 dB	10000

* Uniformly selected between the given interval

is the directed Hausdorff distance where

$$d(p_d, \hat{\boldsymbol{P}}_d) = \inf \{d(p_d, \hat{p}_d) \mid \hat{p}_d \in \hat{\boldsymbol{P}}_d\} \tag{6.6}$$

and

$$d(p_d, \hat{p}_d) = |p_d - \hat{p}_d| \tag{6.7}$$

Hausdorff distance measures the how far two subsets of a metric space are from each other. It indicates the longest distance from a point that is selected from one subset to the closest point in the other set. Its applicability for two sets which do not share the same cardinality enables the usage for the evaluation of the experiments in this chapter. The unit for Hausdorff distance is degree.

In addition to Hausdorff distance, source number estimation accuracy is utilized for evaluating source enumeration network. It is calculated by

$$\text{Accuracy} = \frac{1}{D}\sum_{d=1}^{D} \mathbb{1}_{N_d}(\hat{N}_d) \tag{6.8}$$

where N_d is the true number of sources and $\hat{N}_d$ is the estimated number of sources for d^{th} sample of test set, $\mathbb{1}_{N_d}$ is indicator function and gets value of 1 if $N_d = \hat{N}_d$ and 0 otherwise.

6.2 Results and Discussion

The loss curves obtained during training source enumeration network are shown in Figure 6.2. The network is trained for 134 epochs and parameters $\mathbf{\Psi}_{\text{SEN}}$ are set when the validation loss is minimized.

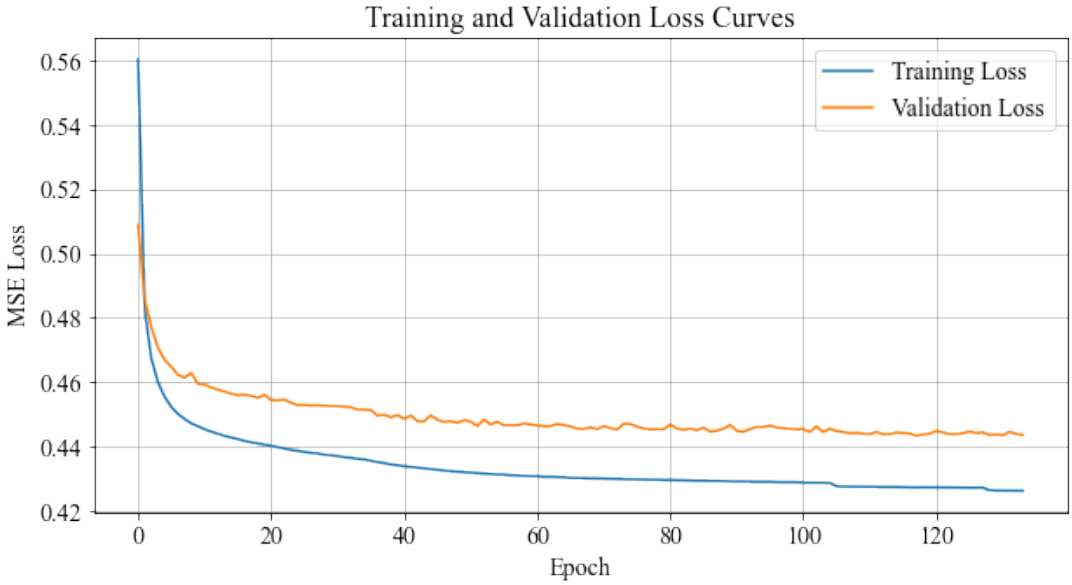

Figure 6.2: Loss curves for source enumeration network.

After training source enumeration network, its performance is evaluated on test set and source enumeration accuracy is compared with other methods for various SNR levels and snapshot numbers as shown in Figures 6.3 and 6.4. It can be observed that the proposed method achieves the highest accuracy especially for low SNR. For

higher SNR regions, it shows comparable performance to MDL [72], Eigen Threshold [76] and T-GANE [74] where the accuracy values fall in the interval of 1.2%. Furthermore, the proposed method outperforms the others for all of the snapshot numbers.

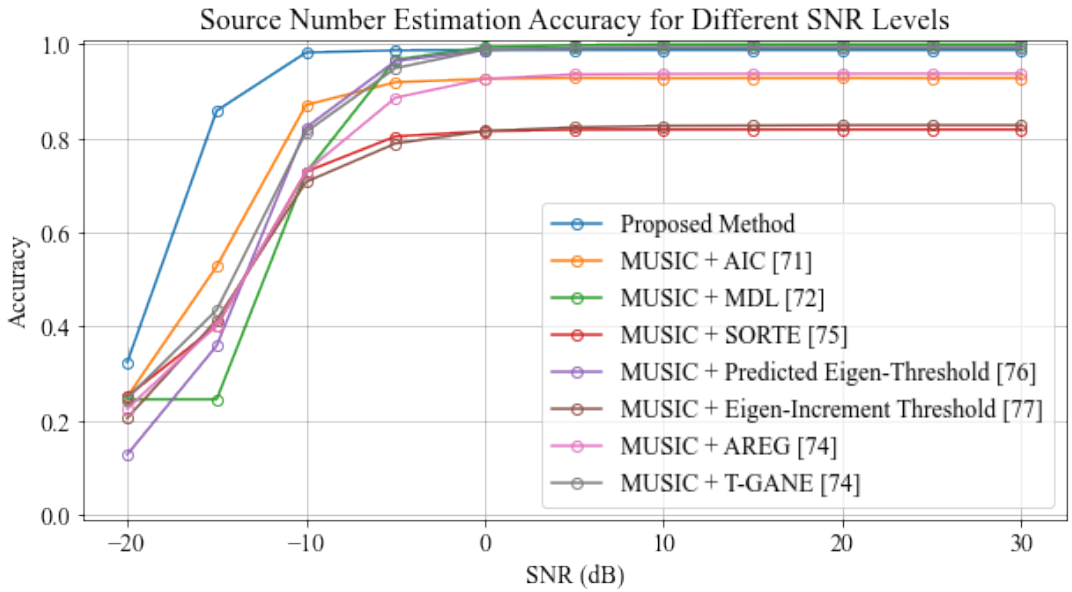

Figure 6.3: The comparison of source enumeration accuracy for different SNR levels.

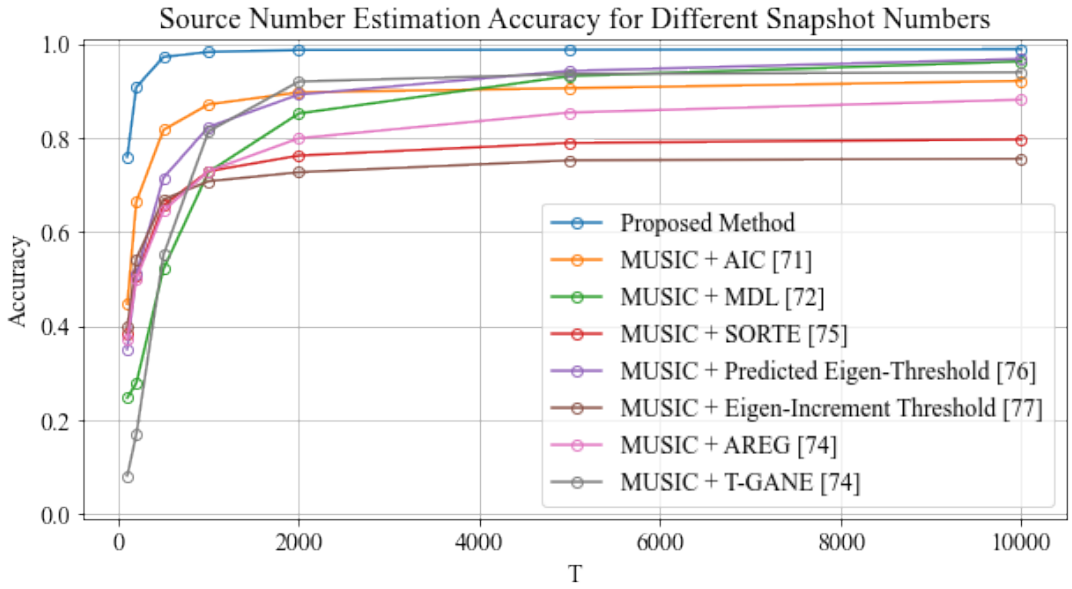

Figure 6.4: The comparison of source enumeration accuracy for different snapshot numbers.

Accuracy improvement achieved by the proposed method over the highest accuracy levels among other methods is illustrated in Figures 6.5 and 6.6 for various SNR and snapshot levels respectively. As it can be observed, the proposed method improves accuracy for low SNR levels and the whole snapshot region. For high SNR regime, it shows comparable performance to the best performing method.

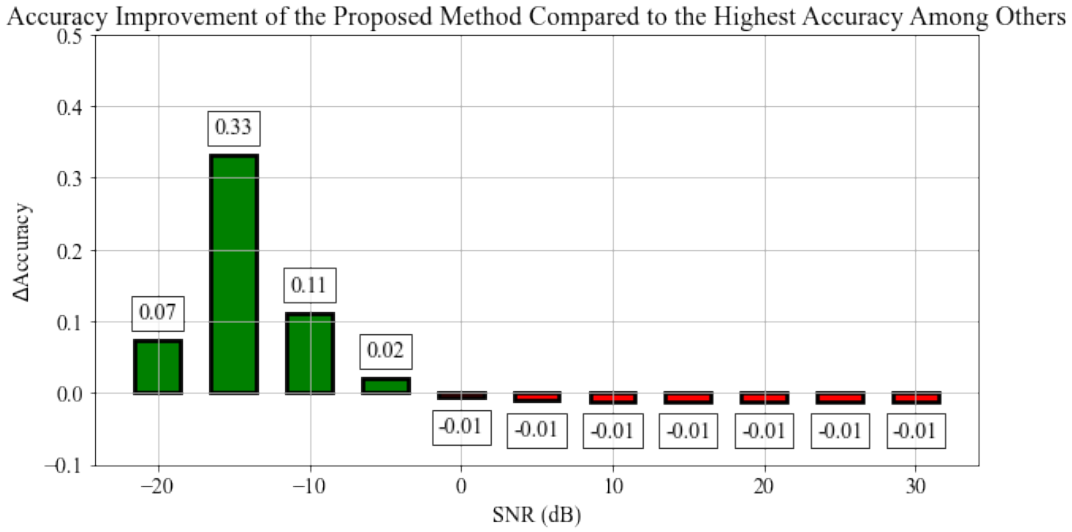

Figure 6.5: Accuracy improvement by the proposed method for different SNR levels.

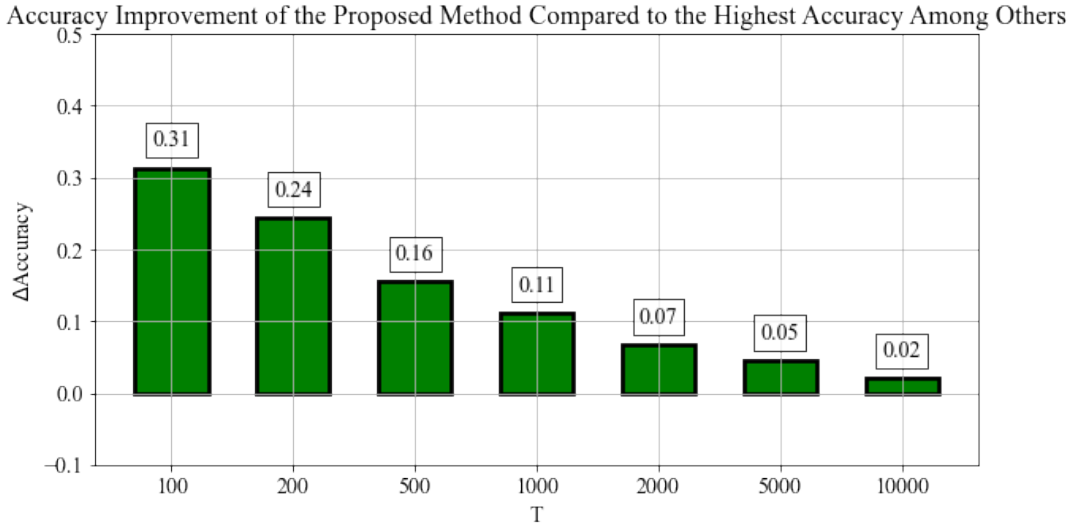

Figure 6.6: Accuracy improvement by the proposed method for different snapshot numbers.

Confusion matrices which indicate the percentage for the actual and estimated num-

ber of sources by the proposed method are given in Figure 6.7. For SNR of -20 dB, accuracy for any number of sources doesn't exceed 60% while at least 96% is achieved for SNR larger than -10 dB. A flooring effect is observed for SNR levels higher than 10 dB which can also be seen in Figure 6.3. Similar observations can be made for snapshot numbers as well. For snapshot number of 100, accuracy levels are relatively low while it gets better as the snapshot number increases to some extent after which flooring effect appears.

Figures 6.8 and 6.9 show Hausdorff distance values obtained by each method for different SNR levels and snapshot numbers respectively. In these graphs, the performance of the proposed method and MUSIC [22] for known number of sources are also included. The proposed method leads to the lowest Hausdorff distance in case of unknown number of sources for low SNR. MUSIC + MDL [72] achieves the lowest Hausdorff distance in the high SNR region while MUSIC + T-GANE [74] and the proposed method have the lowest distance values after it within the interval of 0.2. In addition, it can be observed that the proposed method has lower Hausdorff distance when the number of sources known compared to the case where source enumeration network is used to estimate the number of sources. This is expected since prior information about the number of sources prevents estimated and true angle sets to have different cardinalities which in turn decreases the possibility of higher distance calculations. On the other hand, it can be seen that the proposed method with source enumeration network yields lower distance values for some SNR levels compared to MUSIC [22] with known number of sources. A similar observation can be made for different snapshot values as well where the proposed method has higher performance compared to both MUSIC [22] with known number of sources and other methods with unknown number of sources.

The improvement provided by the proposed method in Hausdorff distance is shown in Figures 6.10 and 6.11 for different SNR and snapshot levels. A significant decrease in Hausdorff distance can be observed especially for low SNR and snapshot regions. For large snapshot numbers, there is still improvement while similar performance to the best performing method is achieved in high SNR levels.

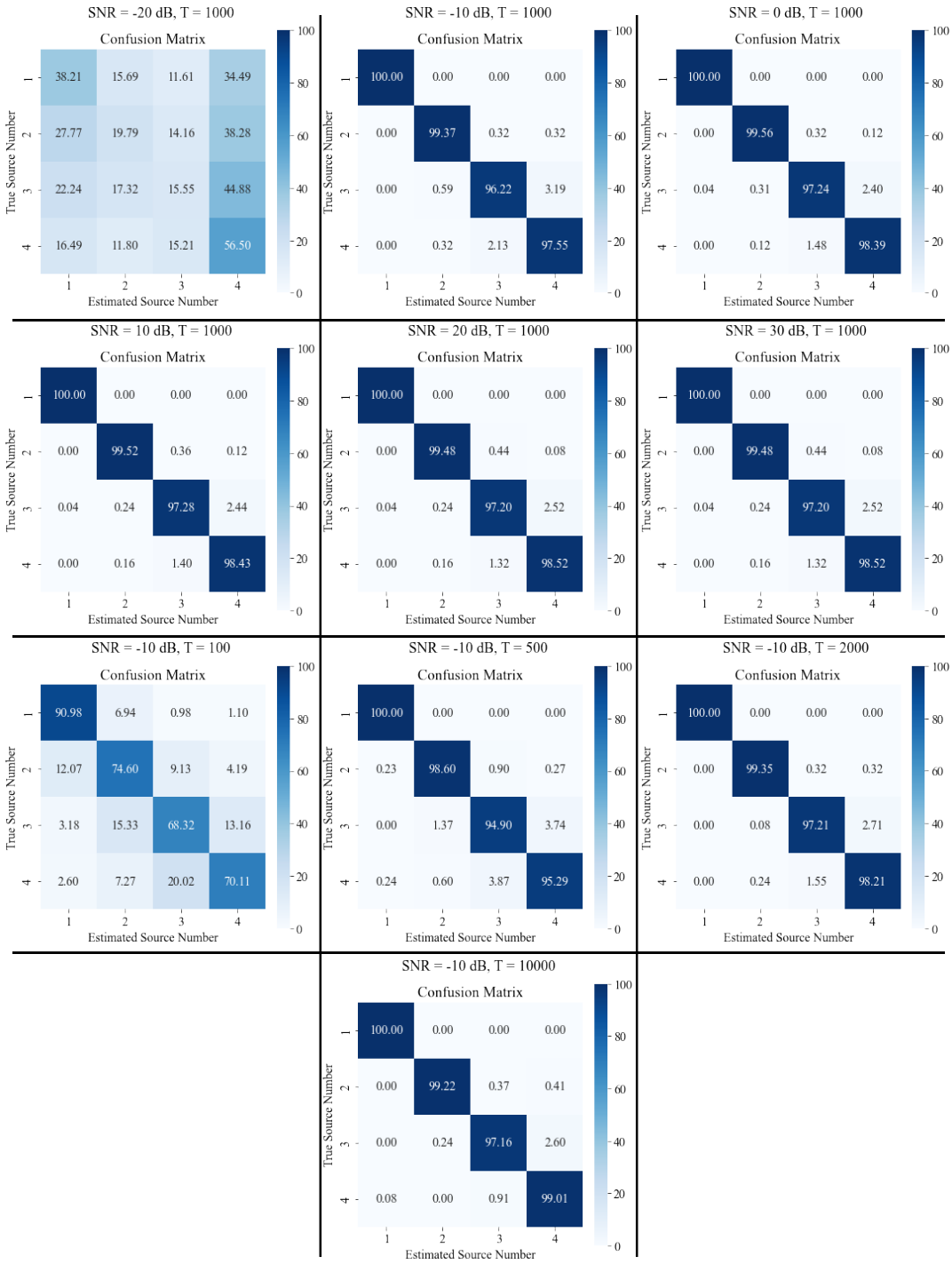

Figure 6.7: Confusion matrices of the proposed method for different SNR levels and snapshot numbers.

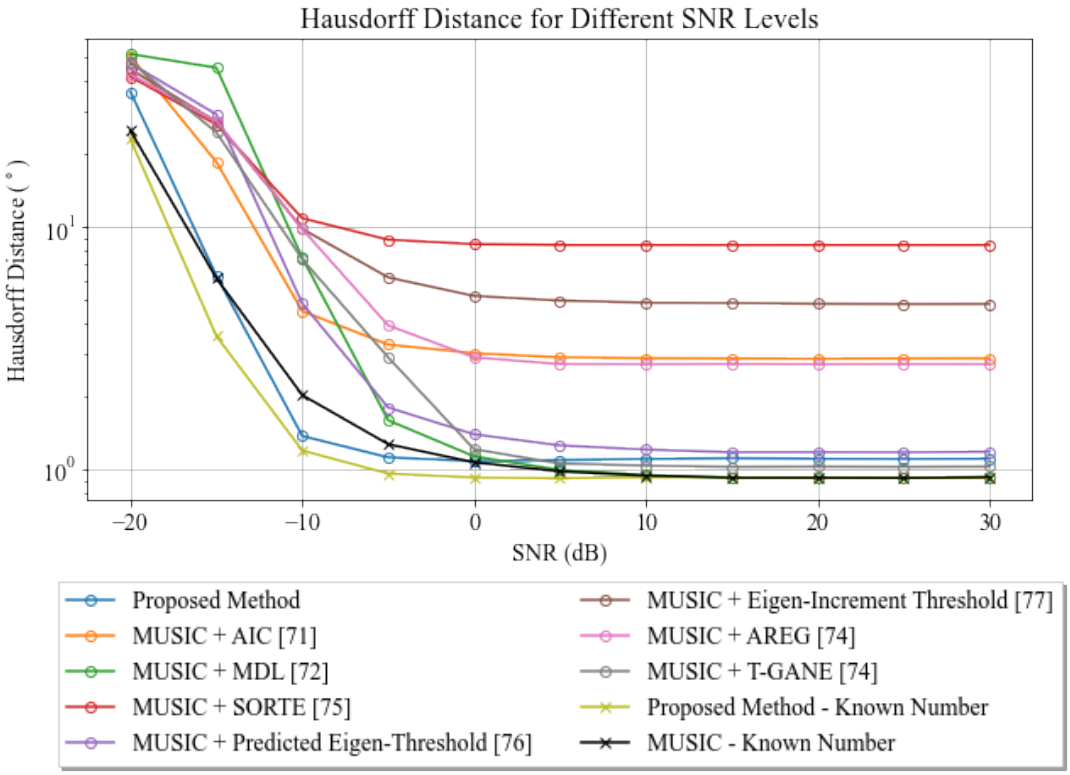

Figure 6.8: Hausdorff distance comparison for different SNR levels.

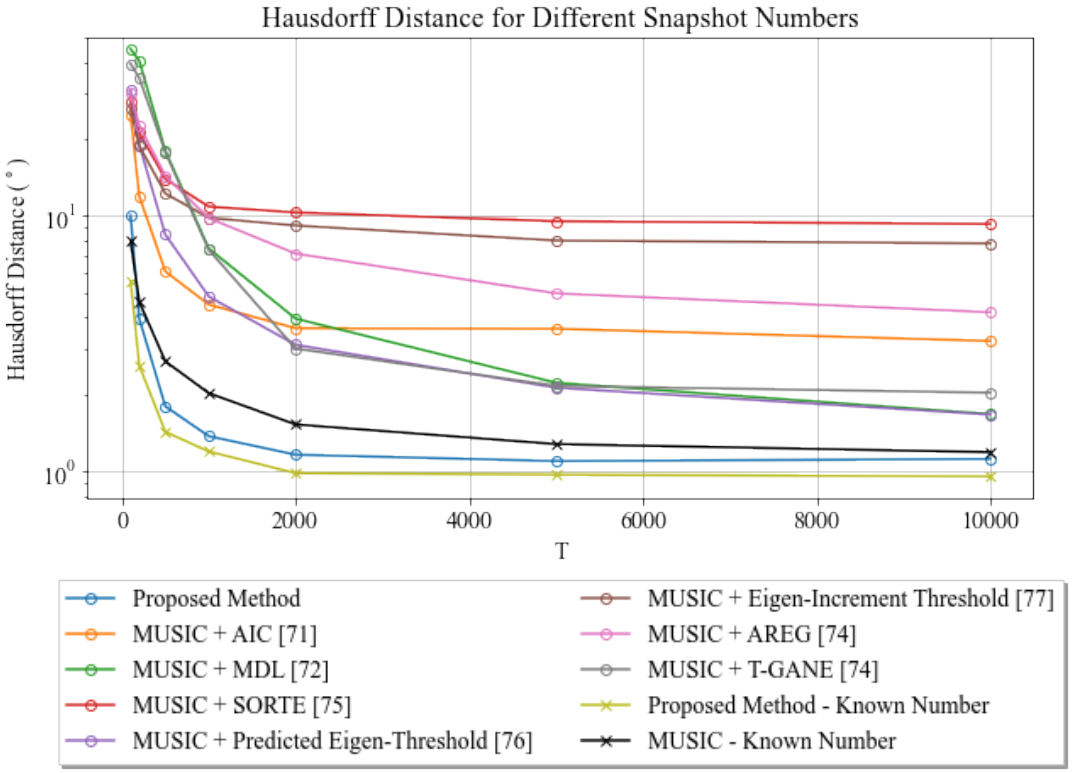

Figure 6.9: Hausdorff distance comparison for different snapshot numbers.

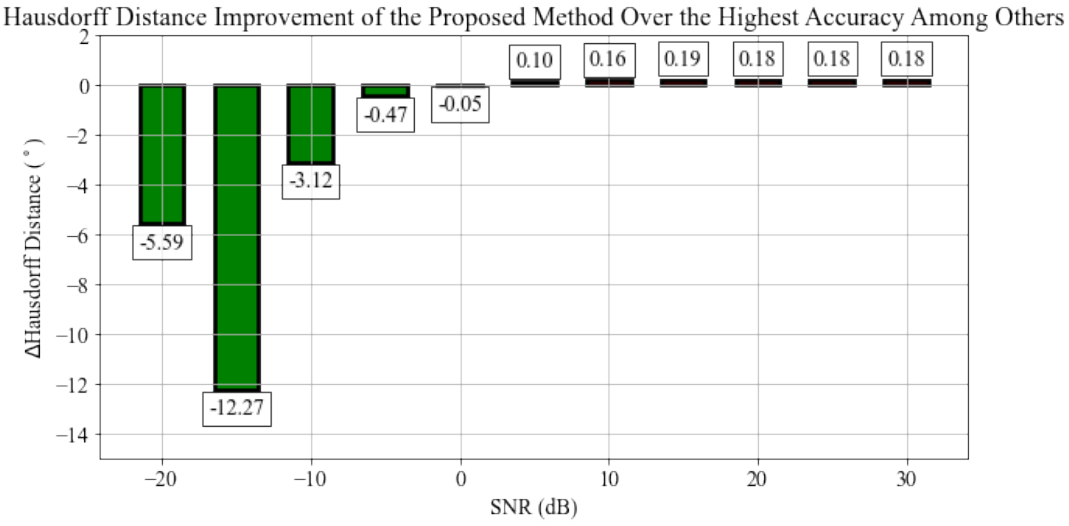

Figure 6.10: Hausdorff distance improvement by the proposed method for different SNR levels.

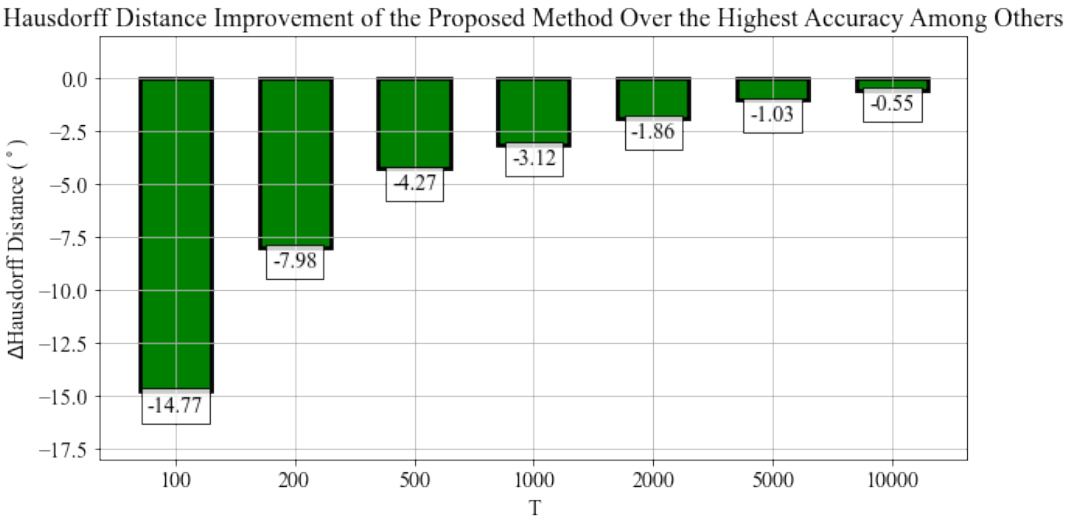

Figure 6.11: Hausdorff distance improvement by the proposed method for different snapshot numbers.

CHAPTER 7

CONCLUSION

7.1 Conclusion

In this thesis, a transformer-based DOA estimation and source enumeration method is proposed. It is developed with purpose of generalization and robustness in the joint presence of sensor malfunctions and unknown number of sources. Sparse linear arrays are the main array type that this method is proposed for due to their advantages over ULA [4].

For evaluating the proposed method, three experiments are conducted for which the scenarios are ordered from general to specific.

In the first experiment, performance of the proposed method is evaluated for different types of sparse arrays such as minimum redundant array, nested array and coprime array. A single array configuration is selected for each type and performance evaluations and comparisons are made for them.

It is shown that covariance reconstruction network, which is the first stage of the proposed method, is able to enhance the noisy covariance matrix for different SNR levels and snapshot numbers. This observation is valid for all the mentioned sparse array types. Another noteworthy aspect is that the enhancement level gets larger as SNR level and snapshot number decreases. This indicates that covariance reconstruction network can mitigate the distorting effects of high level of noise and small amount of sampled data.

The proposed method, with the first two stages are activated, is compared with other subspace-based, beamforming, sparsity-inducing and data-driven methods in terms

of RMSE values obtained in DOA estimation task. It is observed that it achieves the lowest RMSE for low SNR levels and small number of snapshots for all the specified sparse array types. This demonstrates the resilience of the proposed method into the presence of noise and relatively small amount of data collected by array independent of the sparse array type. For high SNR levels and large number of snapshots, it doesn't exhibit the best performance but provides a similar performance to the best performing method by obtaining an RMSE value which is 0.03 $^{\circ}$ larger than the lowest achieved. In these experiments, the best performing methods are generally data-driven methods and the proposed method outperforms conventional methods such as MUSIC, Capon beamformer and L1-SVD. This demonstrates the effectiveness of learning the nonlinear relationships in the sensor measurements over utilizing only signal model for DOA estimation. The proposed method improves the performance of data-driven methods especially in low SNR levels and snapshot numbers without considerable performance loss in high SNR levels and snapshot numbers. In terms of spatial spectrum generation, it generates higher peaks in the source direction compared to conventional methods. A drawback of the proposed method is that other data-driven methods can achieve higher peaks however, it should be noted that other data-driven methods cannot be used in case of sensor malfunctions which limits their applicability in practice. From the point of processing times, the proposed method is the least demanding method among data-driven and hybrid methods and has the lowest time after Capon Beamformer [27] and MUSIC [22] among all the compared methods. Therefore, it provides performance improvement without too much computational burden.

In the second experiment, a sparse array configuration is selected and two scenarios are employed for performance evaluation and comparison. The first scenario consists of intact array while in the second one, random sensor malfunctions are introduced in the array. In this experiment, data-driven methods that are employed in the first experiment couldn't be used because their input formulations are not suitable to handle sensor malfunctions which lead to different sensor number/configurations.

As in the case of the first experiment, it is observed that covariance reconstruction network is able to enhance the noisy covariance matrix in case of sensor malfunctions as well. Mitigating effect is larger for low SNR levels and snapshot numbers.

In terms of DOA estimation, the proposed network is compared with subspace-based, beamforming, sparsity-inducing methods and other methods which are proposed specifically for sensor malfunctions. In the first scenario, it has consistent performance and achieves the best results for a range of SNR and snapshot values. For the second scenario, it achieves lowest RMSE for all snapshot numbers and low SNR levels. SVT + MUSIC [79] and L1-SVD [37] have better performance for high SNR levels, however they are computationally much more demanding which makes them impractical. When RMSE change between two scenarios are compared, the proposed method is the most robust one against sensor malfunctions for most of the SNR and snapshot levels while providing low error for both scenarios and having feasible processing time. In terms of spatial spectrum, the proposed method have the highest peaks for the source directions and these height of the peaks are not affected by sensor malfunctions. This indicates that the spatial spectrum generated by the proposed method is robust against sensor failures. Furthermore, it has the one of the lowest processing times which shows its practicality.

In the third experiment, the same array configuration is used as in the second experiment. There are sensor malfunctions which occur randomly in the array. Additionally, the number of sources is unknown. Data-driven methods that are proposed in the second experiment specifically for sensor malfunctions couldn't be used in this experiment since they cannot estimate the number of sources.

Source enumeration network, which composes the third stage of the proposed method, is evaluated in terms of source number estimation accuracy. It is shown that the proposed method shows the highest accuracy for low SNR levels and the whole snapshot numbers. It doesn't have the highest accuracy for high SNR but shows comparable performance to the best performing method by having an accuracy 1.2% lower than the highest achieved. Furthermore, it is observed that the error made by the proposed method for the case of unknown number of sources is lower than that of MUSIC [22] with known number of sources. This demonstrates the effectiveness of the proposed method.

Overall, experiments indicate that the proposed method provides

- denoising effect in the covariance matrix through reconstruction network

- lower error for low SNR and snapshot region
- similar error to the best performing method for high SNR and snapshot region
- more robust performance to sensor malfunctions which lead to array configuration/size variations
- lower processing time compared to the most of the data-driven and hybrid techniques
- higher source enumeration accuracy especially for low SNR and snapshot numbers
- similar source enumeration accuracy to the best performing method for high SNR and snapshot region

7.2 Future Work

In this thesis, sensor malfunctions are inspected by limiting the scope into complete failures in the known sensors. In practice, other classes of malfunction can also be encountered. For instance, it is possible that a faulty sensor is not recognized by the operator or system. For these kinds of situations, enriching the ability of the proposed method by array diagnosis may increase its practical applicability. Array diagnosis can be a part of covariance reconstruction network since reconstruction operation includes detection of distortions in the data.

Another important research direction would be to reformulate the proposed method such that it can function for sparse planar arrays as well. Planar arrays have a plenty of application areas in practice and adapting the proposed method for these arrays may increase its potential application fields. In such a problem, difference coarray would be considered as 2-dimensional sequential data and separate transformer models can be applied on each dimension.

The proposed method is constructed to have on-grid angle estimates. On-grid approach limits the performance of DOA estimation methods because of flooring effect which is observed in Chapter 4 while comparing the results with a gridless method.

An improvement to this study might be to develop an additional block over the output layer of the proposed network such that it produces off-grid angle estimates. This block can be based upon conventional or recent data-driven interpolation methods.

REFERENCES

[1] H. L. Van Trees, *Optimum array processing: Part IV of detection, estimation, and modulation theory*. John Wiley & Sons, 2002.

[2] H. Krim and M. Viberg, "Two decades of array signal processing research: the parametric approach," *IEEE Signal Processing Magazine*, vol. 13, no. 4, pp. 67–94, 1996.

[3] Z. Yang, X. Chen, and X. Wu, "A robust and statistically efficient maximum-likelihood method for DOA estimation using sparse linear arrays," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 5, pp. 6798–6812, 2023.

[4] S. R. Pavel, M. W. T. Chowdhury, Y. D. Zhang, D. Shen, and G. Chen, "Machine learning-based direction-of-arrival estimation exploiting distributed sparse arrays," in *2021 55th Asilomar Conference on Signals, Systems, and Computers*, pp. 241–245, IEEE, 2021.

[5] R. Chellappa and S. Theodoridis, "Academic press library in signal processing volume 3: Array and statistical signal processing," 2013.

[6] S. Ge, K. Li, and S. N. B. M. Rum, "Deep learning approach in DOA estimation: A systematic literature review," *Mobile Information Systems*, vol. 2021, no. 1, p. 6392875, 2021.

[7] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[8] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[9] Q. Pan, C. Mei, N. Tian, B. W.-K. Ling, and E. X. Wang, "Source enumeration

based on a uniform circular array in a determined case," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 700–712, 2018.

[10] J. Rogers, J. E. Ball, and A. C. Gurbuz, "Estimating the number of sources via deep learning," in *2019 IEEE Radar Conference (RadarConf)*, pp. 1–5, IEEE, 2019.

[11] Y. Yang, F. Gao, C. Qian, and G. Liao, "Model-aided deep neural network for source number detection," *IEEE Signal Processing Letters*, vol. 27, pp. 91–95, 2019.

[12] G. C. Lee, A. S. Rawat, and G. W. Wornell, "Robust direction of arrival estimation in the presence of array faults using snapshot diversity," in *2019 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 1–5, IEEE, 2019.

[13] C.-L. Liu and P. P. Vaidyanathan, "Robustness of difference coarrays of sparse arrays to sensor failures—part i: A theory motivated by coarray MUSIC," *IEEE Transactions on Signal Processing*, vol. 67, no. 12, pp. 3213–3226, 2019.

[14] A. V. Oppenheim, *Discrete-time signal processing*. Pearson Education India, 1999.

[15] S. Zhang, A. Ahmed, Y. D. Zhang, and S. Sun, "Enhanced DOA estimation exploiting multi-frequency sparse array," *IEEE Transactions on Signal Processing*, vol. 69, pp. 5935–5946, 2021.

[16] D. H. Johnson, "Array signal processing," *Concepts and Techniques*, 1993.

[17] A. Moffet, "Minimum-redundancy linear arrays," *IEEE Transactions on Antennas and Propagation*, vol. 16, no. 2, pp. 172–175, 1968.

[18] E. Vertatschitsch and S. Haykin, "Nonredundant arrays," *Proceedings of the IEEE*, vol. 74, no. 1, pp. 217–217, 1986.

[19] P. Pal and P. P. Vaidyanathan, "Nested arrays: A novel approach to array processing with enhanced degrees of freedom," *IEEE Transactions on Signal Processing*, vol. 58, no. 8, pp. 4167–4181, 2010.

[20] P. P. Vaidyanathan and P. Pal, "Sparse sensing with co-prime samplers and arrays," *IEEE Transactions on Signal Processing*, vol. 59, no. 2, pp. 573–586, 2010.

[21] T. E. Tuncer and B. Friedlander, *Classical and modern direction-of-arrival estimation*. Academic Press, 2009.

[22] R. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Transactions on Antennas and Propagation*, vol. 34, no. 3, pp. 276–280, 1986.

[23] C.-L. Liu and P. Vaidyanathan, "Super nested arrays: Linear sparse arrays with reduced mutual coupling—part i: Fundamentals," *IEEE Transactions on Signal Processing*, vol. 64, no. 15, pp. 3997–4012, 2016.

[24] J. Liu, Y. Zhang, Y. Lu, S. Ren, and S. Cao, "Augmented nested arrays with enhanced DOF and reduced mutual coupling," *IEEE Transactions on Signal Processing*, vol. 65, no. 21, pp. 5549–5563, 2017.

[25] L. Wu, Z.-M. Liu, and Z.-T. Huang, "Deep convolution network for direction of arrival estimation with sparse prior," *IEEE Signal Processing Letters*, vol. 26, no. 11, pp. 1688–1692, 2019.

[26] P.-J. Chung, M. Viberg, and J. Yu, "DOA estimation methods and algorithms," in *Academic Press Library in Signal Processing*, vol. 3, pp. 599–650, Elsevier, 2014.

[27] J. Capon, "High-resolution frequency-wavenumber spectrum analysis," *Proceedings of the IEEE*, vol. 57, no. 8, pp. 1408–1418, 1969.

[28] G. K. Papageorgiou, M. Sellathurai, and Y. C. Eldar, "Deep networks for direction-of-arrival estimation in low SNR," *IEEE Transactions on Signal Processing*, vol. 69, pp. 3714–3729, 2021.

[29] R. Roy and T. Kailath, "ESPRIT-estimation of signal parameters via rotational invariance techniques," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 7, pp. 984–995, 1989.

[30] S. Reddi, "Multiple source location-a digital approach," *IEEE Transactions on Aerospace and Electronic Systems*, no. 1, pp. 95–105, 1979.

[31] S. Marcos, A. Marsal, and M. Benidir, "The propagator method for source bearing estimation," *Signal Processing*, vol. 42, no. 2, pp. 121–138, 1995.

[32] T. K. Sarkar and O. Pereira, "Using the matrix pencil method to estimate the parameters of a sum of complex exponentials," *IEEE Antennas and Propagation Magazine*, vol. 37, no. 1, pp. 48–55, 1995.

[33] A. Barabell, "Improving the resolution performance of eigenstructure-based direction-finding algorithms," in *ICASSP'83. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 8, pp. 336–339, IEEE, 1983.

[34] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[35] R. Zheng, S. Sun, H. Liu, H. Chen, M. Soltanalian, and J. Li, "Antenna failure resilience: Deep learning-enabled robust DOA estimation with single snapshot sparse arrays," *arXiv preprint arXiv:2405.02788*, 2024.

[36] D. L. Donoho and M. Elad, "Optimally sparse representation in general (nonorthogonal) dictionaries via 1 minimization," *Proceedings of the National Academy of Sciences*, vol. 100, no. 5, pp. 2197–2202, 2003.

[37] D. Malioutov, M. Cetin, and A. S. Willsky, "A sparse signal reconstruction perspective for source localization with sensor arrays," *IEEE Transactions on Signal Processing*, vol. 53, no. 8, pp. 3010–3022, 2005.

[38] Z. Yang, J. Li, P. Stoica, and L. Xie, "Sparse methods for direction-of-arrival estimation," in *Academic Press Library in Signal Processing, Volume 7*, pp. 509–581, Elsevier, 2018.

[39] S. F. Cotter, B. D. Rao, K. Engan, and K. Kreutz-Delgado, "Sparse solutions to linear inverse problems with multiple measurement vectors," *IEEE Transactions on Signal Processing*, vol. 53, no. 7, pp. 2477–2488, 2005.

[40] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 58, no. 1, pp. 267–288, 1996.

[41] Z. Zhang and B. D. Rao, "Sparse signal recovery with temporally correlated source vectors using sparse Bayesian learning," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 5, pp. 912–926, 2011.

[42] G. Tang, B. N. Bhaskar, P. Shah, and B. Recht, "Compressed sensing off the grid," *IEEE Transactions on Information Theory*, vol. 59, no. 11, pp. 7465–7490, 2013.

[43] B. Allen and M. Ghavami, *Adaptive array systems: fundamentals and applications*. John Wiley & Sons, 2005.

[44] P. Chen, Z. Chen, L. Liu, Y. Chen, and X. Wang, "SDOA-net: An efficient deep learning-based DOA estimation network for imperfect array," *IEEE Transactions on Instrumentation and Measurement*, 2024.

[45] D. Chen, S. Shi, X. Gu, and B. Shim, "Robust DOA estimation using denoising autoencoder and deep neural networks," *IEEE Access*, vol. 10, pp. 52551–52564, 2022.

[46] G. K. Papageorgiou and M. Sellathurai, "Fast direction-of-arrival estimation of multiple targets using deep learning and sparse arrays," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4632–4636, IEEE, 2020.

[47] S. Feintuch, J. Tabrikian, I. Bilik, and H. Permuter, "Neural-network-based DOA estimation in the presence of non-gaussian interference," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 60, no. 1, pp. 119–132, 2023.

[48] H. Xiang, B. Chen, M. Yang, S. Xu, and Z. Li, "Improved direction-of-arrival estimation method based on LSTM neural networks with robustness to array imperfections," *Applied Intelligence*, vol. 51, pp. 4420–4433, 2021.

[49] Y. Yang, M. Zhang, S. Peng, M. Ye, and Y. Zhang, "Direction-of-arrival estimation for a random sparse linear array based on a graph neural network," *Sensors*, vol. 24, no. 1, p. 91, 2023.

[50] D. H. Shmuel, J. P. Merkofer, G. Revach, R. J. Van Sloun, and N. Shlezinger, "Deep root MUSIC algorithm for data-driven DOA estimation," in *ICASSP*

2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1–5, IEEE, 2023.

[51] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[52] A. Randazzo, M. A. Abou-Khousa, M. Pastorino, and R. Zoughi, "Direction of arrival estimation based on support vector regression: Experimental validation and comparison with MUSIC," *IEEE Antennas and Wireless Propagation Letters*, vol. 6, pp. 379–382, 2007.

[53] M. Dehghanpour, V. T. Vakili, and A. Farrokhi, "DOA estimation using multiple kernel learning SVM considering mutual coupling," in *2012 Fourth International Conference on Intelligent Networking and Collaborative Systems*, pp. 55–61, IEEE, 2012.

[54] R. Wang, B. Wen, and W. Huang, "A support vector regression-based method for target direction of arrival estimation from HF radar data," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 5, pp. 674–678, 2018.

[55] A. M. Elbir, "DeepMUSIC: Multiple signal classification via deep learning," *IEEE Sensors Letters*, vol. 4, no. 4, pp. 1–4, 2020.

[56] R. Akter, V.-S. Doan, T. Huynh-The, and D.-S. Kim, "RFDOA-net: An efficient convnet for RF-based DOA estimation in UAV surveillance systems," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 11, pp. 12209–12214, 2021.

[57] Y. Yuan, S. Wu, M. Wu, and N. Yuan, "Unsupervised learning strategy for direction-of-arrival estimation network," *IEEE Signal Processing Letters*, vol. 28, pp. 1450–1454, 2021.

[58] J. Yu and Y. Wang, "Deep learning-based multipath DOAs estimation method for mmwave massive mimo systems in low SNR," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 6, pp. 7480–7490, 2023.

[59] K. Liu, X. Wang, J. Yu, and J. Ma, "Attention based DOA estimation in the presence of unknown nonuniform noise," *Applied Acoustics*, vol. 211, p. 109506, 2023.

[60] M. Chen, Y. Gong, and X. Mao, "Deep neural network for estimation of direction of arrival with antenna array," *IEEE Access*, vol. 8, pp. 140688–140698, 2020.

[61] 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.

[62] Y. Guo, Z. Zhang, and Y. Huang, "Dual class token vision transformer for direction of arrival estimation in low SNR," *IEEE Signal Processing Letters*, 2023.

[63] W. Wang, L. Zhou, K. Ye, H. Sun, and S. Hong, "A DOA estimation method based on an improved transformer model for uniform linear arrays with low SNR," *IET Signal Processing*, vol. 2024, no. 1, p. 6666395, 2024.

[64] X. Lan, H. Zhai, and Y. Wang, "A novel DOA estimation of closely spaced sources using attention mechanism with conformal arrays," *IEEE Access*, vol. 11, pp. 44010–44018, 2023.

[65] A. Barthelme and W. Utschick, "DOA estimation using neural network-based covariance matrix reconstruction," *IEEE Signal Processing Letters*, vol. 28, pp. 783–787, 2021.

[66] X. Wu, X. Yang, X. Jia, and F. Tian, "A gridless DOA estimation method based on convolutional neural network with Toeplitz prior," *IEEE Signal Processing Letters*, vol. 29, pp. 1247–1251, 2022.

[67] J. P. Merkofer, G. Revach, N. Shlezinger, T. Routtenberg, and R. J. Van Sloun, "DA-MUSIC: Data-driven DOA estimation via deep augmented MUSIC algorithm," *IEEE Transactions on Vehicular Technology*, 2023.

[68] J. Ji, W. Mao, F. Xi, and S. Chen, "TransMUSIC: A transformer-aided subspace method for DOA estimation with low-resolution adcs," in *ICASSP 2024-*

2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 8576–8580, IEEE, 2024.

[69] M. Wax and T. Kailath, "Detection of signals by information theoretic criteria," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 33, no. 2, pp. 387–392, 1985.

[70] F. Izedi, M. Karimi, and M. Derakhtian, "Joint DOA estimation and source number detection for arrays with arbitrary geometry," *Signal Processing*, vol. 140, pp. 149–160, 2017.

[71] H. Akaike, "A new look at the statistical model identification," *IEEE Transactions on Automatic Control*, vol. 19, no. 6, pp. 716–723, 1974.

[72] M. Wax and I. Ziskind, "Detection of the number of coherent signals by the MDL principle," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 8, pp. 1190–1196, 1989.

[73] G. Schwarz, "Estimating the dimension of a model," *The Annals of Statistics*, pp. 461–464, 1978.

[74] Y. Lee, C. Park, T. Kim, Y. Choi, K. Kim, D. Kim, M.-S. Lee, and D. Lee, "Source enumeration approaches using eigenvalue gaps and machine learning based threshold for direction-of-arrival estimation," *Applied Sciences*, vol. 11, no. 4, p. 1942, 2021.

[75] Z. He, A. Cichocki, S. Xie, and K. Choi, "Detecting the number of clusters in n-way probabilistic clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 11, pp. 2006–2021, 2010.

[76] W. Chen, K. M. Wong, and J. P. Reilly, "Detection of the number of signals: A predicted eigen-threshold approach," *IEEE Transactions on Signal Processing*, vol. 39, no. 5, pp. 1088–1098, 1991.

[77] O. Hu, F. Zheng, and M. Faulkner, "Detecting the number of signals using antenna array: a single threshold solution," in *ISSPA'99. Proceedings of the Fifth International Symposium on Signal Processing and its Applications (IEEE Cat. No. 99EX359)*, vol. 2, pp. 905–908, IEEE, 1999.

[78] W. Yun, L. Xiukun, C. Zhimin, H. Jian, M. Haiwei, and W. Zhentao, "Joint signal-to-noise ratio and source number estimation based on hierarchical artificial intelligence units," *Measurement Science and Technology*, vol. 29, no. 9, p. 095104, 2018.

[79] B. Sun, C. Wu, and H. Ruan, "Array diagnosis and DOA estimation for coprime array under sensor failures," *Sensors*, vol. 20, no. 9, p. 2735, 2020.

[80] B. Sun, C. Wu, J. Shi, H.-L. Ruan, and W.-Q. Ye, "Direction-of-arrival estimation under array sensor failures with ULA," *IEEE Access*, vol. 8, pp. 26445–26456, 2019.

[81] A. Setayesh, E. Yazdian, and M. Malek-Mohammadi, "Direction of arrival estimation with missing data via matrix completion," *Signal, Image and Video Processing*, vol. 13, no. 7, pp. 1451–1459, 2019.

[82] B. Jalal, O. Elnahas, and Z. Quan, "Efficient DOA estimation under partially impaired antenna array elements," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 7, pp. 7991–7996, 2022.

[83] C. Zhu, W.-Q. Wang, H. Chen, and H. C. So, "Impaired sensor diagnosis, beamforming, and DOA estimation with difference co-array processing," *IEEE Sensors Journal*, vol. 15, no. 7, pp. 3773–3780, 2015.

[84] T. Yerriswamy and S. Jagadeesha, "Fault tolerant matrix pencil method for direction of arrival estimation," *arXiv preprint arXiv:1110.1627*, 2011.

[85] J. Chen, C. Zhang, S. Fu, and J. Li, "Robust reweighted 2, 1-norm based approach for DOA estimation in mimo radar under array sensor failures," *IEEE Sensors Journal*, vol. 21, no. 24, pp. 27858–27867, 2021.

[86] U. Hamid, S. Wyne, and N. R. Butt, "Joint model-order and robust DOA estimation for underwater sensor arrays," *Sensors*, vol. 23, no. 12, p. 5731, 2023.

[87] S. Vigneshwaran, N. Sundararajan, and P. Saratchandran, "Direction of arrival (DOA) estimation under array sensor failures using a minimal resource allocation neural network," *IEEE Transactions on Antennas and Propagation*, vol. 55, no. 2, pp. 334–343, 2007.

[88] A. M. Ahmed, U. S. M. Thanthrige, A. Sezgin, and F. Gini, "Resilient sparse array radar with the aid of deep learning," in *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)*, pp. 1–5, IEEE, 2023.

[89] C. He, H. Zheng, B. Li, C. Zhou, and Z. Shi, "DOA estimation via meta-learning under array sensor failures," in *2023 IEEE International Radar Conference (RADAR)*, pp. 1–5, IEEE, 2023.

[90] M. Wang, Z. Zhang, and A. Nehorai, "Direction finding using sparse linear arrays with missing data," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3066–3070, IEEE, 2017.

[91] E. G. Larsson and P. Stoica, "High-resolution direction finding: the missing data case," *IEEE Transactions on Signal Processing*, vol. 49, no. 5, pp. 950–958, 2001.

[92] S. Kamyab, Z. Azimifar, R. Sabzi, and P. Fieguth, "Deep learning methods for inverse problems," *PeerJ Computer Science*, vol. 8, p. e951, 2022.

[93] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[94] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," in *International Conference on Machine Learning*, pp. 1243–1252, PMLR, 2017.

[95] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

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

[97] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, pp. 448–456, pmlr, 2015.

[98] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[99] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 807–814, 2010.

[100] 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.

[101] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," *arXiv preprint arXiv:1811.03378*, 2018.

[102] R. Parhi and R. D. Nowak, "The role of neural network activation functions," *IEEE Signal Processing Letters*, vol. 27, pp. 1779–1783, 2020.

[103] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 315–323, JMLR Workshop and Conference Proceedings, 2011.

[104] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[105] A. L. Maas, A. Y. Hannun, A. Y. Ng, *et al.*, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML*, vol. 30, p. 3, Atlanta, GA, 2013.

[106] A. Shrestha and A. Mahmood, "Review of deep learning algorithms and architectures," *IEEE Access*, vol. 7, pp. 53040–53065, 2019.

[107] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT Press, 2018.

[108] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[109] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization.," *Journal of Machine Learning Research*, vol. 12, no. 7, 2011.

[110] G. Hinton, N. Srivastava, and K. Swersky, "Neural networks for machine learning," *Coursera, video lectures*, vol. 264, no. 1, pp. 2146–2153, 2012.

[111] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[112] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[113] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

APPENDICES

A Network Input Visualizations

For visualizing the patterns in the covariance coarray $\hat{c}$, Algorithm 1 is applied on true covariance matrices generated from the received signals of a uniform linear array with 64 sensors. Sine and cosine of the phase component and real and imaginary components of the covariance coarray for different DOA angles are illustrated in Figures A.1, A.2, A.3 and A.4 respectively.

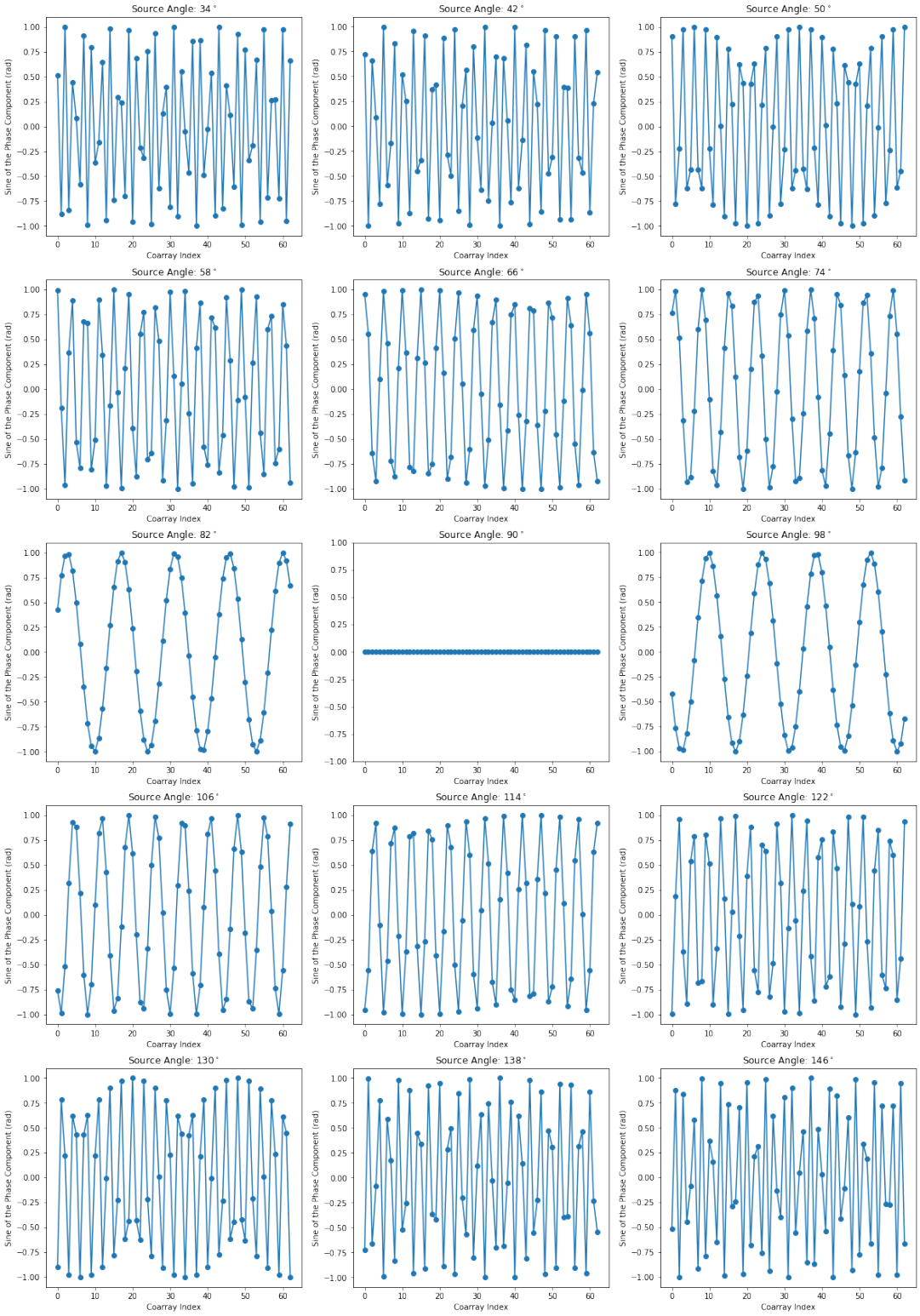

Figure A.1: Sine of the phase component of covariance coarray.

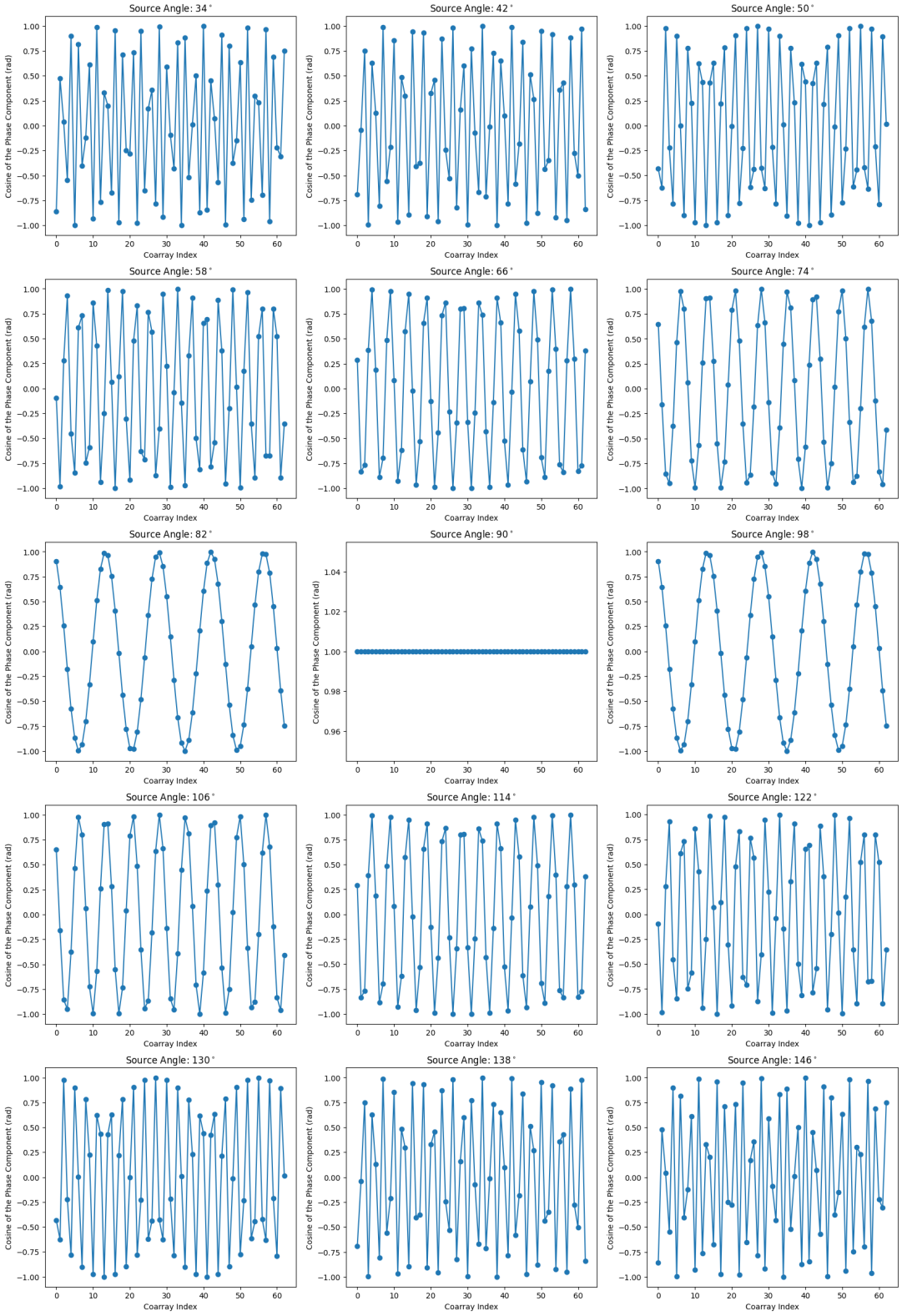

Figure A.2: Cosine of the phase component of covariance coarray.

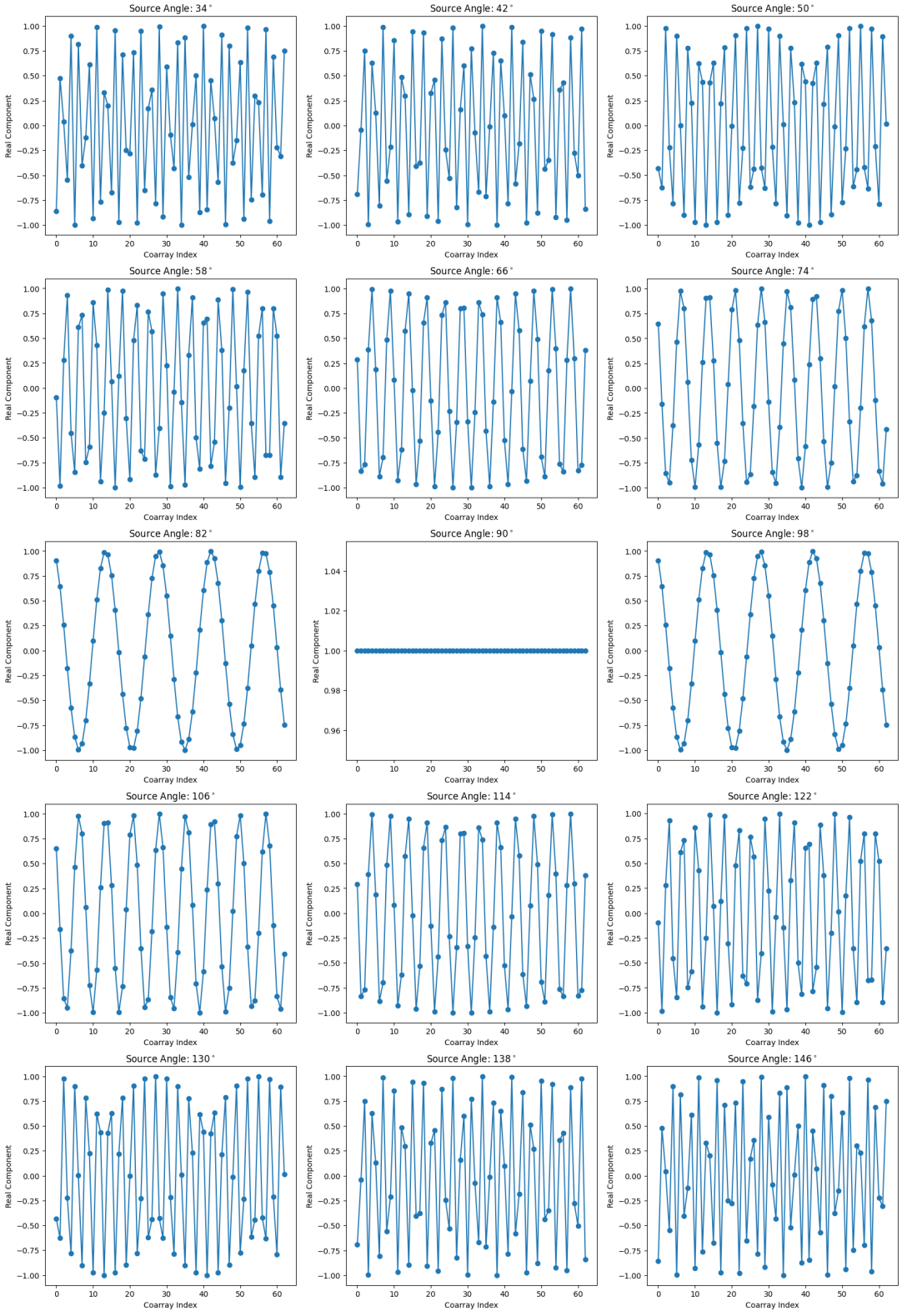

Figure A.3: Real component of covariance coarray.

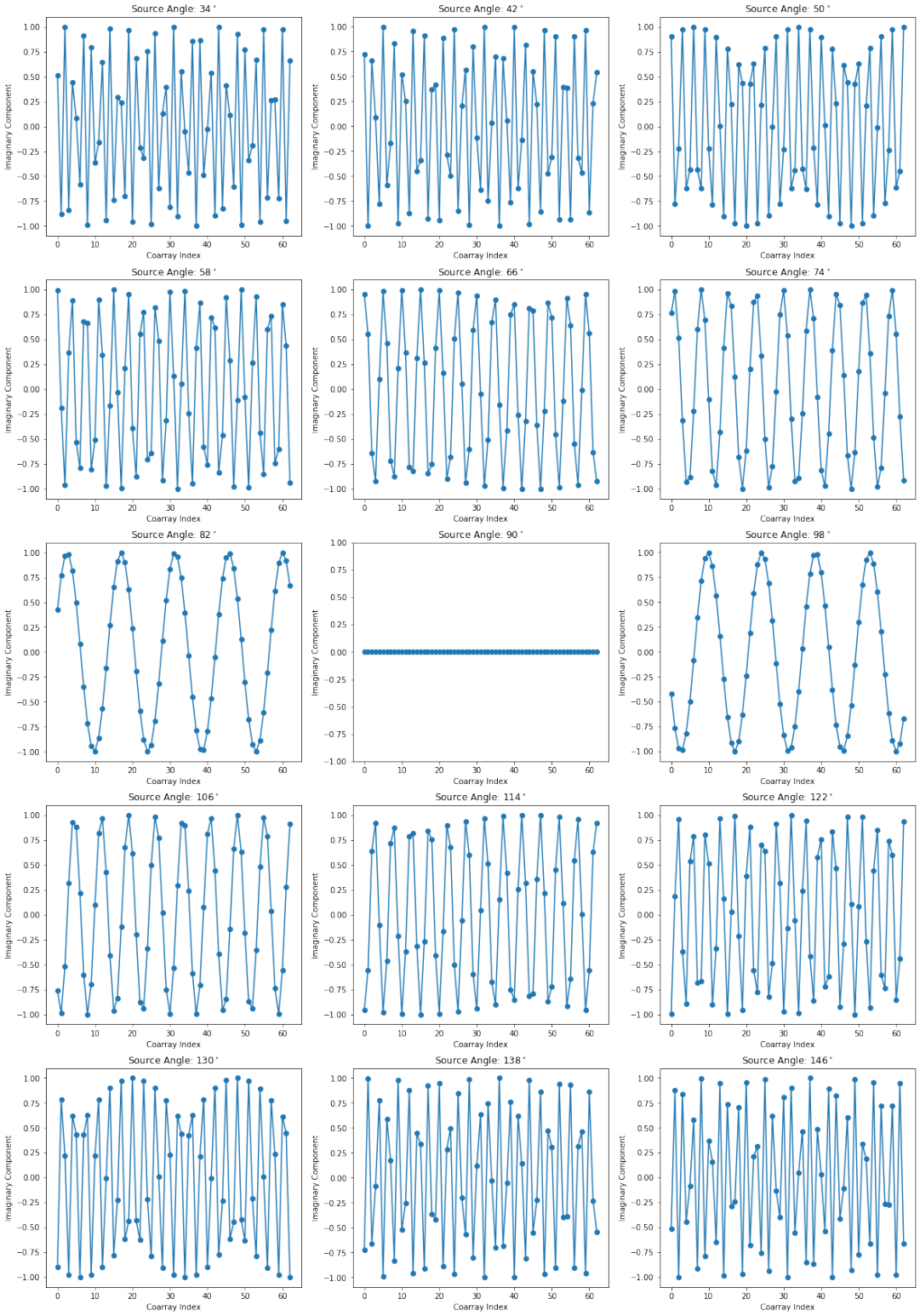

Figure A.4: Imaginary component of covariance coarray.

B Error Illustrations of Each Method for Minimum Redundant Array

Randomly 200 samples are selected from test sets which contain different SNR levels and snapshot numbers. The errors made by each method for these samples are illustrated in Figures B.1, B.2, B.3 and B.4.

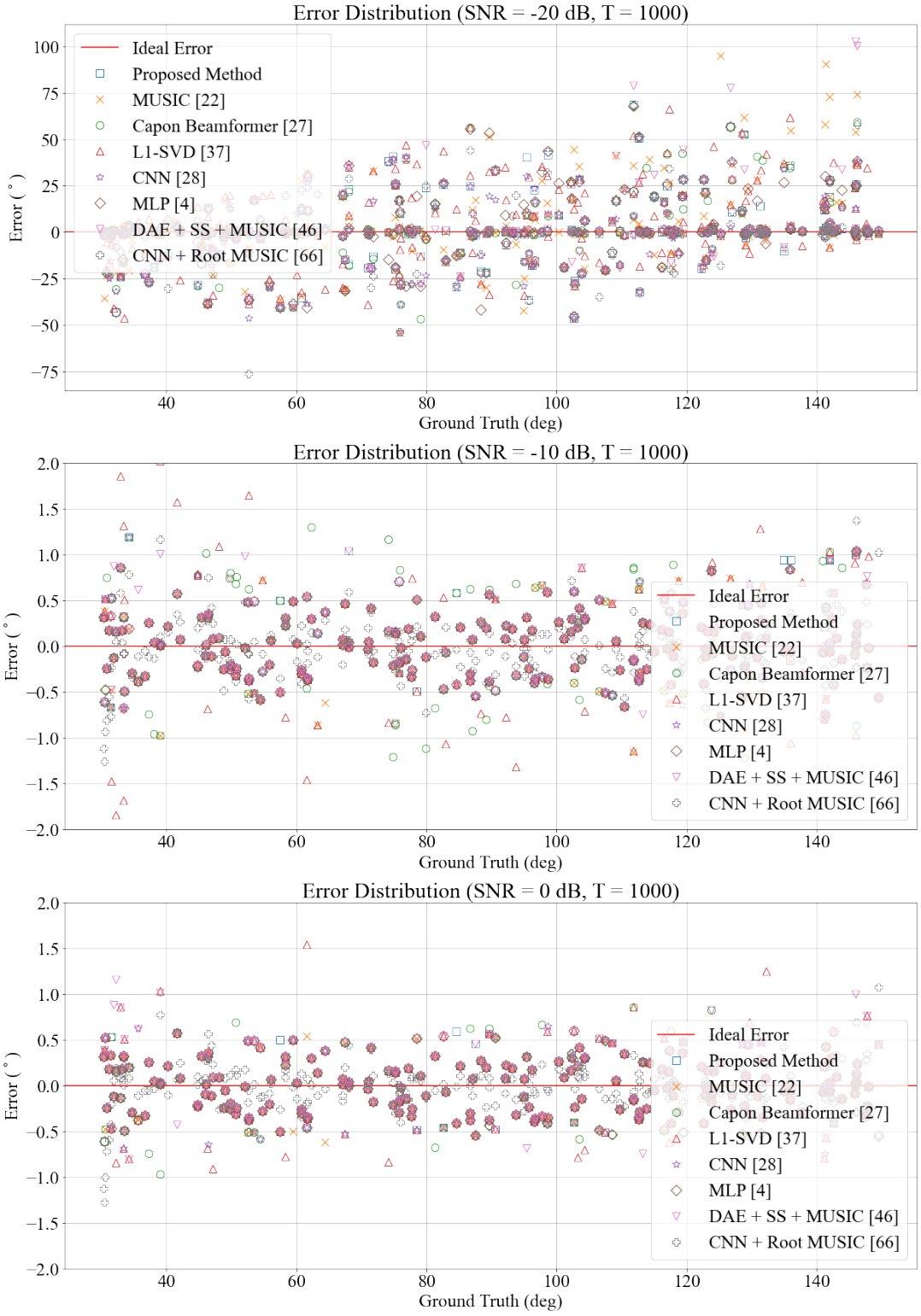

Figure B.1: Errors made by each method for different SNR levels.

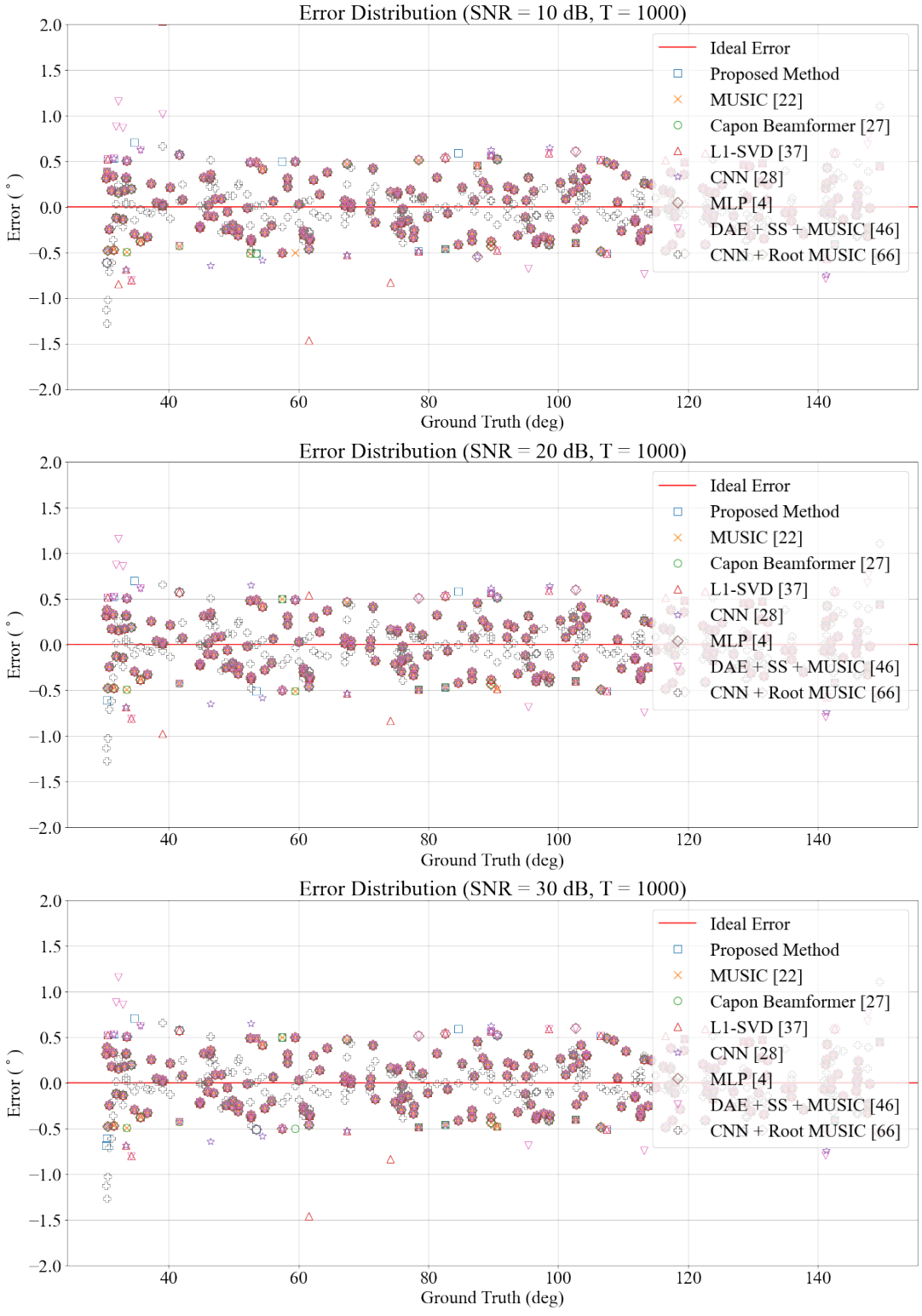

Figure B.2: Errors made by each method for different SNR levels (cont'd).

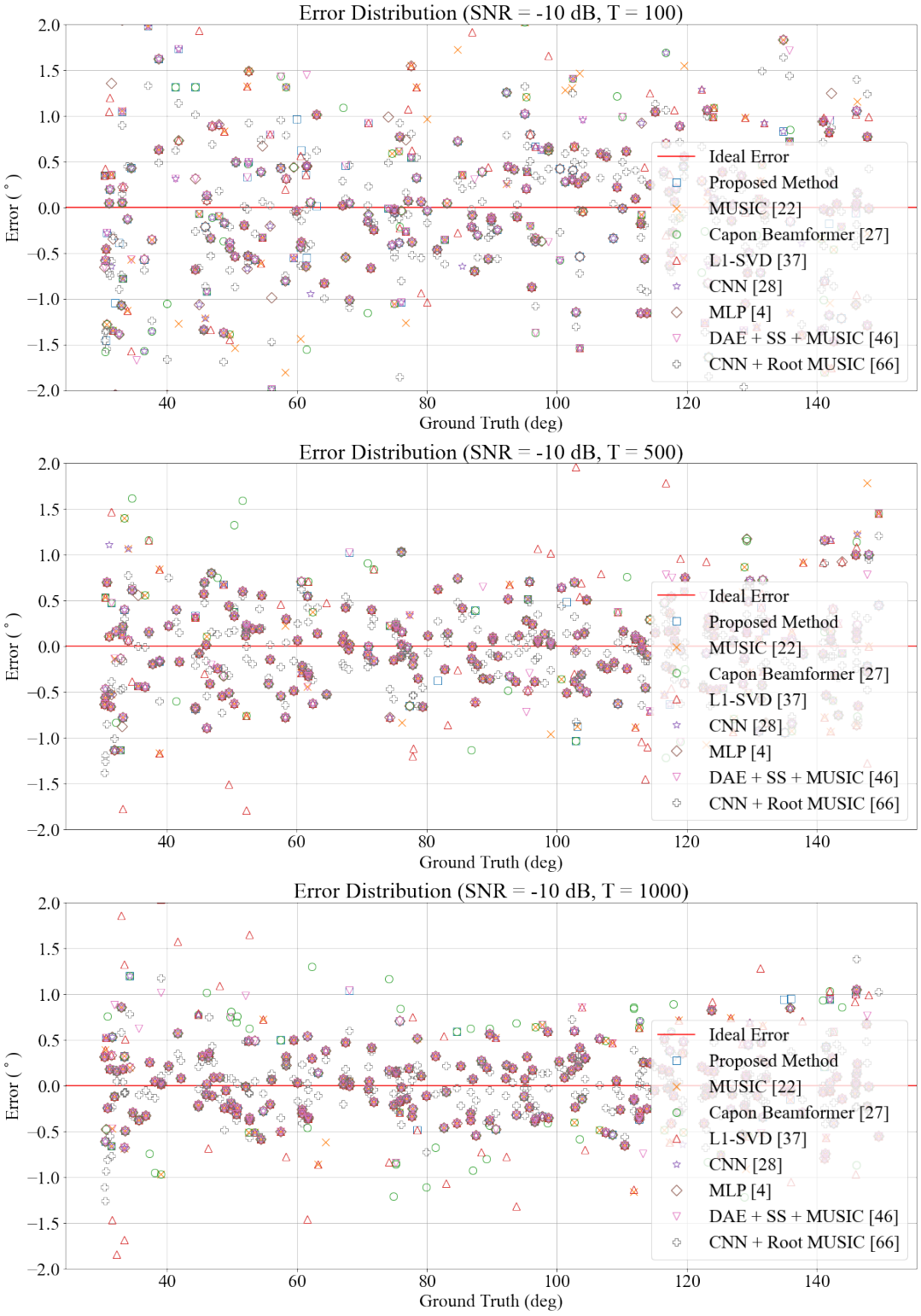

Figure B.3: Errors made by each method for different snapshot numbers.

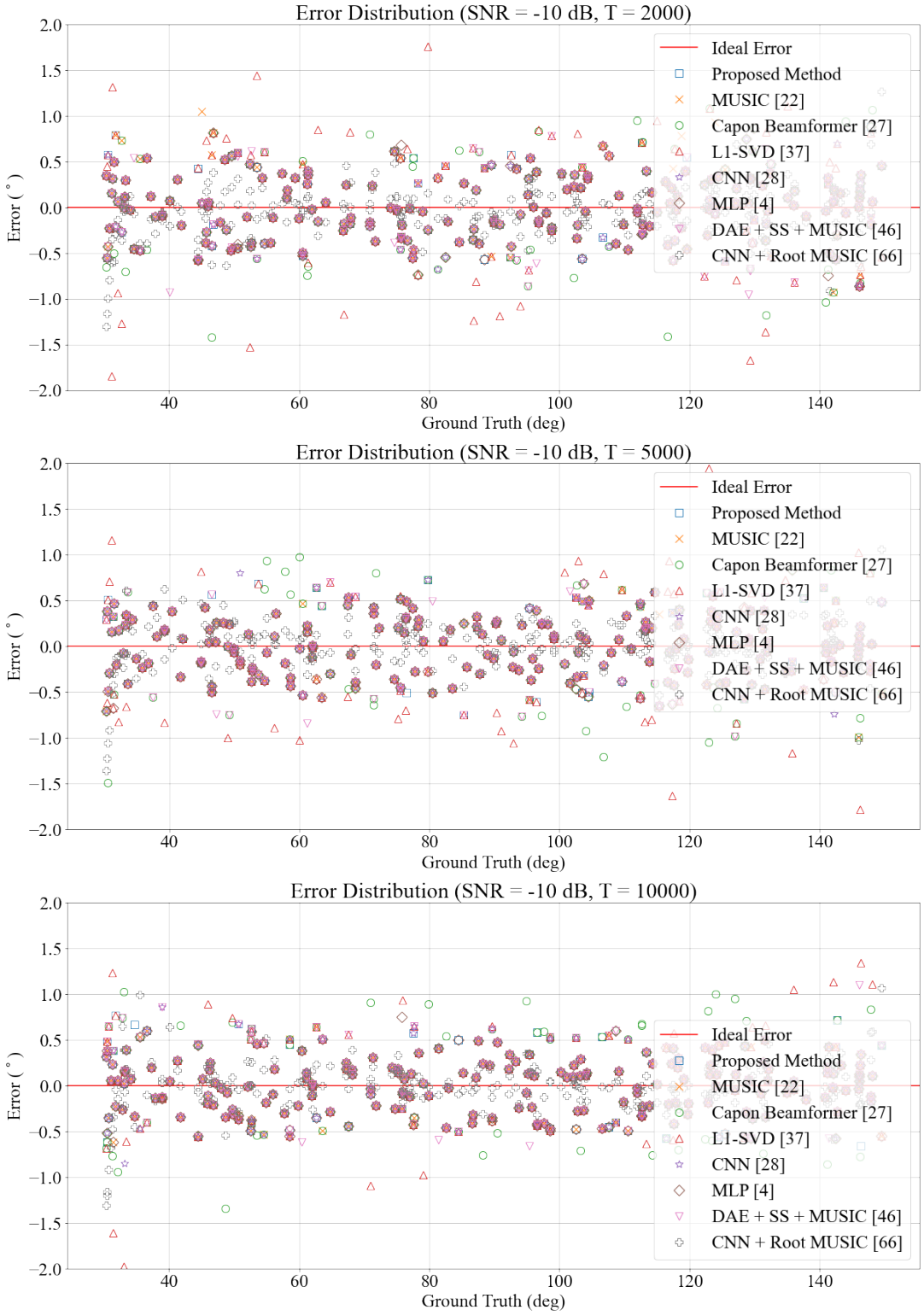

Figure B.4: Errors made by each method for different snapshot numbers (cont'd).

C Error Illustrations of Each Method for Nested Array

For different SNR levels and snapshot number, 200 samples are selected from test sets randomly. The error values for these samples are given in Figures C.1, C.2, C.3 and C.4.

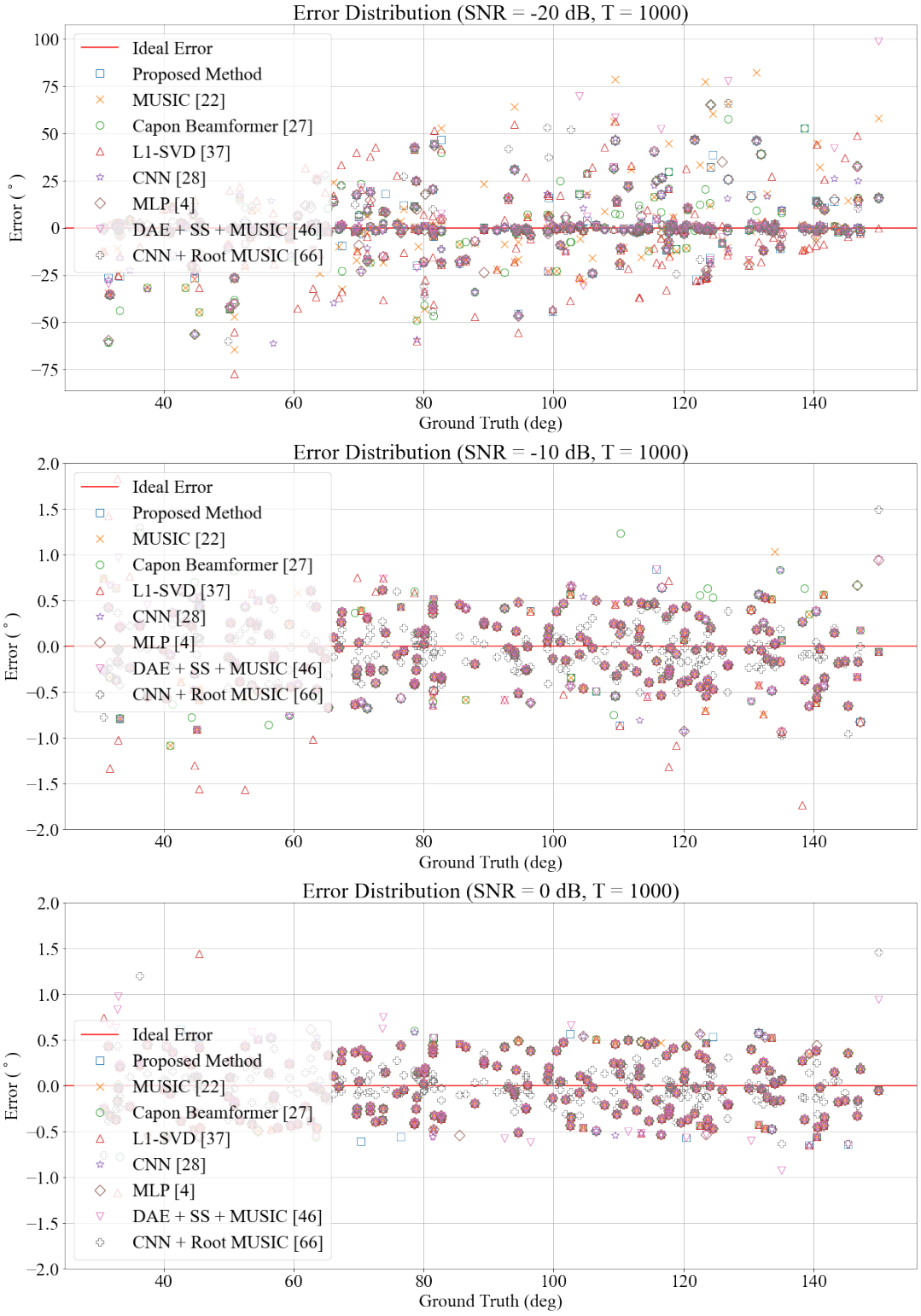

Figure C.1: Errors made by each method for different SNR levels.

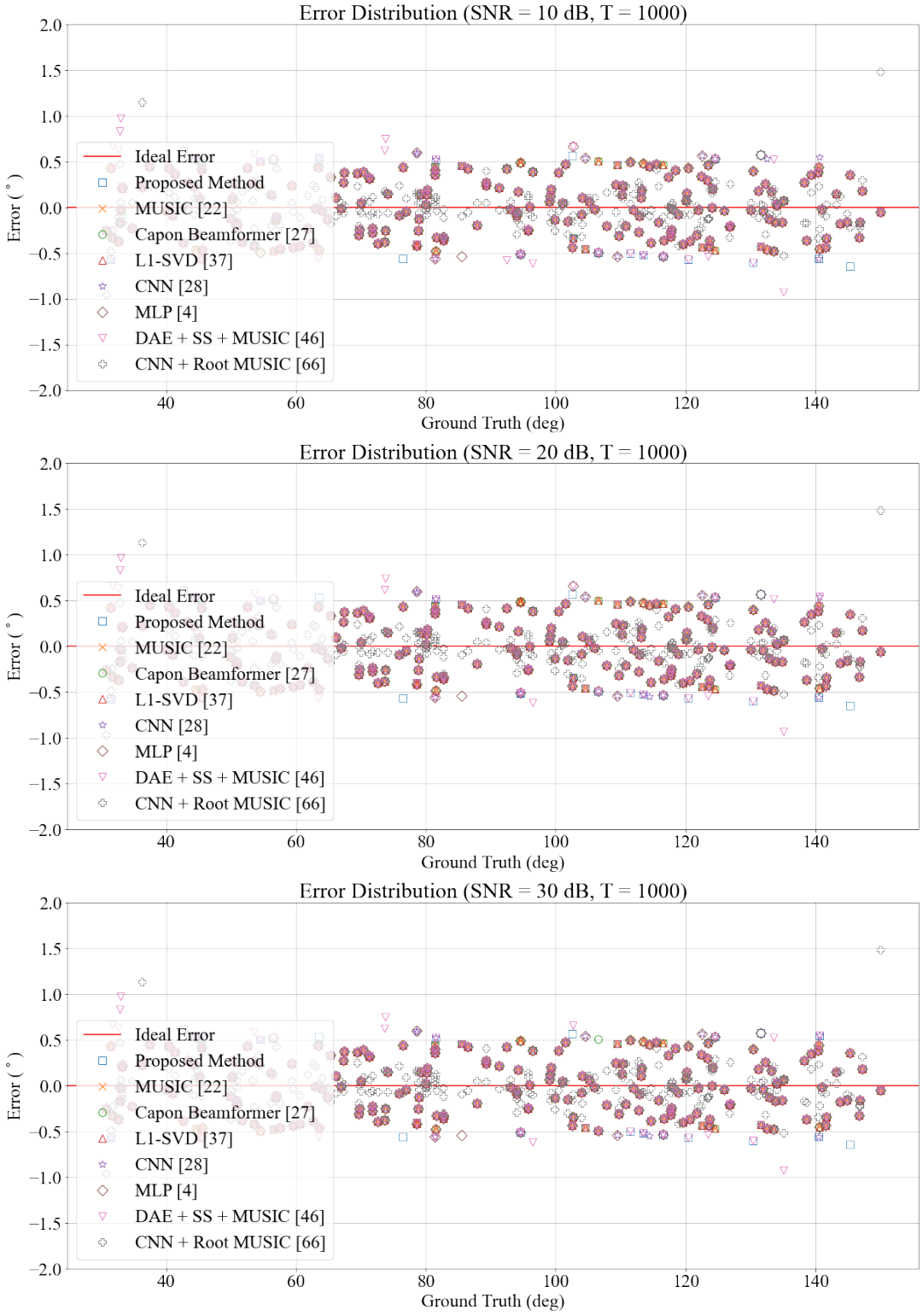

Figure C.2: Errors made by each method for different SNR levels (cont'd).

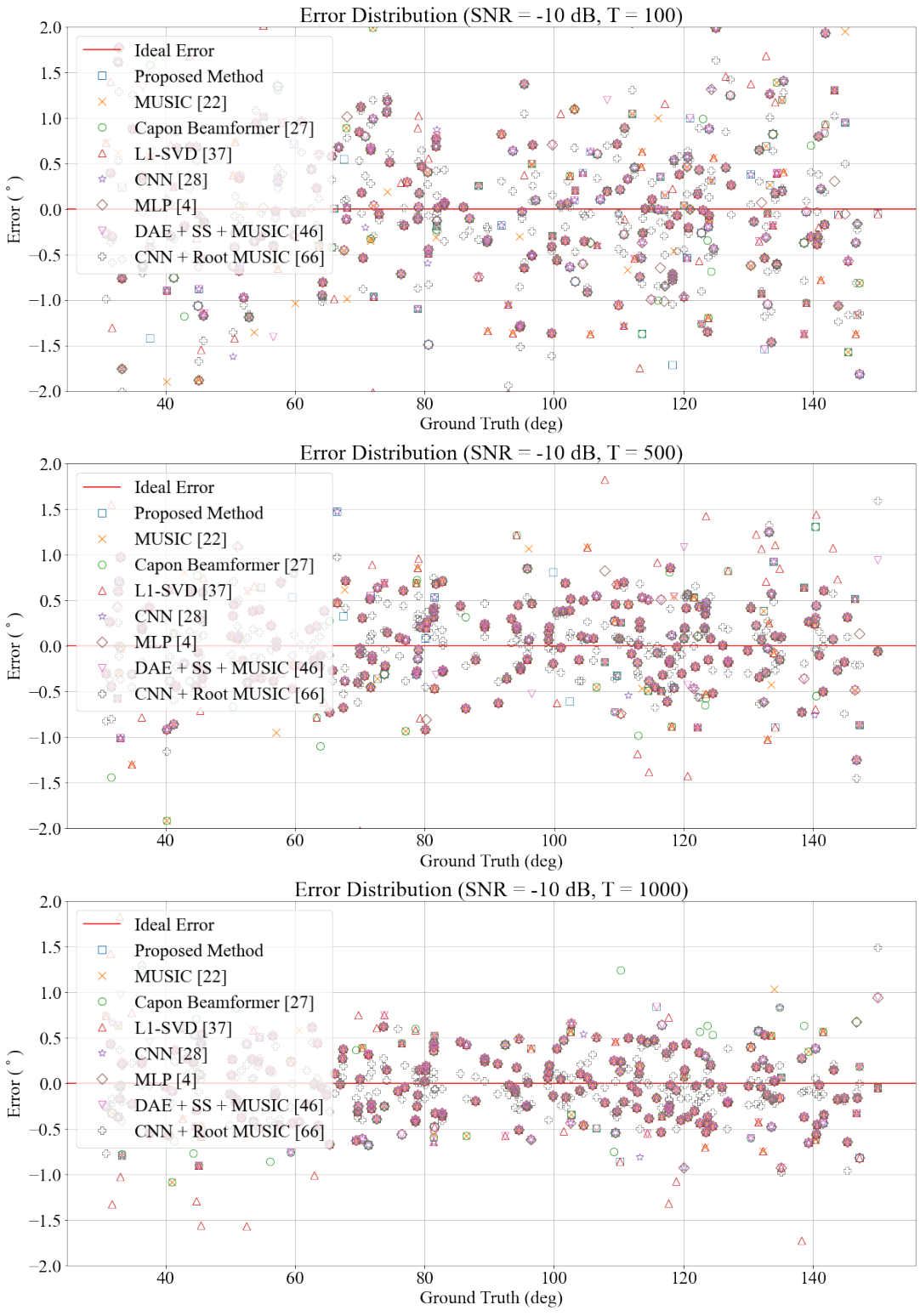

Figure C.3: Errors made by each method for different snapshot numbers.

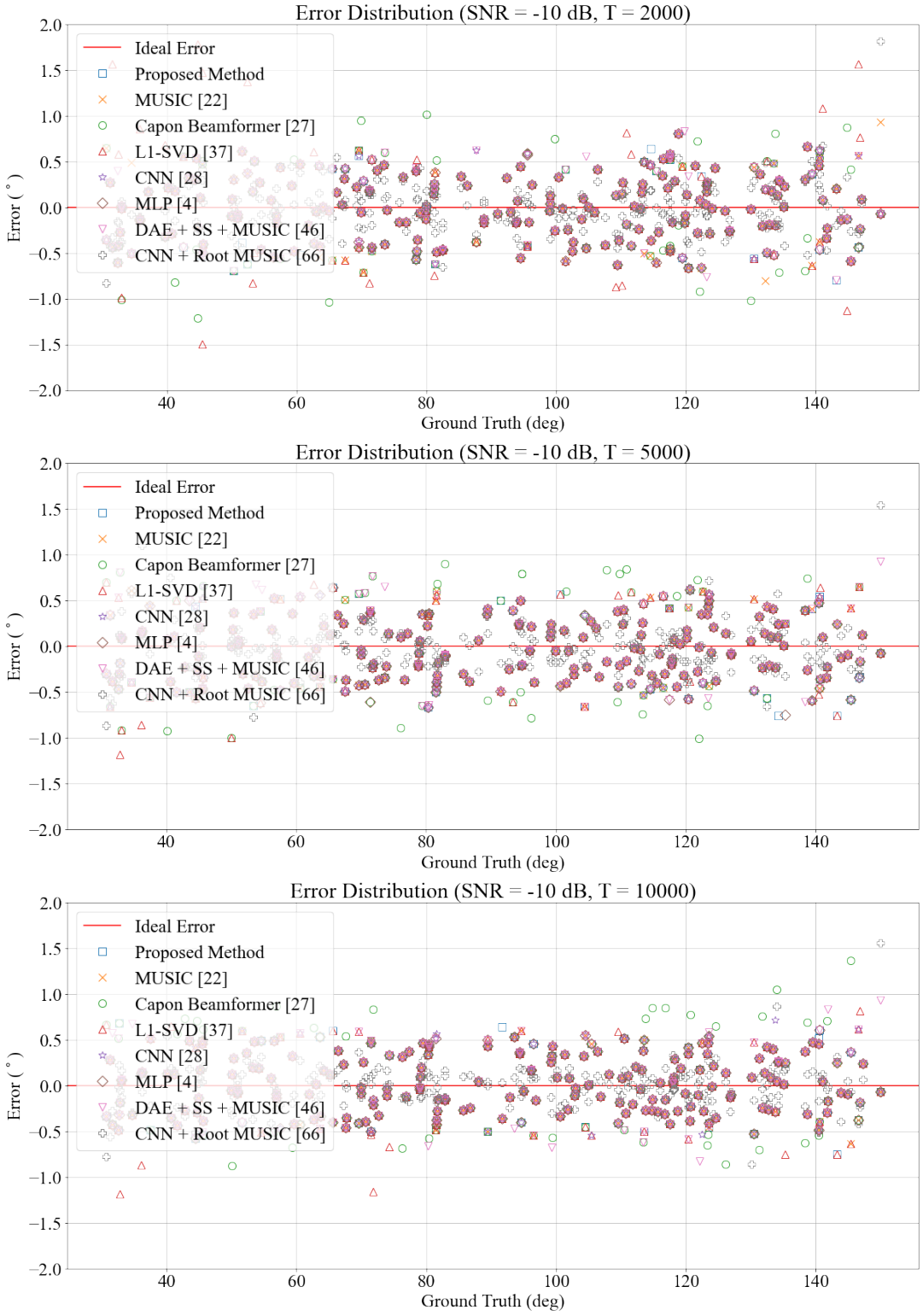

Figure C.4: Errors made by each method for different snapshot numbers (cont'd).

D Error Illustrations of Each Method for Coprime Array

Figure D.1, D.2, D.3 and D.4 show the errors of randomly selected 200 samples from test sets for different SNR levels and snapshot numbers respectively.

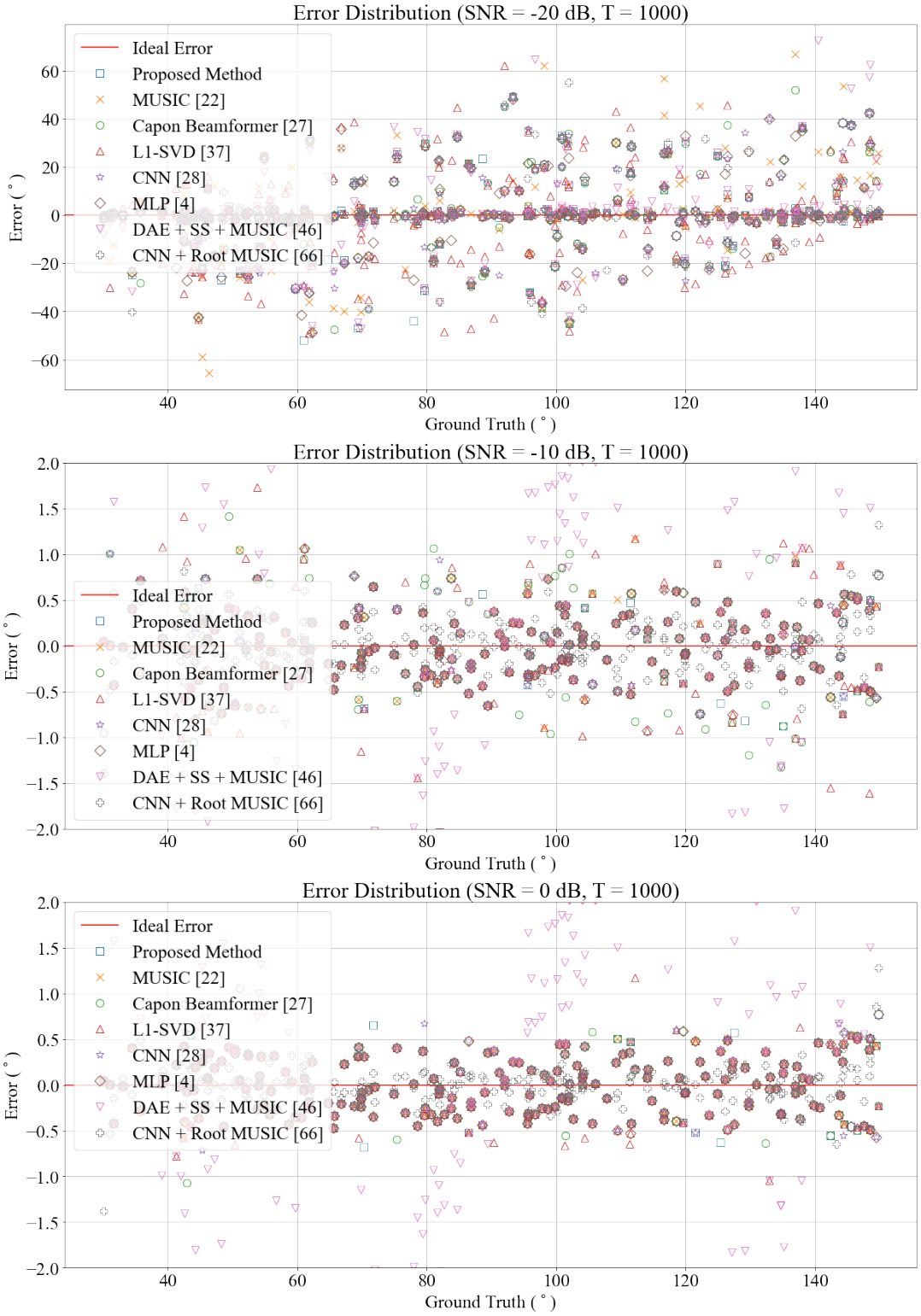

Figure D.1: Errors made by each method for different SNR levels.

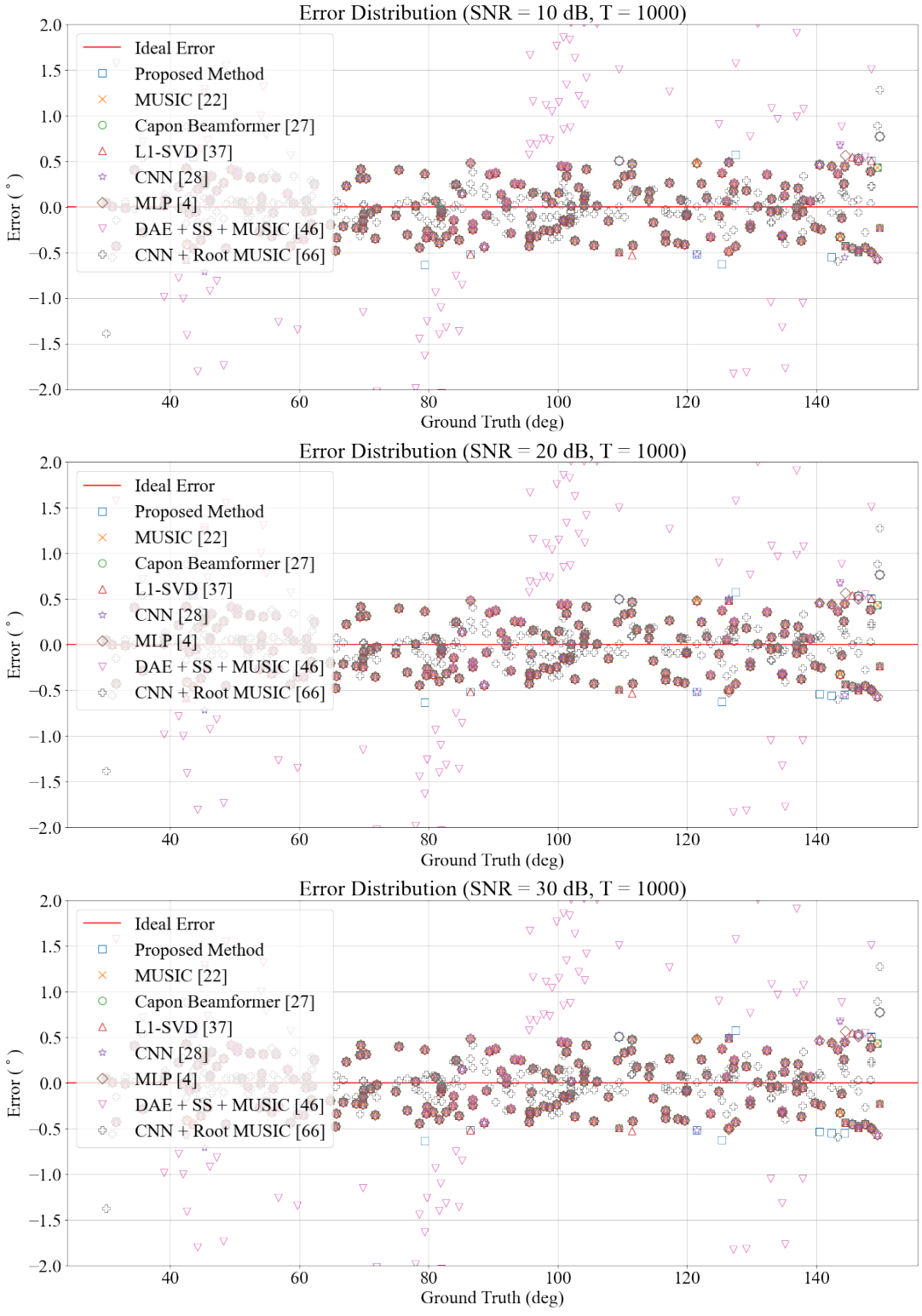

Figure D.2: Errors made by each method for different SNR levels (cont'd).

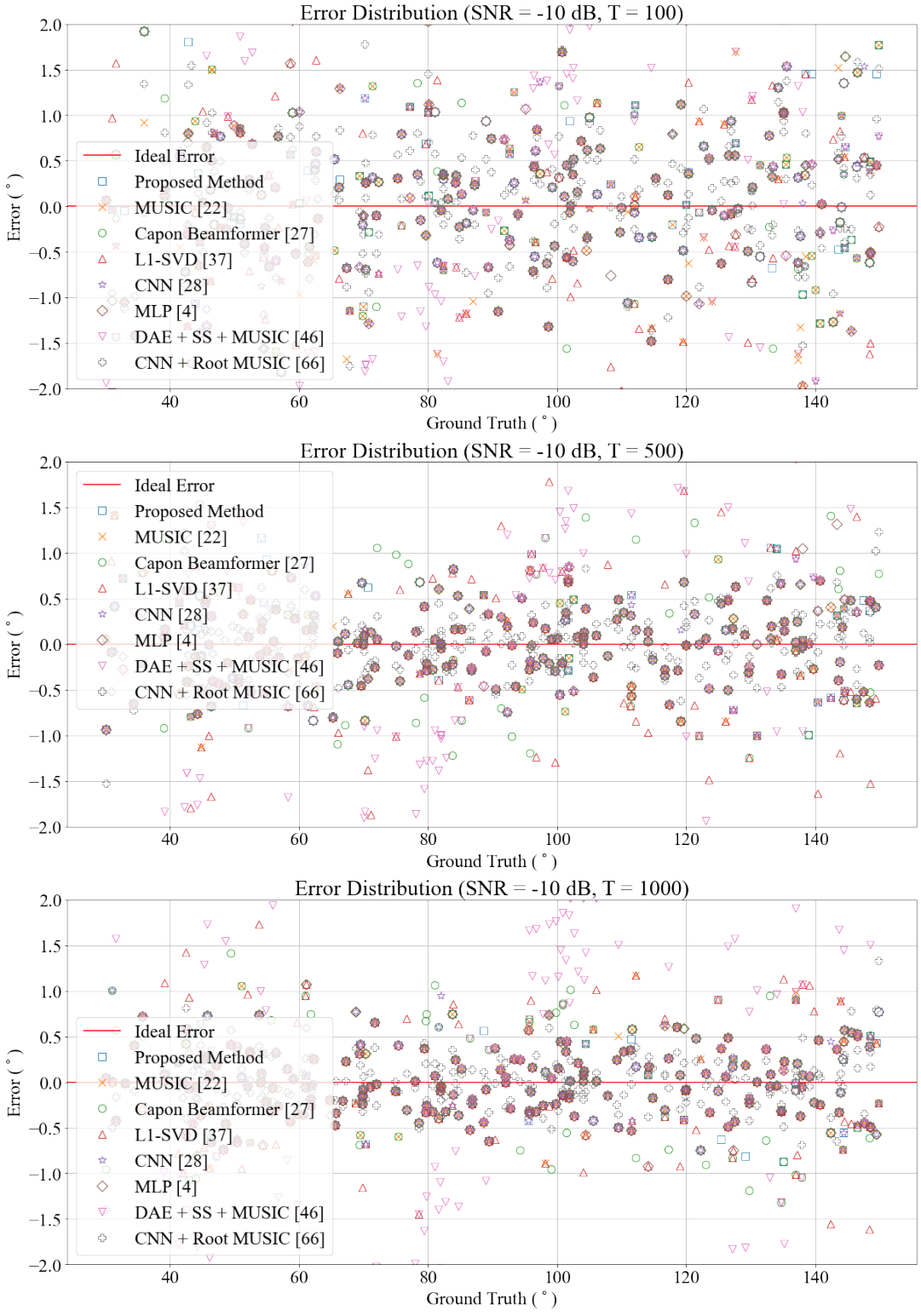

Figure D.3: Errors made by each method for different snapshot numbers.

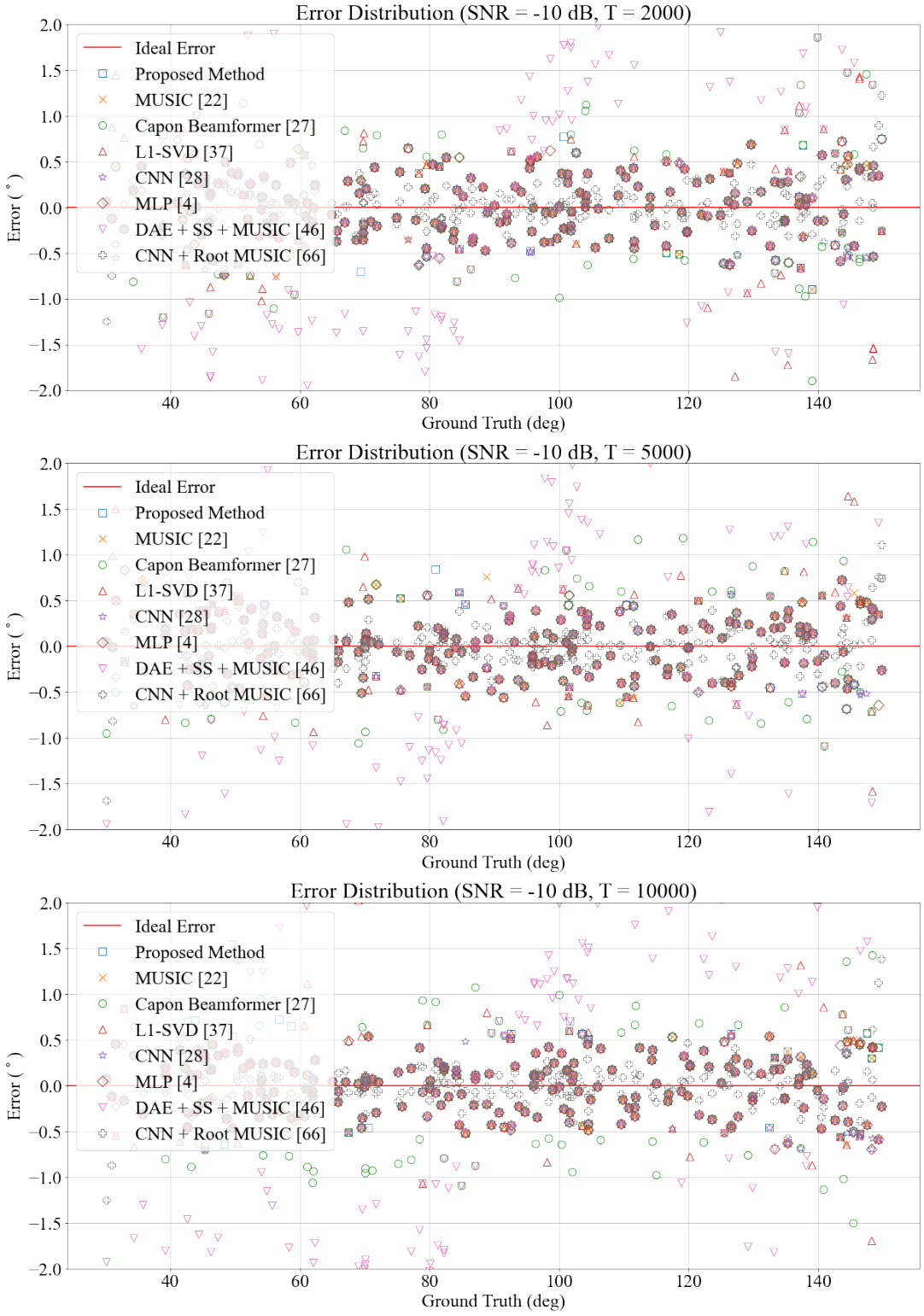

Figure D.4: Errors made by each method for different snapshot numbers (cont'd).

E Error Illustrations of Each Method for Intact Array

For randomly selected 200 samples from test set, the errors illustrated in Figures E.1, E.2, E.3 and E.4 are obtained.

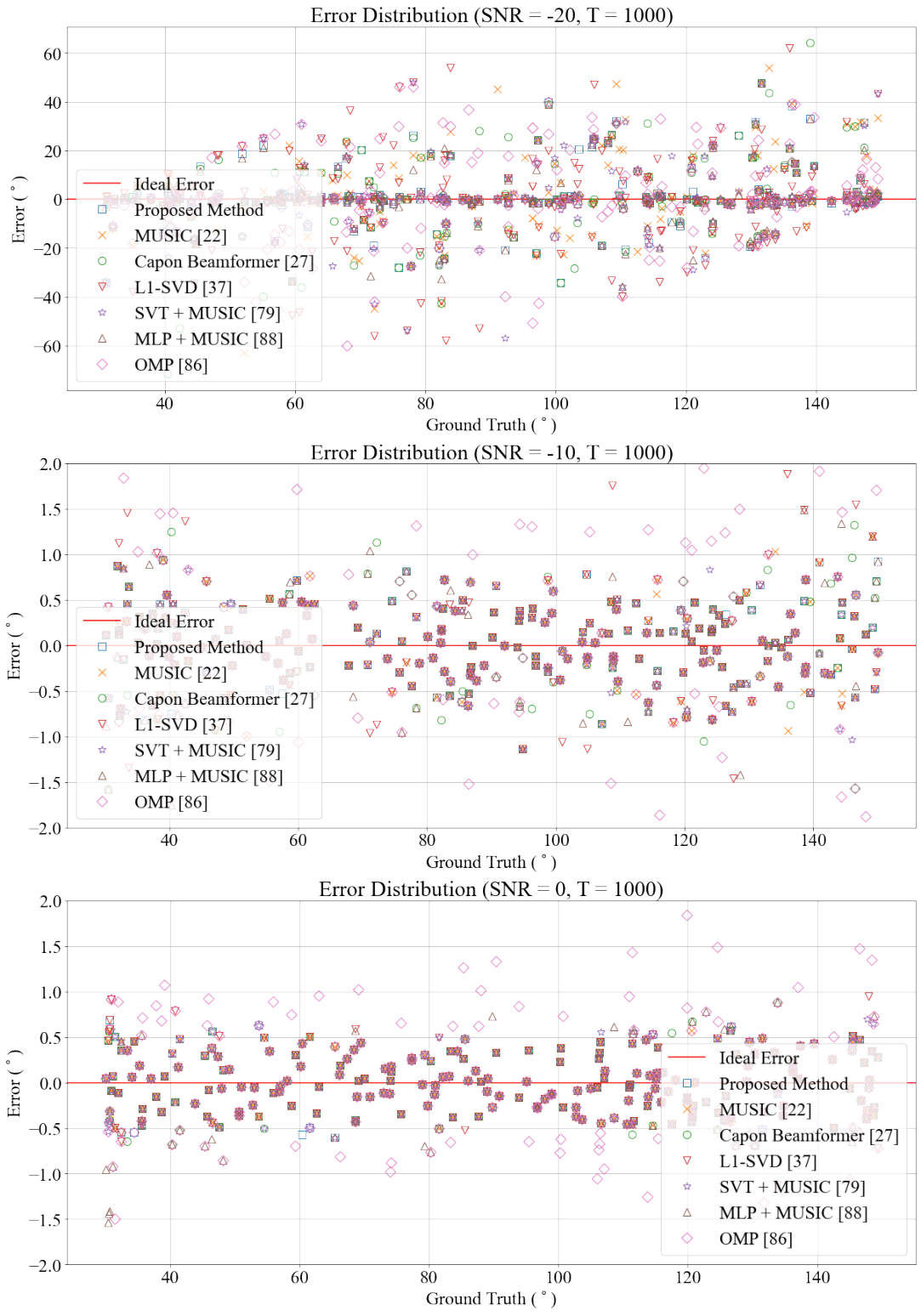

Figure E.1: Errors made by each method for different SNR levels.

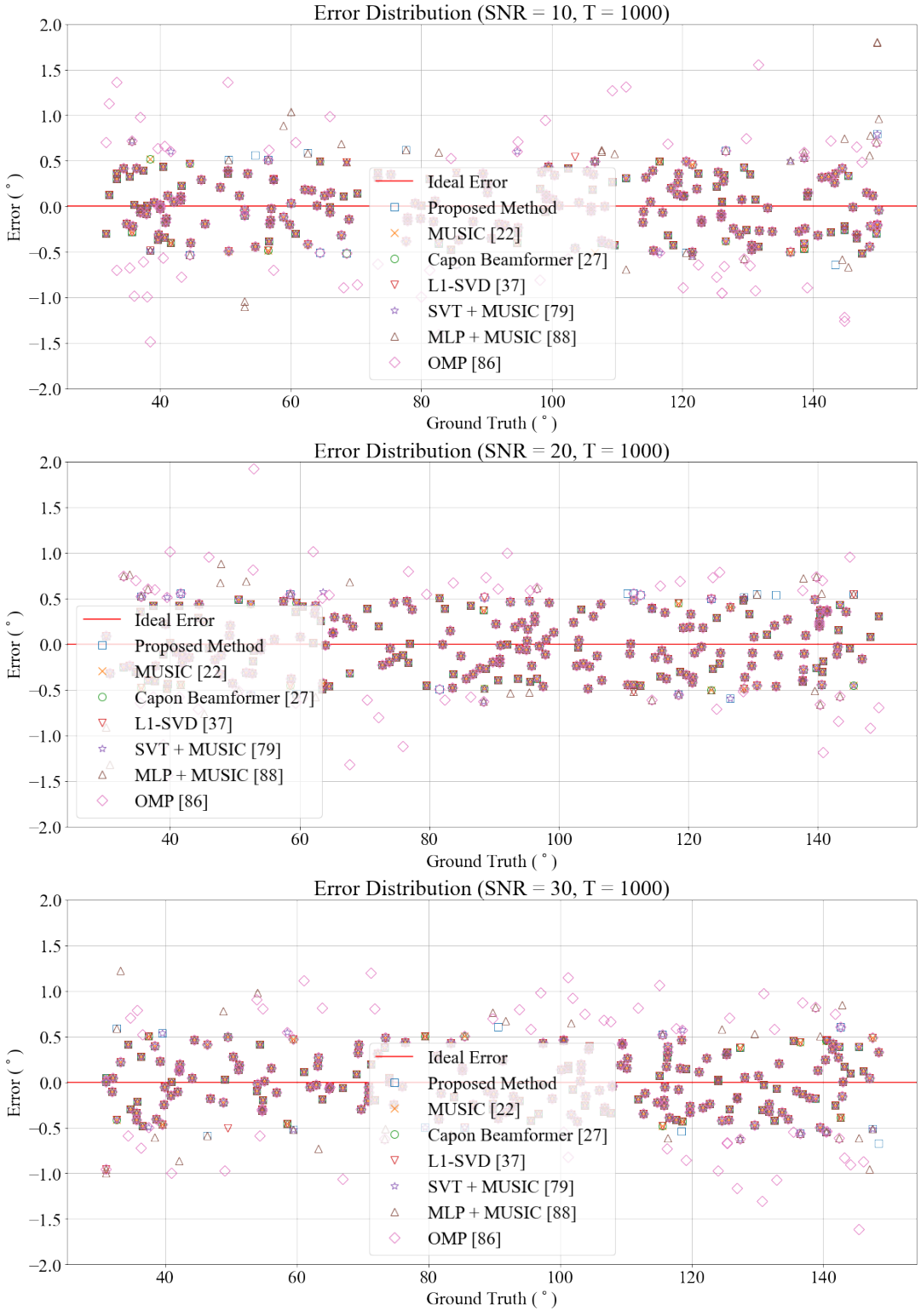

Figure E.2: Errors made by each method for different SNR levels (cont'd).

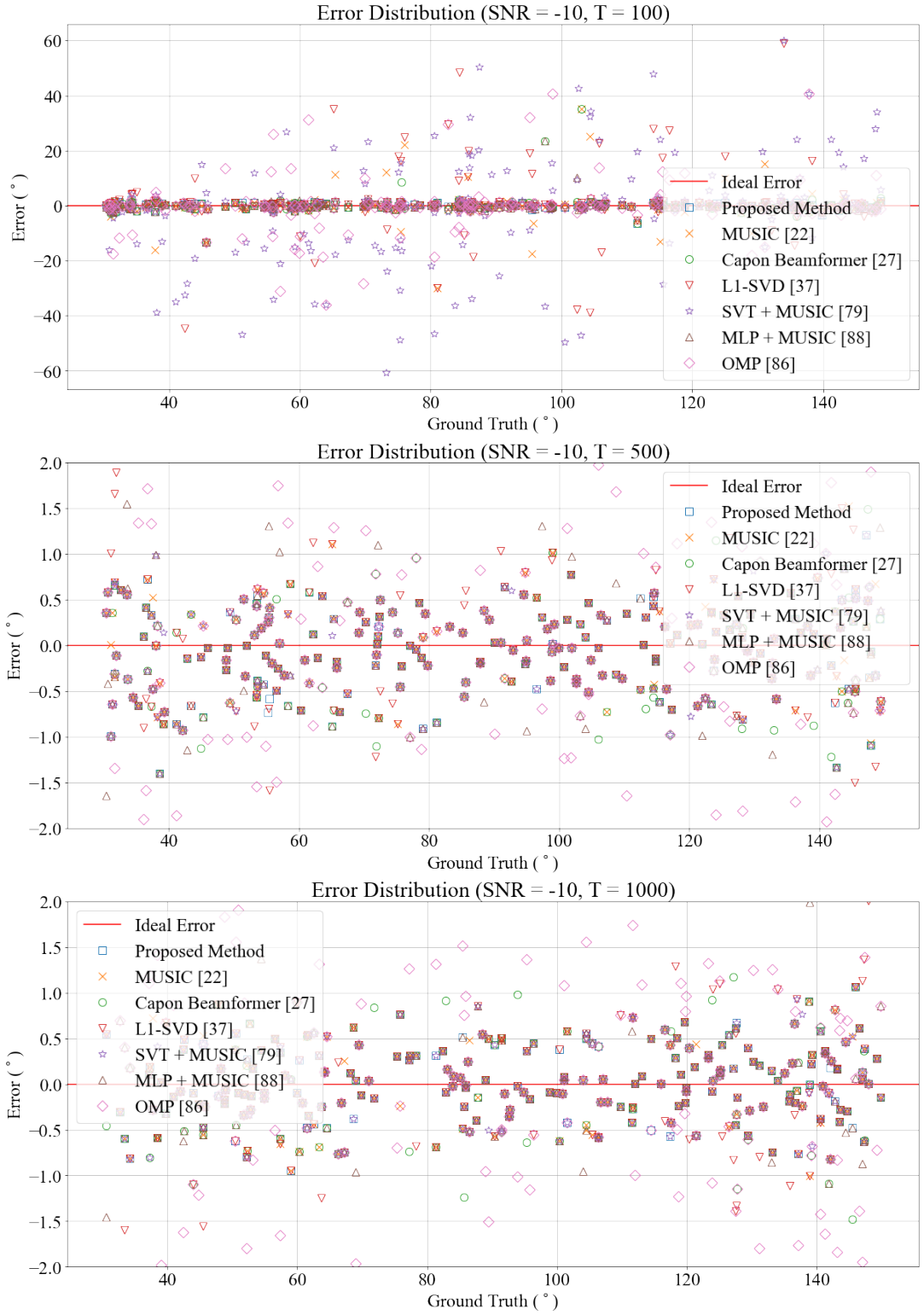

Figure E.3: Errors made by each method for different snapshot numbers.

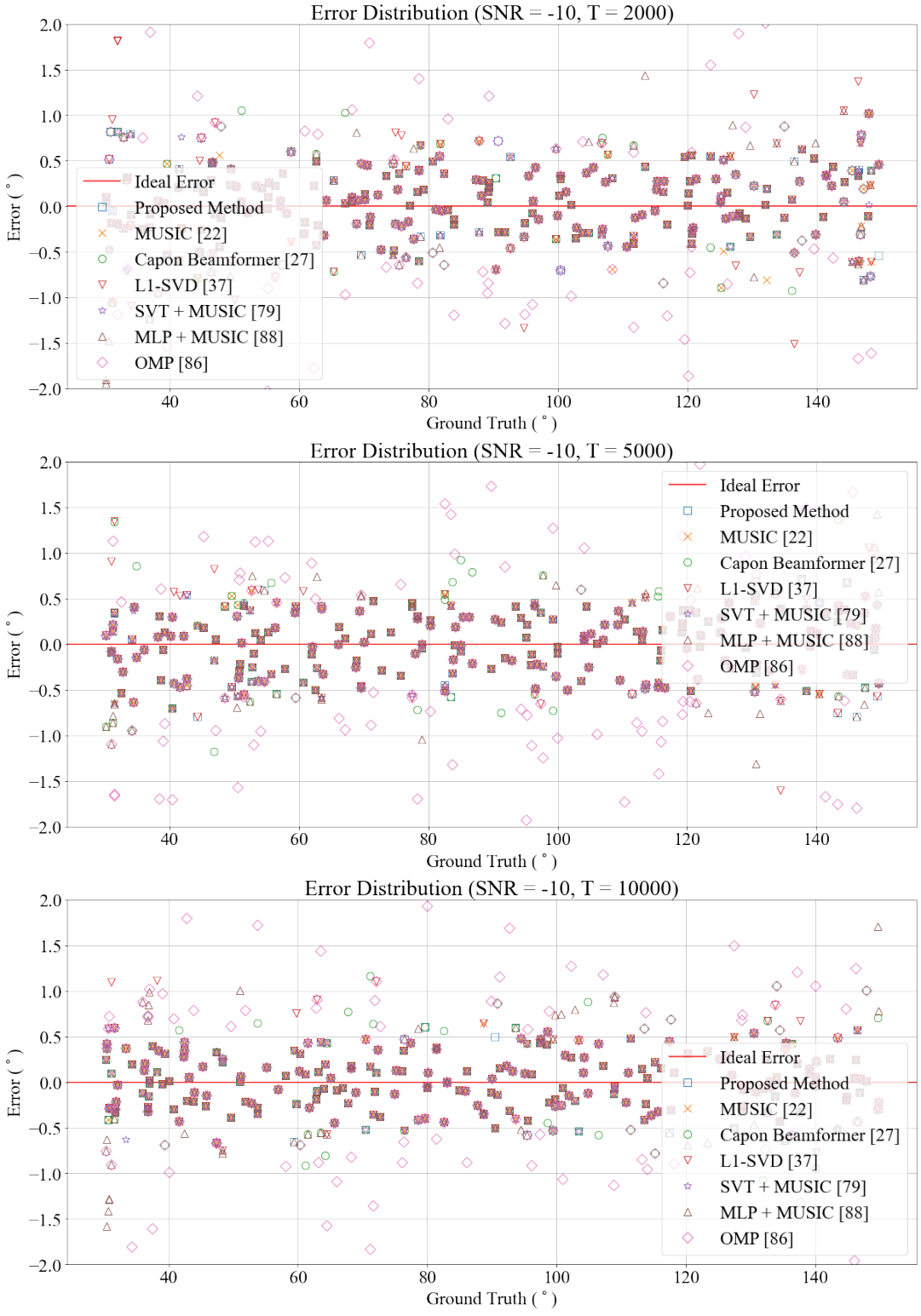

Figure E.4: Errors made by each method for different snapshot numbers (cont'd).

F Error Illustrations of Each Method for Faulty Array

Figures F.1, F.2, F.3 and F.4 illustrate the errors for randomly selected 200 samples from test set.

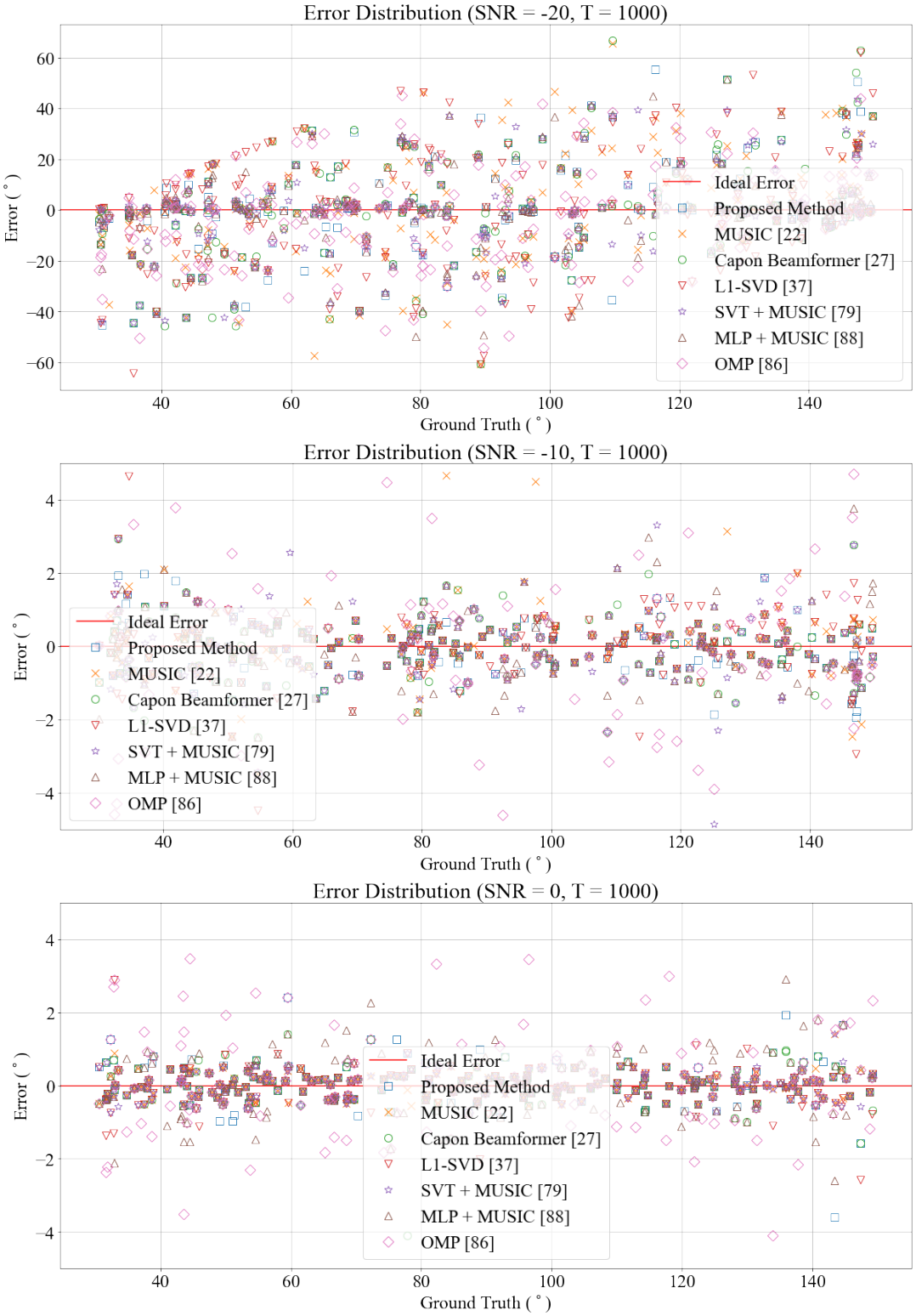

Figure F.1: Errors made by each method for different SNR levels.

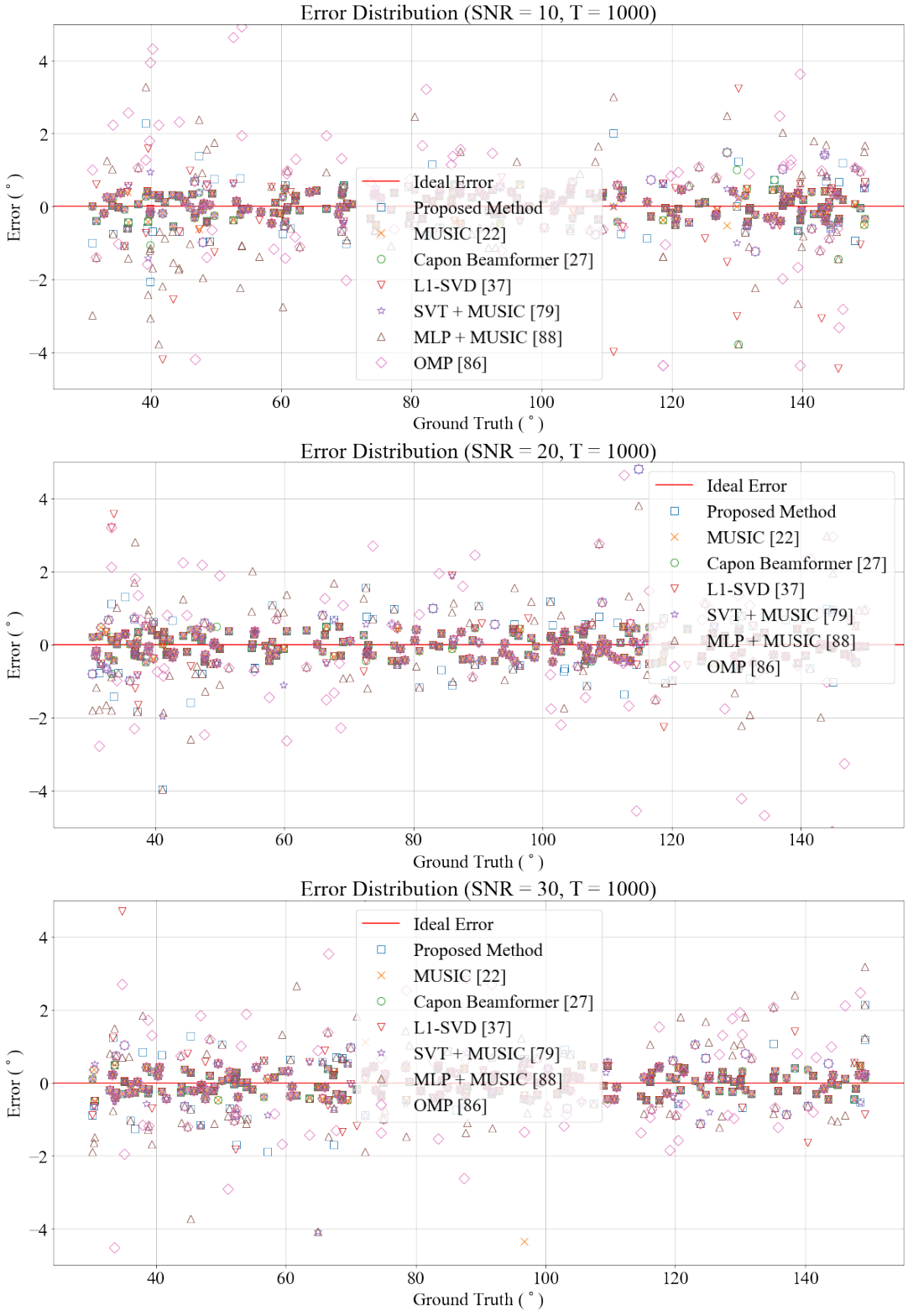

Figure F.2: Errors made by each method for different SNR levels (cont'd).

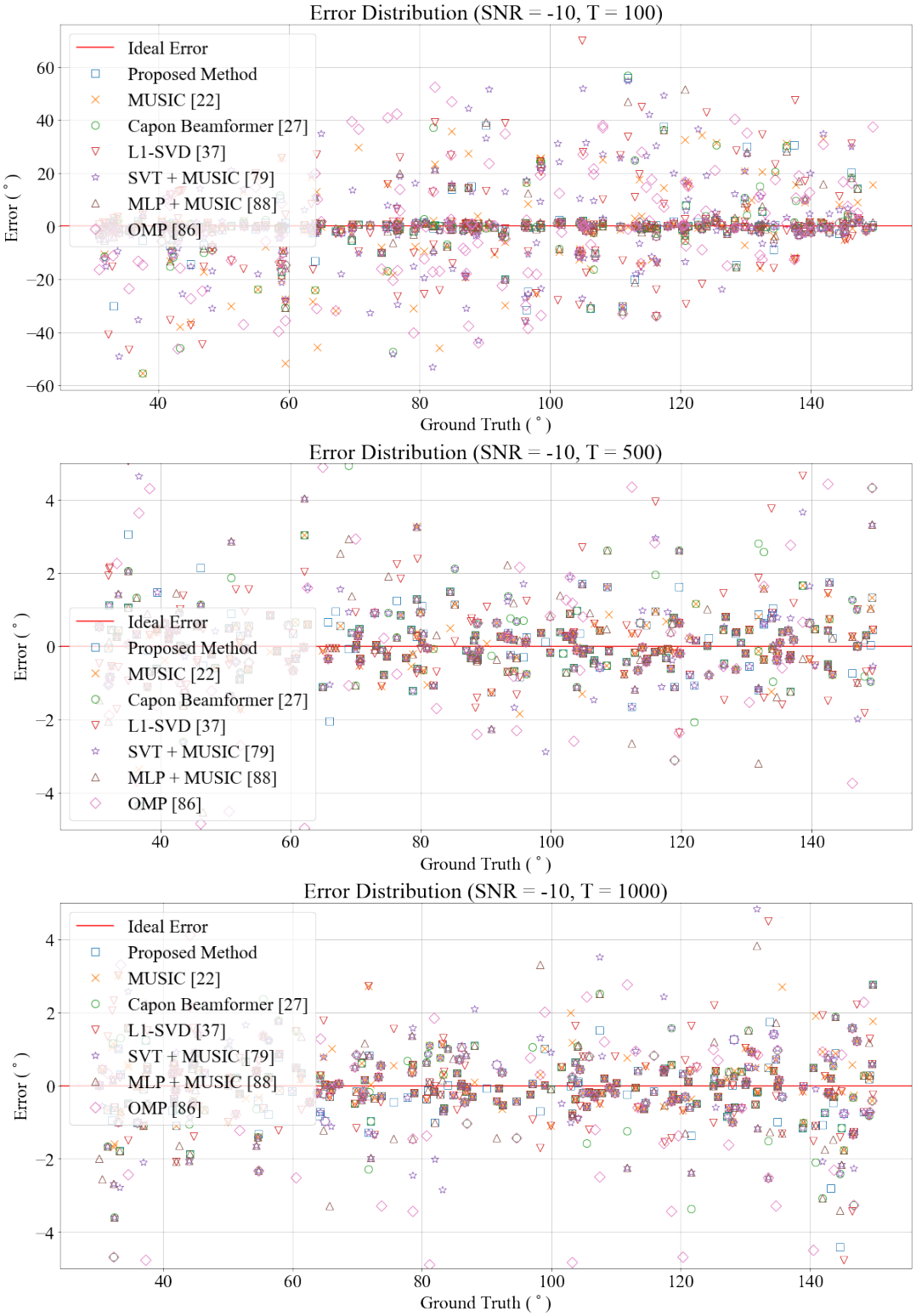

Figure F.3: Errors made by each method for different snapshot numbers.

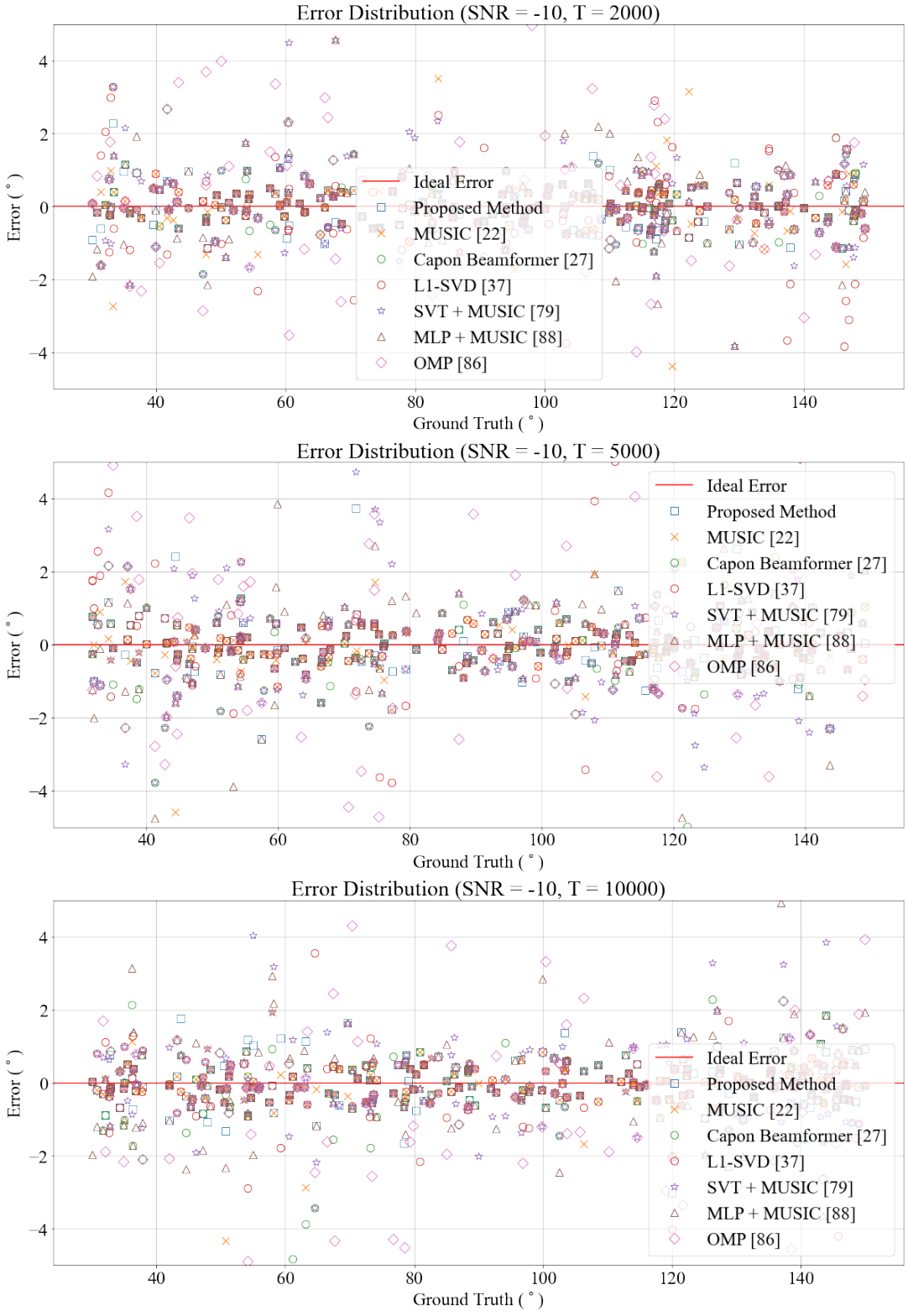

Figure F.4: Errors made by each method for different snapshot numbers (cont'd).