Enhancing Direction of Arrival Trajectory Estimation with Data-Driven Approaches

Thesis submitted in partial fulfillment
of the requirements for the degree of

*Master of Science in **Electronics and Communication Engineering** by Research*

by

Shreyas Jaiswal
2019702016
shreyas.jaiswal@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500032, INDIA
June, 2024

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "**Enhancing Direction of Arrival Trajectory Estimation with Data-Driven Approaches**" by Shreyas Jaiswal, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: Dr. Santosh Nannuru.

To my family

Acknowledgments

I feel great satisfaction in summarizing my comprehensive research journey within the pages of this thesis. Starting with little knowledge, I went from no clear research direction to actively contributing to writing and publishing scholarly papers. My time in research at IIITH is a truly unforgettable chapter in my life. I begin by extending my heartfelt gratitude to **Dr. Santosh Nannuru** for his invaluable guidance as my research adviser. In my first semester, I enrolled in his course of adaptive signal processing and was particularly impressed with his teaching style and the guidance he provided. Upon joining as a research assistant under his mentorship, I initially encountered challenges in doing research. However, his generous support, guidance, and dedicated time, enabling me to delve into and comprehend new topics more profoundly. Working under his guidance has greatly helped me better understand how to look at the bigger picture of my work and improved my skills in assessing and enhancing my research. He stressed the importance of being thorough in writing and experiments. When we publish a paper, we're essentially trying to convince others that our ideas are valid. This requires clear, concise writing, and strong supporting evidence.

Nextly, I would like to express my gratitude to my labmates: Karthik, Sumanth, Souradeep, Ruchi, Ayush, Nikhil, Jigyasu, Ihita, Madhuri Madam, Rishi, and Nilesh. Thank you for your support, collaboration, and the camaraderie we've shared. I also want to thank the juniors in our lab: Anjani, Sasanka, Aman, Vijay, Vaibhav, Deepti, Ankit, Animesh, Aditya, and Ashwini. Your energy and fresh perspectives have made a significant difference. I look forward to continuing to work with all of you. Sports have always brought me great joy and valuable lessons. It's not just about playing the games; it's about the people who make each moment on the court or field truly memorable. I would like to thank everyone with whom I've played and learned different sports. First, I want to express my gratitude to those with whom I played badminton: Chetan, Arpit, Sidd, Ashish, Prateek, Vaibhaw, Sasanka, Aarush, Ajay, and Shiva. You all turned every game into a fantastic experience, and I cherish the memories we've created together. Next, for tennis and basketball, I'd like to thank PK Sir, Vinod PK Sir, Manish Sir, Sankarshan, Manisha, Shubodh, Zeel, Harshit, Saideep, Yash, Abhinaba and Pradeep. Playing Lawn Tennis, Basketball, and Table Tennis with you during the COVID period was a refreshing

escape, providing much-needed energy and camaraderie. Thank you all for making sports such a rewarding.

Most **importantly**, I would like to express my deepest gratitude to my **family**, for their unwavering support and encouragement. Without their love, patience, and guidance, none of this would have been possible. They have been my source of strength through every challenge, and I am profoundly grateful for everything they've done for me. This research has taught me vital lessons about the importance of consistency, sincerity, perseverance, and passion in the pursuit of goals.

Abstract

Localization and tracking play a critical role in various fields, ranging from wireless communications and robotics to surveillance and autonomous vehicles. In the context of signal processing, localization typically involves measuring signals received from multiple sensors to estimate the source's position. Tracking, on the other hand, extends localization to a temporal context. In this thesis, we explore the data-driven approaches to enhance the direction of arrival (DOA) trajectory estimates. DOA estimation methods involve processing multi-channel sensor array data to determine the directions from which signals originate. Classical methods have two key limitations: first, they rely on analytical properties of observed signals which do not generalize well to non-ideal conditions; second they employ block-level processing, assuming static DOA within each block. In this thesis, the first challenge is resolved with a data-driven approachs, while the second is addressed by operating in DOA-trajectory space instead of DOA space. So, instead of estimating individual DOA, the approach focuses on estimating DOA trajectories.

We proposed two data-driven approaches: one grid-based and the other grid-free. In the grid-based approach, we develop a fully convolutional neural network (FCNN) inspired by the computer vison techniques of image translation. The FCNN processes the 2D low resolution spectrum as input and outputs a refined 2D spectrum. The input spectrum typically has broad peaks, while the transformed spectrum has sharp peaks, which allows precise identification of DOA trajectory parameters. In another study, we develop a data-driven approach that is both *grid-free* and *re-useable*. This addresses two issues: first, the limitation of parameter estimation on predefined grids, which affects resolution; and second, the problem of estimating all sources at once, which can make traditional methods dependent on the number of sources. A deep complex network is proposed that directly processes the complex sensor array data, with outputs comprising of complex signal amplitude (per snapshot) and trajectory parameters for a single source. We obtain a residual by removing contribution of the identified source from the input data and this residual is again fed back into the network to identify the next source making it *re-useble* and independent of the number of sources to be estimated. These proposed methods are rigorously evaluated through extensive experiments and analysis, demonstrating

their advantages over existing approaches. These findings have the potential to impact a variety of applications, with room for further improvement.

Contents

Chapter Page

1 Introduction . . . 1
1.1 Related works . . . 1
1.2 Motivation . . . 3
1.3 Contributions . . . 4
1.4 Organization of Thesis . . . 5

2 Background . . . 6
2.1 Signal Model . . . 6
2.2 Baseline Methods . . . 8
2.2.1 TL-CBF . . . 8
2.2.2 TL-SBL . . . 9
2.3 Summary . . . 9

3 Deep architecture for DOA trajectory localization . . . 11
3.1 Network Architecture . . . 11
3.1.1 Encoder . . . 12
3.1.2 Decoder . . . 13
3.2 Training Objectives: Target Map Generation and Loss Functions . . . 15
3.2.1 Target Map . . . 15
3.2.2 Loss functions . . . 16
3.3 Experiments and Results . . . 17
3.3.1 Data generation . . . 17
3.3.2 Implementation details . . . 19
3.3.3 Performance metrics . . . 19
3.3.4 Results and discussions . . . 19
3.4 Summary . . . 30

4 DOA Trajectory Localization using Complex Deep Nets : One Source at a time . . . 31
4.1 Why complex-valued neural networks (CVNNs)? . . . 32
4.2 Network Architecture . . . 33
4.2.1 Shared Feature Encoder . . . 34
4.2.2 Decoder . . . 36
4.2.3 Loss functions . . . 37
4.3 Experiments and Results . . . 38

4.3.1 Data generation 38
4.3.2 Implementation details 40
4.3.3 Performance metrics 40
4.3.4 Results and discussions 41
4.4 Summary 58

5 Conclusions 59
5.1 Future Work 60

Bibliography 62

List of Figures

Figure Page

2.1 TL-CBF Spectrum. Two moving sources of equal amplitudes with source trajectories $(-60, -2)$ and $(40, 2)$ [black cross] at 10 dB SNR. Detected peaks are shown by red circle. 8
2.2 TL-CBF Spectrum. Two moving sources of equal amplitudes with source trajectories $(10, 2)$ and $(20, 4)$ [black cross]. Detected peaks are shown by red circle. 9
2.3 TL-SBL Spectrum. Two moving sources of equal amplitudes with source trajectories $(-60, -2)$ and $(40, 2)$ [white cross]. Detected peaks are shown by red circle. 10

3.1 U-Net architecture with 2 skip connections, annotated with the number of channels above each block. 12
3.2 RMSE-based target map: there are two moving sources of equal amplitudes with DOA trajectories parameterized by $(-60, -2)$ and $(40, 2)$ [white cross]. . 16
3.3 Accuracy (a) and average RMSE over detected sources (b) of various algorithms as SNR is changed. 22
3.4 Accuracy (a) and average RMSE over detected sources (b) of various algorithms as RMSE threshold (η) varies. 22
3.5 Spectrum from various algorithms for -3 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(-4, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively. 23
3.6 Spectrum from various algorithms for 0 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(-4, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively. 24
3.7 Spectrum from various algorithms for 3 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(-4, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively. 24
3.8 Spectrum from various algorithms for 6 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(-4, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively. 25
3.9 Accuracy (a) and average RMSE (b) vs ϕ_0. Two source trajectories are fixed to be $(60, 2)$ and $(5, 1)$ while the third source trajectory is $(\phi_0, -1)$. 25

3.10 Accuracy (a) and average RMSE over detected sources (b) of various algorithms as RMSE threshold (η) varies. 26

3.11 Spectrum from various algorithms for 5 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(-28, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively. 27

3.12 Spectrum from various algorithms for 5 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(-7, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively. 27

3.13 Spectrum from various algorithms for 5 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(-1, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively. 28

3.14 Spectrum from various algorithms for 5 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(20, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively. 28

3.15 Spectrum from various algorithms for 5 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(29, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively. 29

3.16 Spectrum from various algorithms for 5 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(41, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively. 29

4.1 Deep Nets architecture. Number above the blocks represents the number of channels for convoluitional block and hidden size of LSTM. Additionally, the numbers beside each block signify one of the dimensions that decreases as the data progresses through each layer. Various components of the network are – shared feature encoder (shaded pink), amplitude decoder (shaded blue), and trajectory decoder (shaded peach). 35

4.2 An illustration of the complex convolution operator. 36

4.3 [**TestData-1**] Accuracy (a) and average RMSE over detected sources (b) of various algorithms as SNR is changed. 42

4.4 [**TestData-1**] Accuracy (a) and average RMSE over detected (b) sources of various algorithms as RMSE threshold (η) varies. 43

4.5 Amplitude relative error of the two-source scenarios as SNR (a) and ϕ_0 (b) varies for the proposed method. 44

4.6 [**TestData-1**] TL-CBF spectrums: (a) original signal at 1 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (60.4, 3.5) and (50.5, -2.5) [red cross]. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. . . 44

4.7 [**TestData-1**] TL-CBF spectrums: (a) original signal at 5 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (60.4, 3.5) and (50.5, -2.5) [red cross]. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. . . 44

4.8 [**TestData-1**] TL-CBF spectrums: (a) original signal at 10 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (60.4, 3.5) and (50.5, -2.5) [red cross]. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. . . 45

4.9 [**TestData-1**] TL-CBF spectrums: (a) original signal at 15 dB SNR, (b) after removing first source, (c) after removing second source at 15 dB SNR. Source trajectories are (60.4, 3.5) and (50.5, -2.5) [red cross]. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. 45

4.10 [**TestData-2**] Accuracy (a) and average RMSE over detected sources (b) of various algorithms vs ϕ_0. One source trajectory is fixed to be $(20.4, 3.5)$, while the second source trajectory is $(\phi_0, -2.5)$. The parameter ϕ_0 varies with a step size of 6, ranging from -60 to 54. 46

4.11 [**TestData-2**] Accuracy (a) and average RMSE over detected sources (b) of various algorithms as RMSE threshold varies (η). One source trajectory is fixed to be $(20.4, 3.5)$, while the second source trajectory is $(\phi_0, -2.5)$. The parameter ϕ_0 varies with a step size of 6, ranging from -60 to 54. 47

4.12 [**TestData-2**] TL-CBF spectrums: (a) original signal at 10 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (20.4, 3.5) and (ϕ_0, -2.5) [red cross] where $\phi_0 = -24$. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. 47

4.13 [**TestData-2**] TL-CBF spectrums: (a) original signal at 10 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (20.4, 3.5) and (ϕ_0, -2.5) [red cross] where $\phi_0 = 0$. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. 48

4.14 [**TestData-2**] TL-CBF spectrums: (a) original signal at 10 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (20.4, 3.5) and (ϕ_0, -2.5) [red cross] where $\phi_0 = 12$. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. 48

4.15 [**TestData-2**] TL-CBF spectrums: (a) original signal at 10 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (20.4, 3.5) and (ϕ_0, -2.5) [red cross] where $\phi_0 = 24$. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. 48

4.16 [**TestData-2**] TL-CBF spectrums: (a) original signal at 10 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (20.4, 3.5) and (ϕ_0, -2.5) [red cross] where $\phi_0 = 36$. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. 49

4.17 [**TestData-2**] TL-CBF spectrums: (a) original signal at 10 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (20.4, 3.5) and (ϕ_0, -2.5) [red cross] where $\phi_0 = 48$. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. 49

4.18 [**TestData-3**] Accuracy (a) and average RMSE over detected sources (b) of various algorithms as SNR is changed. Source trajectories are fixed to be $(-60.5, 2.5)$, $(-12.5, 3.5)$ and $(5.5, 1.2)$. 50

4.19 [**TestData-3**] Accuracy (a) and average RMSE over detected sources (b) of various algorithms as RMSE threshold varies (η). Source trajectories are fixed to be $(-60.5, 2.5)$, $(-12.5, 3.5)$ and $(5.5, 1.2)$. 50

4.20 Amplitude relative error of the three sources scenarios as SNR (a) and ϕ_0 (b) varies for the proposed method. 51

4.21 [**TestData-3**] TL-CBF spectrums: (a) original signal at 1 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are $(-60.5, 2.5)$, $(-12.5, 3.5)$ and $(5.5, 1.2)$ [red cross]. The estimated source trajectories are indicated by red circle in (a), (b) and (c), which are partially or fully removed in the subsequent spectra (b), (c) and (d) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. 52

4.22 [**TestData-3**] TL-CBF spectrums: (a) original signal at 5 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are $(-60.5, 2.5)$, $(-12.5, 3.5)$ and $(5.5, 1.2)$ [red cross]. The estimated source trajectories are indicated by red circle in (a), (b) and (c), which are partially or fully removed in the subsequent spectra (b), (c) and (d) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. 52

4.23 [**TestData-3**] TL-CBF spectrums: (a) original signal at 10 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are $(-60.5, 2.5)$, $(-12.5, 3.5)$ and $(5.5, 1.2)$ [red cross]. The estimated source trajectories are indicated by red circle in (a), (b) and (c), which are partially or fully removed in the subsequent spectra (b), (c) and (d) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. 53

4.24 [**TestData-3**] TL-CBF spectrums: (a) original signal at 15 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are $(-60.5, 2.5)$, $(-12.5, 3.5)$ and $(5.5, 1.2)$ [red cross]. The estimated source trajectories are indicated by red circle in (a), (b) and (c), which are partially or fully removed in the subsequent spectra (b), (c) and (d) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. 53

4.25 [**TestData-4**] Accuracy (a) and average RMSE over detected sources (b) of various algorithms vs ϕ_0. Two sources have fixed trajectories $(60.5, 2.5)$ and $(5.5, 1.2)$, while the third source trajectory is $(\phi_0, -3.8)$. The parameter ϕ_0 varies with a step size of 6, ranging from -60 to 54. 54

4.26 [**TestData-4**] Accuracy (a) and average RMSE over detected sources (b) of various algorithms as RMSE threshold varies (η). Two sources have fixed trajectories $(60.5, 2.5)$ and $(5.5, 1.2)$, while the third source trajectory is $(\phi_0, -3.8)$. The parameter ϕ_0 varies with a step size of 6, ranging from -60 to 54. 55

4.27 [**TestData-4**] TL-CBF spectrums: (a) original signal at 5 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are $(60.5, 2.5)$, $(5.5, 1.2)$ and (ϕ_0, -2.5) [red cross] where $\phi_0 = -12$. The estimated source trajectories are indicated by red circle in (a), (b) and (c), which are partially or fully removed in the subsequent spectra (b), (c) and (d) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. 56

4.28 [**TestData-4**] TL-CBF spectrums: (a) original signal at 5 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are $(60.5, 2.5)$, $(5.5, 1.2)$ and (ϕ_0, -2.5) [red cross] where $\phi_0 = 0$. The estimated source trajectories are indicated by red circle in (a), (b) and (c), which are partially or fully removed in the subsequent spectra (b), (c) and (d) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. 56

4.29 [**TestData-4**] TL-CBF spectrums: (a) original signal at 5 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are $(60.5, 2.5)$, $(5.5, 1.2)$ and (ϕ_0, -2.5) [red cross] where $\phi_0 = 48$. The estimated source trajectories are indicated by red circle in (a), (b) and (c), which are partially or fully removed in the subsequent spectra (b), (c) and (d) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. 57
4.30 [**TestData-4**] TL-CBF spectrums: (a) original signal at 5 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are $(60.5, 2.5)$, $(5.5, 1.2)$ and (ϕ_0, -2.5) [red cross] where $\phi_0 = 54$. The estimated source trajectories are indicated by red circle in (a), (b) and (c), which are partially or fully removed in the subsequent spectra (b), (c) and (d) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed. 57

List of Tables

Table Page

3.1 Concise overview of the neural network architecture, detailing operations, channels, kernel size, stride, padding and activation and parameters for each layer in the encoder and decoder. 14
3.2 Accuracy (in $\%$) and average processing time of various algorithms on different test datasets. 20
3.3 Average RMSE over detected sources of various algorithms on different test datasets. 20
3.4 Accuracy (in $\%$) of various algorithms on different test datasets as detected peaks (P) varies from K to $2K+1$, where K represents the number of sources present. 21
3.5 Average RMSE over detected sources of various algorithms on different test datasets as detected peaks (P) varies from K to $2K+1$, where K represents the number of sources present. 21

4.1 Accuracy (in $\%$) of various algorithms on different test datasets ($\eta = 2.4$). . . . 41
4.2 Average RMSE over detected sources of various algorithms on different test datasets. 42

Chapter 1

Introduction

Localization and tracking of objects have many applications including human-robot interaction, autonomous navigation, and smart homes, [1, 2, 3, 4, 5, 6, 7]. Recently, these techniques have also been applied in urban sound sensing [8], wildlife monitoring [9] and surveillance [10]. These applications showcase the versatility and importance of localization and tracking technologies in various domains. The techniques for localization and tracking vary based on the specific application and the sensors in use. For instance, acoustic localization methods utilize sound signals for object positioning, whereas vision-based localization rely on cameras or imaging sensors to identify and track visual features. In this work, our emphasis lies on sensor array data analysis, specifically for the task of direction of arrival (DOA) estimation. DOA estimation is a fundamental signal processing task that involves determining the direction from which a signal or wavefront arrives at a sensor array. This information is crucial for localizing and tracking sources, identifying the origin of signals, and extracting useful information from the received signals. It is widely used in various fields such as wireless communications, radar systems, sonar systems, and acoustic signal processing. By accurately estimating the DOA of signals, it becomes possible to localize and track sources, improve communication quality, enhance navigation systems, and facilitate target detection and identification.

1.1 Related works

DOA estimation techniques typically involve processing multichannel sensor data to estimate the angles or directions from which signals originate. To accurately estimate the DOA, a range of traditional techniques are utilized, including conventional beamforming [11], Multiple Signal Classification (MUSIC) [12], Estimation of Signal Parameters via Rotational Invariance Techniques (ESPRIT) [13], Root-MUsic SIgnal (Root-MUSIC) [14], and others [15]. Each technique has its own advantages and limitations, making them suitable for different applications and scenarios. MUSIC algorithm is powerful tool and it exploits the orthogonality

between the signal and noise subspaces to estimate the DOAs of the sources. It offers high resolution and accuracy, but it can be computationally intensive and require prior knowledge of the number of sources present. Another popular method is ESPRIT, which exploits the phase differences between pairs of sensors to estimate the DOAs of the sources. Unlike MUSIC, ESPRIT directly estimates the signal subspace using the data from the sensors without the need for eigen decomposition. This approach can offer computational advantages over MUSIC. Root-MUSIC is a variant of the MUSIC algorithm that operates by computing the eigen-decomposition of the auto-correlation matrix of the received signals. It exploits the roots of the polynomial to directly estimate the DOAs of the sources. Unlike MUSIC and ESPRIT, this method eliminates the requirement of grid, making it more efficient approach for DOA estimation. While these methods are effective, they all require a sufficient number of data snapshots to accurately estimate DOA, especially in scenarios with low signal-to-noise ratios (SNRs). Additionally, they typically assume that the DOA remains constant and that the number of sources is known beforehand.

Over the past few years, substantial progress has been achieved in DOA estimation through the adoption of deep learning techniques. These methods harness the power of neural networks to learn complex mappings from sensor array data to source directions, offering potential advantages in terms of flexibility and adaptability. A key advantage of employing data-driven learning methods is their capacity to refine DOA estimation accuracy with larger and more diverse training datasets. [16] proposed a learning-based approach utilizing Multilayer Perceptrons (MLP) for DOA estimation. Their approach formulates the DOA estimation as a classification problem with 360 classes, corresponding to angles ranging from 0° to 359°. They utilized the input features extracted from the Generalized Cross Correlation (GCC) [17] vectors. Following training, the classifier generates posterior probabilities for all 360 classes based on an input, and the class with the highest probability is identified as the estimated class. Moreover, [18, 19, 20] also utilized GCC as an input feature and exoerimented with different Convolutional Neural Network (CNN) architectures for DOA classification. In [21] and [22], the authors designed $M - 1$ layer CNNs with three fully connected dense layer to estimate DOA for each time frame containing one and two speakers, respectively, in reverberant environments. Here M represents the number of microphones in a microphone array. The authors didn't extract explicit features; instead, they utilized the multichannel short-time Fourier transform (STFT) phase spectrograms directly. In multi-speaker localization, as outlined in [22], the authors make use of the assumption that speakers are not active simultaneously across time-frequency (TF) bins. This assumption is commonly referred to as the W-disjoint orthogonality (WDO) principle [23]. As more innovative architectures have emerged, researchers have utilized these architectures to address challenges in DOA estimation [24]. For instance, in [25], the authors employed the U-Net architecture [26] to estimate one time-frequency (TF) mask for

each considered DOA, where each TF bin was associated with a particular DOA. This spectral mask was then utilized for source separation. Building upon this idea, [27] utilized a similar approach to dynamically locate multiple speakers. They utilized the WDO property, which states that each TF bin is dominated by at most a single speaker, allowing it to be associated with a single DOA. In [28], weakly-supervised methods based on manifold learning were utilized for sound source localization (SSL). The fundamental idea was that the high-dimensional multichannel observed data reside in a low-dimensional acoustic space, which is influenced by a few hidden variables. These studies utilized features derived from the relative transfer functions (RTFs) between microphone. Similarly, [29] used this principle through a Variational Autoencoder (VAE) [30]. They trained the VAE to generate the phase of RTFs between microphones, concurrently with a DOA classifier based on phase of RTF. [31] adapted a pre-trained neural network to unseen conditions in a unsupervised way. In [32], a 3D CNN was employed on steered response power with phase transform (SRP-PHAT) power maps to perform single source tracking. Inspired by computer vision techniques of image translation, [33] adopted a supervised image mapping approach. This method involves mapping a 2D input image with broad peaks to an output 2D image with sharp peaks. Subsequently, the source location is determined based on the output image. Recently, DL-based methods are also used for joint sound event localization and detection (SELD) task [34, 35].

1.2 Motivation

The primary motivation of this work is to address two key challenges of the existing DOA estimation algorithms. *First*, classical localization approaches, such as conventional beamforming [11], MUSIC [12], Root-MUSIC [14] and ESPRIT [13] rely on analytical properties of observed sensor array signals for DOA estimation. Though successful, these methods do not generalize well to non-ideal behaviors such as multi-path signal propagation and uncertain noise characteristics. *Second*, most methods use block-level processing assuming the DOA to be constant within a block (multiple measurements) followed by using tracking filters on these block estimates. While suitable for slow-moving targets, this assumption becomes problematic in scenarios involving fast motion.

Recognizing these challenges, there's a strong drive to explore new methods that can tackle these issues and improve the reliability and precision of DOA estimation. The *first* challenge is addressed by using a data-driven approach that harnesses the power of general function approximators, particularly neural networks, to learn and adapt to non-ideal behaviors. Deep learning-based methods have demonstrated promising results for source localization tasks in environments characterized by low SNR and reverberant [24, 36]. The *second* challenge is addressed by working in DOA-trajectory space instead of DOA space. In trajectory localization

(TL), source trajectories are estimated instead of point estimates. Recently, Bayesian method [37] and neural networks [38, 32] have been used for estimating trajectories. Earlier work [39] developed conventional beamforming (TL-CBF) and sparse Bayesian learning (TL-SBL) algorithms for trajectory localization (TL) when DOAs are linearly changing in a block. By leveraging these advancements and adopting a data-driven paradigm, this works aims to overcome the limitations of traditional DOA estimation methods and pave the way for more accurate and adaptable localization techniques capable of effectively addressing real-world challenges.

1.3 Contributions

The thesis explores data-driven methods to estimate source trajectories parameters from sensor array data, offering a novel method to capture non-ideal behaviors through general function approximators such as neural networks. To achieve this objective, the thesis examines two different methods: *grid-based* and *grid-free*.

1. In the grid-based approach, we build a fully convolutional neural network (FCNN) with encoder-decoder structure, inspired by U-Net [26]. The FCNN takes low resolution TL-CBF spectrum as input and outputs a refined spectrum allowing precise identification of source trajectory parameters leading to better track estimates. We demonstrates the generalization ability of FCNN across various scenarios and compare its accuracy with baseline methods TL-CBF and TL-SBL [39]. We also compare the average processing time by each algorithms.

2. A *drawback* of grid-based methods is that the parameters are estimated over a predefined grid, limiting their resolution. They assume that the DOA (or DOA parameters) strictly falls on the grid points, leading to potential inaccuracies when estimating DOA (or DOA parameters) for sources that do not align precisely with the grid. The grid-free methods proposed in [40] involve sequential identification and correction of source trajectory parameters. We develop data-driven method which is *grid-free* and *re-useable*. A deep complex [41] regression network is proposed to achieve grid-free DOA trajectory parameter estimation. The network outputs are complex signal amplitude (per snapshot) and trajectory parameters for a single source. A residual is obtained by removing contributions of this source from the input data. This residual is then fed into the same network to identify the next source. This can be repeated until some stopping criteria is met thus making the network re-useable and independent of the number of sources to be estimated. The re-useable network is motivated by the residual-based processing commonly employed in greedy methods such as matching pursuit [42, 43].

1.4 Organization of Thesis

1. In Chapter 2, we will explore the background of the standard signal model, specifically on the Multiple Measurement Vectors (MMV) signal model for DOA estimation. Additionally, we will also explore the MMV signal model specifically designed where the DOA varies linearly with snapshot, offering insights into dynamic situations. Furthermore, the chapter reviews previously proposed methods, including TL-CBF and TL-SBL, to provide a comprehensive understanding of existing approaches in the field.

2. In Chapter 3, we will discuss the proposed data-driven approach for joint localization and tracking task, employing a deep learning-based U-Net architecture. We will explore how this architecture can be utilized to estimate linear trajectory parameters effectively.

3. In Chapter 4, we will discuss the proposed data-driven method which is grid-free and reuseable for the joint localization and tracking task, employing deep complex architecture to estimate DOA trajectory parameters of one source at a time.

4. We present our concise summary and potential future research in Chapter 5.

Chapter 2

Background

The aim of DOA estimation is to determine the direction from which signals arrive at a sensor array, providing valuable spatial information about the sources emitting these signals. An essential part of DOA estimation is creating a precise signal model. This model helps us understand how signals travel from their source to the sensor array. In this chapter, we will present a comprehensive overview of the signal model employed in DOA estimation, with a specific emphasis on the far-field model and its mathematical formulation. Additionally, we will examine the multiple measurement vectors (MMV) model, which accommodates scenarios involving both static and dynamically changing DOA. Following this discussion, we will delve into the baseline methods utilized to achieve precise and reliable DOA estimation.

2.1 Signal Model

The measurement $\mathbf{y} \in \mathbb{C}^N$, captured by a uniform linear array (ULA) comprising N sensors, in the presence of K sources, is described as follows:

$$\mathbf{y} = \sum_{k=1}^{K} \mathbf{a}(\theta_k) s_k + \mathbf{w} = \mathbf{A}(\mathbf{\Theta})\mathbf{s} + \mathbf{w}. \tag{2.1}$$

Here, in the above equation, $\mathbf{a}(\theta_k) = \mathbf{a}_k = [1 \quad e^{j2\pi \frac{d}{\lambda}\sin(\theta_k)} \quad \cdots \quad e^{j2\pi(N-1)\frac{d}{\lambda}\sin(\theta_k)}]^T$ is the steering vector corresponding to k^{th} source with DOA θ_k. The term s_k denotes the k^{th} source amplitude, while $\mathbf{A}(\mathbf{\Theta}) = [\mathbf{a}_1(\theta_1) \cdots \mathbf{a}_K(\theta_K)] \in \mathbb{C}^{N \times K}$ is the sensing matrix with $\mathbf{\Theta} = [\theta_1, \cdots, \theta_K]^T$. Furthermore, $\mathbf{s} = [s_1, \cdots, s_K]^T$ represents the source amplitude vector, and $\mathbf{w} \in \mathbb{C}^N$ accounts for the additive noise. Narrowband wavelength is λ and d is the sensor separation in ULA.

In scenarios where multiple observations are available and under the assumption of static DOA, the multiple measurement vector (MMV) signal model is given as:

$$\mathbf{Y} = \mathbf{A}\mathbf{X} + \mathbf{W} = [\mathbf{A}\mathbf{x}_1 \cdots \mathbf{A}\mathbf{x}_L] + \mathbf{W}, \tag{2.2}$$

where $\mathbf{Y} = [\mathbf{y}_1 \cdots \mathbf{y}_L] \in \mathbf{C}^{N\times L}$ is the L-snapshot measurement matrix, $\mathbf{A} = [\mathbf{a}_1 \cdots \mathbf{a}_K]$ is the sensing matrix, $\mathbf{X} = [\mathbf{x}_1 \cdots \mathbf{x}_L] \in \mathbf{C}^{K\times L}$ represents the source amplitudes of K sources at L snapshots, and $\mathbf{W} = [\mathbf{w}_1 \cdots \mathbf{w}_L] \in \mathbf{C}^{N\times L}$ is the additive noise. In this MMV model, each L-snapshot is utilized to estimate the DOA parameters. The utilization of multiple snapshots enables the exploitation of temporal information, contributing to more reliable estimates of DOA parameters.

The above MMV model assumes static DOA and therefore is not applicable for modelling moving sources. However, in practical scenarios, sources often exhibit dynamic behavior, leading to changes in their DOA. The difference between the assumption of unchanging DOA and the dynamic behavior observed in real-world highlights the need for alternative models capable of accommodating the dynamic nature of sources. To address this limitation, [39] introduces a first order correction by incorporating linear motion of DOA across snapshots within a block. This adjustment allows the model to better capture the movement of sources over time, enhancing its applicability to scenarios involving dynamic sources. Thus, under the assumption of linearly changing DOA, the DOA θ_k^l for k^{th} source at l^{th} snapshot is given as,

$$\theta_k^l = \phi_k + (\frac{l-1}{L-1})\alpha_k, \quad l = 1, 2, \cdots, L. \tag{2.3}$$

The parameter pair (ϕ_k, α_k) models DOA trajectory of the k^{th} source. Here, ϕ_k denotes the initial DOA observed in the first snapshot, while α_k quantifies the cumulative change in DOA from the initial observation. The MMV model for linear DOA motion [39] can be expressed as

$$\mathbf{Y} = \tilde{\mathbf{A}}\tilde{\mathbf{X}} + \mathbf{W} \tag{2.4}$$

where $\tilde{\mathbf{A}} = [\tilde{\mathbf{A}}_1(\phi_1, \alpha_1) \cdots \tilde{\mathbf{A}}_K(\phi_K, \alpha_K)] \in \mathbf{C}^{N\times KL}$ is a matrix that contains steering matrix for changing DOA across L snapshots for each of the K sources. $\tilde{\mathbf{A}}_k(\phi_k, \alpha_k) = [\mathbf{a}(\theta_k^1) \cdots \mathbf{a}(\theta_k^L)]$, $\tilde{\mathbf{X}} = [\tilde{\mathbf{X}}_1 \cdots \tilde{\mathbf{X}}_K]^T \in \mathbf{C}^{KL\times L}$ with $\tilde{\mathbf{X}}_k = \text{diag}(\mathbf{x}_k) \in \mathbf{C}^{L\times L}$, $\mathbf{x}_k = [s_k^1 \cdots s_k^L]$ is a vector of L amplitudes of the k^{th} source.

Extending the linear motion model to higher-order polynomials or other parametric trajectories is feasible, although it comes with increased computational demands. While higher-order polynomial trajectories or complex parametric models offer enhanced flexibility in capturing dynamic source behavior, they also escalate computational complexity.

2.2 Baseline Methods

2.2.1 TL-CBF

In [39], TL-CBF was introduced as a modification of conventional beamforming [11] for the linearly changing DOA signal model. This methodology aims to capture the linear variation in DOA observed within a data block, thereby facilitating enhanced localization of moving sources in contrast to conventional method. The power spectrum for each (ϕ, α) pair is computed by averaging the squared inner products between the received signals and their corresponding steering vectors across L snapshots. This process yields a 2D representation of the power distribution across different DOA trajectory parameters. Mathematically, it is expressed as:

$$P_{\text{TL-CBF}}(\phi, \alpha) = \frac{1}{L} \sum_{l=1}^{L} |\mathbf{y}_l^H \mathbf{a}(\theta^l)|^2. \tag{2.5}$$

The power spectrum is computed over a pre-defined grid. The location of the peaks in the spectrum provides estimates of the DOA trajectories within the L-snapshot block. An example TL-CBF spectrum is shown in Figure 2.1. Since the computation of the power spectrum involves correlating the received signals with steering vectors. So, even small deviations in (ϕ, α) from the true DOA parameters (ϕ^*, α^*) can result in very high correlations, leading to broad peaks in the spectrum.

Pros and Cons: This method is computationally efficient. However, it suffers from broad peak issues, which restrict its ability to accurately locate nearby source trajectories. An example of scenarios with nearby source trajectories scenarios is shown in Figure 2.2, where detected peaks are notably distant from the true DOA source trajectories.

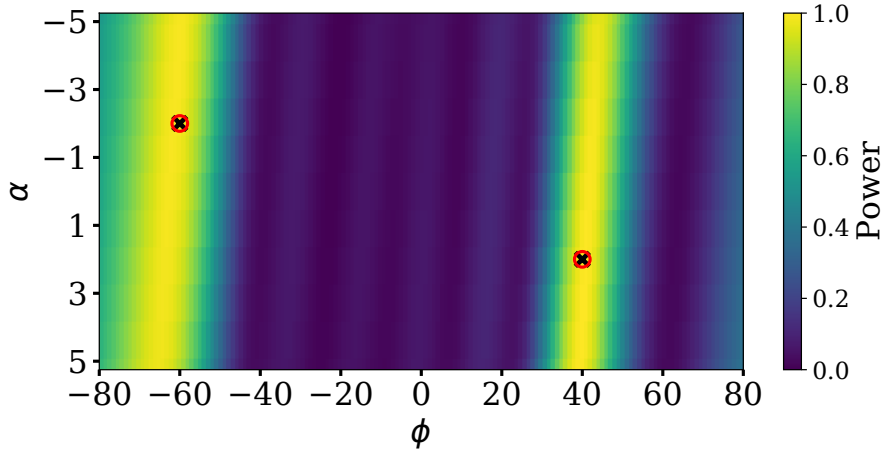

Figure 2.1: TL-CBF Spectrum. Two moving sources of equal amplitudes with source trajectories $(-60, -2)$ and $(40, 2)$ [black cross] at 10 dB SNR. Detected peaks are shown by red circle.

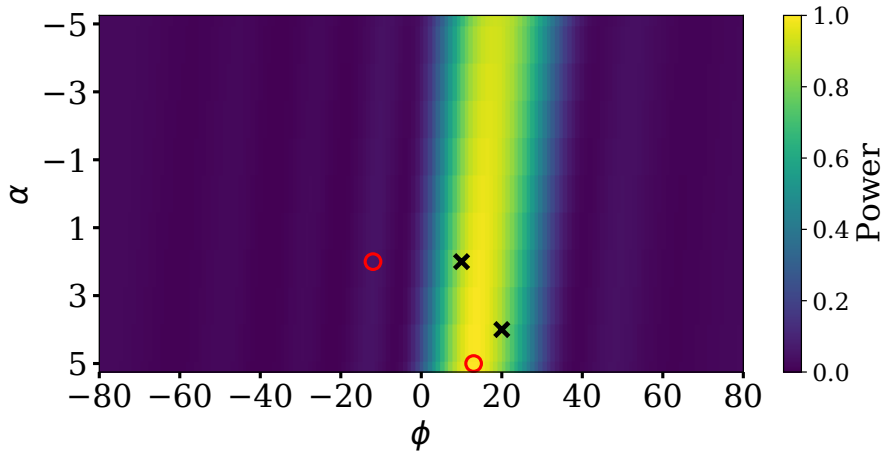

Figure 2.2: TL-CBF Spectrum. Two moving sources of equal amplitudes with source trajectories $(10, 2)$ and $(20, 4)$ [black cross]. Detected peaks are shown by red circle.

2.2.2 TL-SBL

Sparse Bayesian learning (SBL) is a compressive sensing algorithm [44, 45], which has been instrumental in various signal processing applications. Its derivative, TL-SBL was developed [39] to estimate DOA trajectory parameters. By formulating a sparse model, the update rule for TL-SBL is derived as

$$\hat{\gamma}_m^{\text{new}} = \hat{\gamma}_m^{\text{old}} \frac{\mathbf{y}_v^H \boldsymbol{\Sigma}_{\mathbf{y}_v} \hat{\mathbf{A}}_m \hat{\mathbf{A}}_m^H \boldsymbol{\Sigma}_{\mathbf{y}_v}^{-1} \mathbf{y}_v}{\text{Tr}[\boldsymbol{\Sigma}_{\mathbf{y}_v}^{-1} \hat{\mathbf{A}}_m \hat{\mathbf{A}}_m^H]}, \tag{2.6}$$

where $\hat{\mathbf{A}}_m = \mathbf{I}_L \otimes \tilde{\mathbf{A}}_m$, and $\text{Tr}[\cdot]$ denotes trace of a matrix. The mth grid point represents a potential (ϕ_m, α_m) pair with corresponding steering matrix $\tilde{\mathbf{A}}_m$. The parameter vector $\boldsymbol{\gamma}$ is sparse and the locations of non-zero entries of $\boldsymbol{\gamma}$ signify the source DOA trajectory estimates. An example TL-SBL spectrum is shown in Figure 2.3 and has sharp peaks.

Pros and Cons: This method provides precise peak resolution, which is advantageous for accurately estimating source trajectories even for nearby source. However, its high computational complexity presents challenges, particularly in real-time processing scenarios where computational resources are limited.

2.3 Summary

In this chapter, we discussed about the signal model and baseline methods (TL-CBF and TL-SBL) for the joint localization and tracking task. We also discuss about their advantages and disadvantages in terms of computational speed and precision. Achieving an optimal trade-

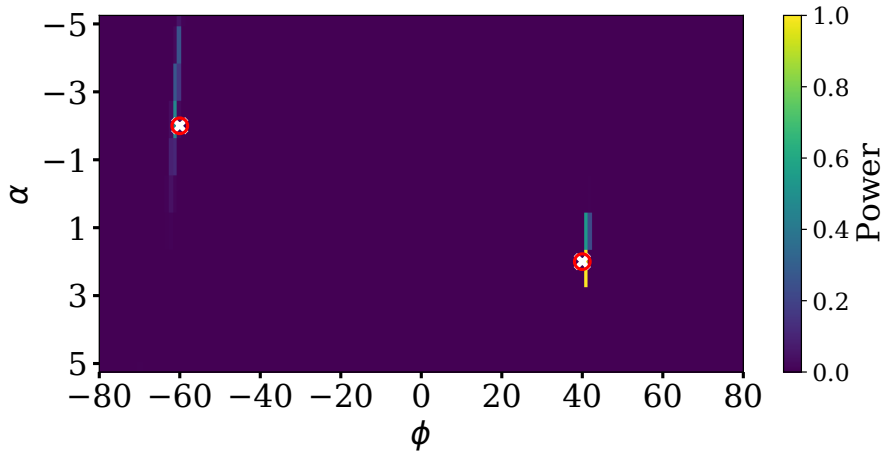

Figure 2.3: TL-SBL Spectrum. Two moving sources of equal amplitudes with source trajectories $(-60, -2)$ and $(40, 2)$ [white cross]. Detected peaks are shown by red circle.

off between these performance indicators is essential for enhancing the performance of DOA estimation methods in real-world scenarios.

Chapter 3

Deep architecture for DOA trajectory localization

Trajectory localization poses inherent challenges in diverse scenarios. In the previous chapter, we discussed about the signal model for lineraly changing DOA and previously proposed algorithms, namely TL-CBF and TL-SBL [39], to address this task. TL-CBF, while computationally efficient, grapples with broad peak issues, limiting its effectiveness in locating nearby trajectories. In contrast, TL-SBL offers precise peak resolution but at the cost of increased computational burden, posing challenges in scenarios where real-time processing is crucial. This trade-off between computational speed and precision is crucial in trajectory localization. Achieving high accuracy in estimating source trajectories often requires complex algorithms that can handle various scenarios effectively. However, these sophisticated methods may come at the cost of increased computational demands, which can be impractical in real-world applications where real-time processing is essential. Recognizing the trade-offs inherent in these algorithms, a different approach is proposed to bridge the gap between computational efficiency and precision in trajectory localization. This involves the development of a neural network explicitly tailored to transform low-resolution TL-CBF spectra into high-resolution counterparts. The motivation behind this lies from the belief that valuable information is contained within the peak structure of TL-CBF spectra, particularly regarding merged peaks. The neural network, by harnessing this latent information, aims to enhance peak resolution and, consequently, improve the localization accuracy. This approach holds particular promise in domains such as robotics and autonomous systems, where the ability to accurately and swiftly localize trajectories is paramount for effective decision-making and operation. This chapter discusses the proposed deep architecture and its impact on locating source trajectory parameters.

3.1 Network Architecture

Inspired by the U-Net [26] architecture, we use a deep fully convolutional neural network (FCNN) model with an encoder-decoder structure and skip connections to perform refinement

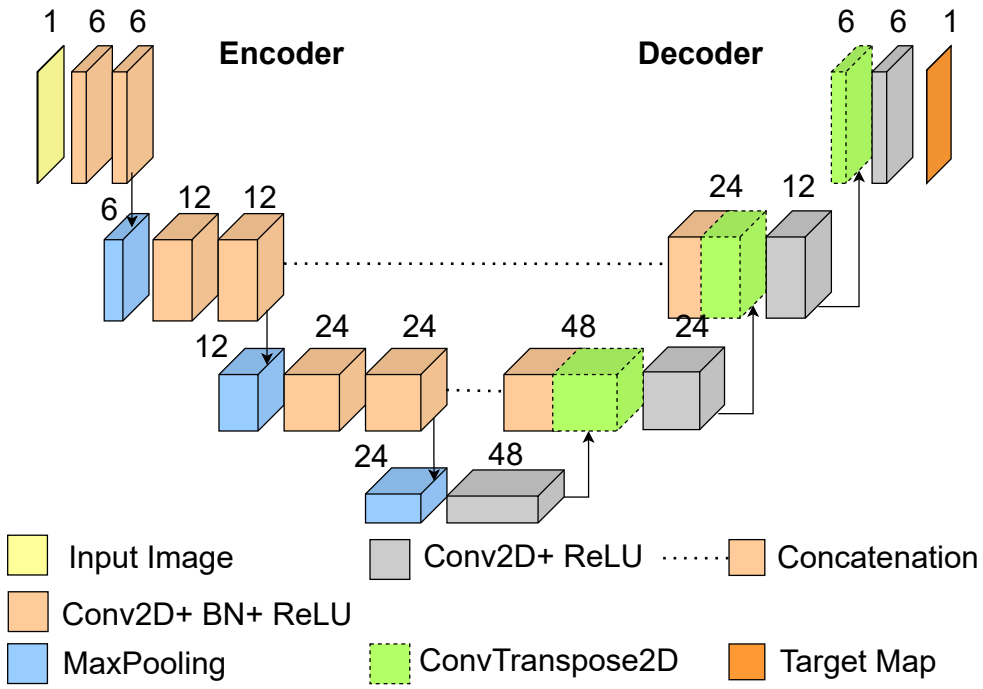

Figure 3.1: U-Net architecture with 2 skip connections, annotated with the number of channels above each block.

of TL-CBF spectrum. The architecture features a distinctive U-shape comprising two primary paths: a contracting path (encoder), and an expansive path (decoder). In general, the contracting path utilized the convolutional and max pooling layers to decrease the spatial resolution of the input image while simultaneously increasing the number of feature channels. This process allows the network to capture the high-level features of the image. On the other hand, the expansive path employs transposed convolutional layers to enhance the spatial resolution of the feature maps while reducing the number of feature channels. A notable strength of the U-Net architecture lies in its ability to seamlessly combine contextual information and intricate details, making it particularly well-suited for tasks requiring precise localization. This is achieved through skip connections between the contracting and expansive paths, which allow the network to combine low-level and high-level features during the decoding process, resulting in more comprehensive feature representations.

3.1.1 Encoder

In the architectural design, the encoder network plays a pivotal role in extracting spatial features from two-dimensional input images. This process is visualized in Figure 3.1. This initial stage of feature extraction sets the foundation for subsequent layers in the network, con-

tributing to the model's overall capacity to understand and interpret spatial information. As the encoded features progress through the network, they form a compact and abstract representation that serves as the basis for subsequent decoding stages in the architectural framework. The encoder employs a series of multiple convolutional blocks and max pooling layers to systematically capture and distill hierarchical features from the input data. For example, in image classification, lower-level features could be edges, corners, or textures, while higher-level features represent objects like cars, animals, or buildings. This features allows the network to focus on relevant patterns. Within each convolutional block, different filters are employed to convolve the input, extracting relevant features through non-linear transformations. Subsequently, the max pooling layers perform downsampling, reducing the spatial dimensions of the feature maps while retaining essential information. This hierarchical feature extraction process enables the network to focus on most relevant patterns and spatial relationships within the input images.

The basic convolutional blocks within the architecture are comprised of a dilated convolutional layer (Conv2D*), a batch normalization (BNorm) layer, and Rectified Linear Unit (ReLU) activation as the non-linear function. Specifically, the parameters for the convolutional operation include a kernel size of 3, a dilation rate of 2, and a stride of 1 which defines the step size at which the filter is applied to the input feature. Additionally, padding of 2 ensures that the spatial dimensions of the feature maps remain consistent throughout the convolutional process, preserving information at the borders of the input data. The dilated convolutional layer, with its dilation rate, enables the network to capture wider context and long-range dependencies in the input data. Batch normalization contributes to stable training dynamics by normalizing intermediate feature maps, and the ReLU activation introduces non-linearity, enabling the network to learn complex patterns. The max pooling layers are configured with a pool size of 2 and a stride of 2. These settings determine the dimensions and overlap of the pooling regions, influencing the downsampling process. Additionally, the final convolutional block in the encoder is composed of a non-dilated convolutional (Conv2D) layer succeeded by a ReLU activation function. The specific parameters for this operation are a kernel size of 3, a stride of 1, and padding of 1. These hyperparameters and introduction of non-linearity, shape the behavior of the convolutional operation, influencing factors such as receptive field size, feature extraction capability, and spatial resolution. Careful parameter selection enhances the encoder's effectiveness in capturing essential spatial features.

3.1.2 Decoder

The decoder network is responsible for reconstructing the spatial features extracted by the encoder into the target map. This process involves transposed convolutional blocks, whose output is concatenated with high-level feature maps for each channel, as illustrated by the skip

Layer	Operation	Channels	Kernel size	Stride	Padding	Activation	Parameters
			Encoder				
1	Conv2D* + BNorm	6	(3, 3)	1	2	ReLU	60 + 12
2	Conv2D* + BNorm	6	(3, 3)	1	2	ReLU	330 + 12
3	Max Pooling	-	(2, 2)	2	-	-	-
4	Conv2D* + BNorm	12	(3, 3)	1	2	ReLU	660 + 24
5	Conv2D* + BNorm	12	(3, 3)	1	2	ReLU	1308 + 24
6	Max Pooling	-	(2, 2)	2	-	-	-
7	Conv2D* + BNorm	24	(3, 3)	1	2	ReLU	2616 + 48
8	Conv2D* + BNorm	24	(3, 3)	1	2	ReLU	5208 + 48
9	Max Pooling	-	(2, 2)	2	-	-	-
10	Conv2D	48	(3, 3)	1	2	ReLU	10416
			Decoder				
11	ConvTranspose2D	24	(4, 4)	2	1	-	18456
12	Concatenation	24+24	-	-	-	-	-
13	Conv2D	24	(3, 3)	1	1	ReLU	10392
14	ConvTranspose2D	12	(5, 5)	2	1	-	7212
15	Concatenation	12+12	-	-	-	-	-
16	Conv2D	12	(3, 3)	1	1	ReLU	2064
17	ConvTranspose2D	6	(5, 5)	2	1	-	1806
18	Conv2D	6	(3, 3)	1	1	ReLU	330
19	Conv2D	1	(1, 1)	1	0	ReLU	7
			Total Parameters				61573

Table 3.1: Concise overview of the neural network architecture, detailing operations, channels, kernel size, stride, padding and activation and parameters for each layer in the encoder and decoder.

connections in Figure 3.1. The use of skip connections is a distinctive feature of the architecture, linking corresponding layers between the encoder and decoder. This connectivity aids in the integration of high-resolution details from the encoder, contributing to the precise localization and reconstruction of the target map. The transposed convolutional (ConvTranspose2D) layer play a pivotal role in upsampling the spatial features to match the original resolution, allowing for a detailed and accurate representation of the target map. The concatenation of these features through skip connections enhances the network's ability to capture both fine-grained and contextual information, resulting in a more robust and effective decoding process. The absence of a skip connection in the last transposed convolutional block is intentional to to re-

duce any potential noise or distortion that may arise from low-resolution TL-CBF input, with a focus on maintaining high-resolution details essential for accurate target map generation. The parameters for the transposed convolution operations are specified as a kernel size of $[4, 5, 5]$, a stride of 2, and padding of 1. The convolutional blocks utilized within the decoder consist of a convolutional layer followed by Rectified Linear Unit (ReLU) activation. For these blocks, the kernel size, stride, and padding are set at 3, 1, and 1 respectively. Finally, to attain a prediction map of the same size as the input dimension, a convolutional block with a kernel size of 1 is employed. This specific operation ensures that the output of the network aligns with the original spatial dimensions, providing a prediction map consistent with the input data. Table 3.1 offers a detailed overview of the neural network architecture, including key parameters and specifications.

3.2 Training Objectives: Target Map Generation and Loss Functions

Since our approach involves supervised learning, obtaining ground truth is essential. Specifically, for the task of refining the low-resolution TL-CBF spectrum into high-resolution counterparts, we need a target map. This target map serves as a representation of the true trajectory parameters, offering a reference for the model to learn and refine its predictions.

3.2.1 Target Map

The U-Net architecture serves to enhance input images to achieve the desired target output. In the context of trajectory localization, a potential target image is conceived as a binary representation, with ones precisely positioned at the coordinates of true source trajectory parameters. However, utilizing this binary target map can sometimes lead to an exceedingly small loss during training, which in turn can cause gradient underflows [46]. Gradient underflows occur when the gradient of the loss function becomes too small to effectively update the network's weight during the training. This can result in slow convergence or even halt the training process, emphasizing the importance of carefully designing target maps to ensure stable and meaningful training outcomes.

To overcome this challenge, exploring alternative target map representations or adjusting the loss function is required to ensure accurate refinement of the low-resolution TL-CBF spectrum into high-resolution counterparts. Thus, to achieve our objective of predicting DOA trajectories as close as possible to the true source trajectories, we utilize an RMSE-based target map, as depicted in Figure 3.2. Let θ_k^{*l} be the DOA trajectory corresponding to the k^{th} source (ϕ_k^*, α_k^*). For any point (ϕ, α) in the target map with DOA trajectory θ_k^l for the k^{th} source, we have

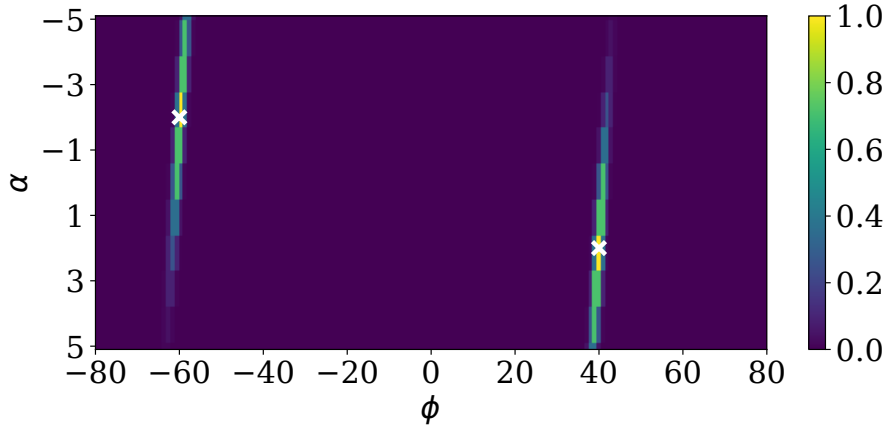

Figure 3.2: RMSE-based target map: there are two moving sources of equal amplitudes with DOA trajectories parameterized by $(-60, -2)$ and $(40, 2)$ [white cross].

$$E_k = \sqrt{\frac{\sum_{l=0}^{L-1}(\theta_k^{*l} - \theta_k^l)^2}{L}}, \quad k = 1, 2, \ldots, K, \tag{3.1}$$

$$I_k(\phi, \alpha) = e^{-E_k^2/\sigma^2}, \quad I(\phi, \alpha) = \sum_{k=1}^{K} I_k(\phi, \alpha), \tag{3.2}$$

where σ is hyperparameter controlling the rate of decay. Equation 3.1, denotes the RMSE between the ground truth trajectory and the trajectory generated from different grid pairs (ϕ, α) for the k^{th} source. Exponential decay is used to assign higher intensity values to the grid points in the target map where the RMSE is small. The target map is obtained by normalizing $I(\phi, \alpha)$ through division by its maximum value, resulting in a range from 0 to 1.

In Figure 3.2, two moving sources are denoted by cross markers. The target map exhibits non-zero values in the proximity of these markers, signifying that the associated trajectories fall within an acceptable error range. The extent of this acceptable error can be fine-tuned by adjusting the hyperparameter σ in the target map calculation. It do not use any source amplitude information corresponding to ground truth DOA trajectory. This approach allows for a more robust and meaningful representation of the relationship between predicted and true DOA trajectories during the training process.

3.2.2 Loss functions

To compare the output of the proposed architecture against the target map, we employ two distinct loss functions: l_2-norm and l_2-norm with l_1 regularization. These loss functions are

defined as follows:

$$l_2 = ||I_{\text{output}} - I_{\text{target}}||_2, \tag{3.3}$$

$$l_{21} = ||I_{\text{output}} - I_{\text{target}}||_2 + \lambda||I_{\text{output}}||_1 \tag{3.4}$$

Here, I_{output} represents the output of the proposed architecture, and I_{target} is the corresponding target map. The l_2-norm quantifies the Euclidean distance between the output and target maps, while the l_2-norm with l_1 regularization introduces a regularization term $\lambda||I_{\text{output}}||_1$ to promote sparsity in the output map, where λ is a hyperparameter controlling the strength of this regularization. This regularization also known as Lasso regularization [47] encourages concise representations, focusing on essential features and potentially improving generalization and model performance by reducing overfitting. These loss functions provide a means to measure the dissimilarity between the predicted and true representations. By minimizing these loss functions, the model aims to generate output maps that closely align with the ground truth, facilitating accurate refinement of the low-resolution TL-CBF spectrum.

3.3 Experiments and Results

In this section, we cover data generation methodologies, implementation specifics, and performance metrics, followed by a comprehensive discussion of results. Together, these aspects give us a better understanding of how the experiments were set up, how the architecture was put into practice, and a detailed assessment of how well the proposed model improves the low-resolution TL-CBF spectrum.

3.3.1 Data generation

To evaluate how well our algorithm performs, we conducted validation tests using data we created synthetically. An 8-element Uniform Linear Array (ULA) with $\frac{\lambda}{2}$ spacing was employed, generating $L = 100$ snapshot data. A grid spanning $\phi \in [-80, 80]$ and $\alpha \in [-5, 5]$ with a unit separation was constructed. The data generation process involved simulating various multi-source DOA trajectories will be discussed in detail in the following sections. These trajectories were generated within the defined grid, giving us a varied and representative dataset to work with.

Training: We simulate numerous two ($K = 2$) source combinations with a minimum separation of 5° in ϕ. This allows the network to learn from diverse multi-source input patterns with varying degrees of source trajectory separation and amplitude differences. One of the sources has its amplitude sampled per snapshot independently and identically distributed (IID) from standard normal distribution $\mathcal{N}(0, 1)$. The other source also has its amplitude sampled

per snapshot in an IID manner from $\mathcal{N}(0, u)$ where u follows uniform distribution in the range $[0.5, 1]$. For each combination, we considered 10 uniformly random SNR values in the range of $[0, 20]$ dB. The output of TL-CBF applied to the sensor data $\mathbf{Y}$ yields an input image to the network with dimensions of 11×161. In our implementation, we partitioned the training data into three sets: train (*TrainData*), validation (*ValData*), and test (*TestData*). These sets contained 240,000, 2,500, and 610 examples, respectively. The target map, crucial for training, was generated using the equation (3.2). This simulation setup allowed us to create a diverse dataset, essential for training and evaluating the proposed deep-learning model in handling multiple source scenarios with varying conditions.

Testing: While the training phase utilized data featuring only 2 sources ($K = 2$), our objective extends to demonstrating the network's capacity to generalize. To achieve this, we assess its performance on data involving 3 sources ($K = 3$). This approach allows us to evaluate the model's adaptability and efficacy in scenarios beyond those encountered during training. By examining its performance under varied source configurations, we gain valuable information about the model's overall robustness and effectiveness. We generate two such challenging datasets:

1. In the dataset labeled **TestData-1**, we generated scenarios featuring 3 sources, varying the SNR values from -12 dB to 15 dB with step size of 3 dB. For each SNR value, we simulated 200 examples, resulting in a total of 2000 data points. The trajectories of two sources were set at $(60, 2)$ and $(5, 1)$, each with amplitude sampled from $\mathcal{N}(0, 1)$. The third source had a trajectory of $(-4, -1)$ with amplitude sampled from $\mathcal{N}(0, 0.5)$. This enables the evaluation of the model's performance across a range of noise levels, despite being trained primarily with positive SNR values.

2. In the dataset labeled **TestData-2**, observations were simulated at a fixed SNR value of 5 dB with 3 sources present. The trajectories of two sources were kept constant at $(60, 2)$ and $(5, 1)$, while the trajectory of the third source was set to $(\phi_0, -1)$. The value of ϕ_0 was varied in the range of $[-40, 42]$. Each case resulted in 200 data points, yielding a total of 5600 data points. Sources with fixed trajectory have amplitudes sampled from $\mathcal{N}(0, 1)$ while the third source has amplitude sampled from $\mathcal{N}(0, 0.5)$. This dataset was designed to evaluate the model's performance in scenarios with 3 sources under challenging conditions of low SNR and varying trajectories.

In each scenario, source amplitudes are sampled per snapshot independently and identically distributed (IID) from their respective distributions.

3.3.2 Implementation details

For generating target map, we use $\sigma = 1$. The U-Net model, proposed for this task, was implemented using the PyTorch library [48], with Kaiming initialization [49] applied to initialize the network's parameters. The model was trained independently using the two loss functions defined in equations (3.3) and (3.4) for a total of 50 epochs. During training, a batch size of 256 examples was utilized, with a learning rate of 10^{-3}, and the Adam optimizer [50] was employed. The learning rate was reduced by a factor of 0.1 after 40 epochs to aid convergence. Additionally, for comparison, we implemented trajectory localization algorithms, specifically TL-CBF and TL-SBL [39]. These algorithms serve as benchmarks to assess the performance of the proposed U-Net model in refining the low-resolution TL-CBF spectrum.

3.3.3 Performance metrics

The output of the proposed U-Net architecture is the high-resolution 2D power spectrum obtained by refining TL-CBF feature. In the 2D spectrum, the location of peaks gives the trajectory parameters (ϕ, α) corresponding to each source. When K sources are present, we identify atmost $P = 2K + 1$ peaks in the output spectrum. To quantitatively evaluate the model's performance, we compute the RMSE between each ground truth source trajectory and the estimated source trajectory derived from the identified peaks, as given in Equation 3.5. The Hungarian algorithm [51] is used to solve the assignment problem between the P peaks and K sources. It is a combinatorial optimization technique used to find the optimal assignment of agents to tasks, minimizing the total cost or maximizing the total profit, all within polynomial time complexity. This algorithm efficiently associates each peak with the most likely source trajectory, providing a basis for subsequent accuracy assessment. Once assigned, a source is said to be detected if the RMSE between the true θ_k^{*l} and the assigned track $\hat{\theta}_k^l$ is less than the threshold η. In this evaluation, we set $\eta = 1.2$ as a reasonable criterion for successful source detection. Accuracy is calculated as the ratio of total number of detected sources to the total number of sources present, over the complete dataset. This metric provides a comprehensive measure of the model's ability to precisely identify and associate sources within the refined high-resolution spectrum.

$$RMSE_k = \sqrt{\frac{\sum_{l=0}^{L-1}(\theta_k^{*l} - \hat{\theta}_k^l)^2}{L}}, \quad k = 1, 2, \ldots, K \tag{3.5}$$

3.3.4 Results and discussions

The comparative performance of TL-CBF, TL-SBL, U-Net with l_2 loss, and U-Net with l_{21} loss across various datasets is presented in Table 3.2 and 3.3. In terms of accuracy, the U-Net

		Algorithms			
S.No.	DatasetName / Complexity	TL-CBF	TL-SBL	U-Net-l_2	U-Net-l_{21}
1	TestData	73.68	93.27	94.67	**95.24**
2	TestData-1	50.31	61.31	64.85	**74.61**
3	TestData-2	73.37	**90.60**	88.98	90.04
4	Time (secs)	**0.25**	206.7	0.254	0.254

Table 3.2: Accuracy (in $\%$) and average processing time of various algorithms on different test datasets.

with l_{21} loss outperforms the U-Net with l_2 loss across different test datasets. This highlights the effectiveness of incorporating l_1 regularization in enhancing the model's generalization capabilities. TL-CBF consistently exhibits the lowest accuracy across different datasets compared to other methods. We also report the average time taken by each algorithm to generate the output spectrum image when implemented on the same machine. The proposed U-Net variants have very low time-complexity and additionally requires little time to process the TL-CBF input image. TL-SBL accuracy is relatively similar to the U-Net variants and consistently greater than TL-CBF. However, it has the highest average processing time among all methods. In terms of average RMSE over detected sources, the proposed U-Net variants exhibits the lower values in comparison to the U-Net's input TL-CBF. The average RMSE values quantify the accuracy of trajectory parameter estimation, with lower values indicating more precise results. Notably, for *TestData* and *TestData-1*, TL-SBL demonstrates the lowest average RMSE over detected sources, while for *TestData-2*, U-Net with l_{21} loss exhibits the lowest value. In contrast, TL-CBF shows higher average RMSE values compared to the other methods, indicating lower precision in trajectory parameter estimation.

		Algorithms			
S.No.	DatasetName	TL-CBF	TL-SBL	U-Net-l_2	U-Net-l_{21}
1	TestData	0.286	**0.199**	0.239	0.264
2	TestData-1	0.632	**0.420**	0.627	0.574
3	TestData-2	0.415	0.395	0.389	**0.384**

Table 3.3: Average RMSE over detected sources of various algorithms on different test datasets.

Accuracy and average RMSE over detected sources of various algorithms are reported in Table 3.4 and 3.5, respectively, for both *TestData-1* and *TestData-2* as the number of detected peaks P varies. We vary the number of detected peaks (P) from K to $2K + 1$, where K denotes the number of sources present. As number of detected peaks increases, accuracy of every algorithms increases for both test dataset. For *TestData-1*, U-Net with l_{21} loss has the

highest accuracy for all values of P, while TL-CBF consistently has lowest accuracy. For *TestData-2*, TL-SBl has the highest accuracy when $P = [3, 4, 6]$, while U-Net with l_2 loss has the highest accuracy when $P = [5, 7]$. For *TestData-1*, TL-SBL has the lowest average RMSE over detected sources for different values of P as compared to other methods. In *TestData-2*, TL-SBL has the lowest average RMSE when $P = 3$, but when P is greater than 3, U-Net with l_{21} has the lowest RMSE.

		TestData-1				TestData-2			
S.No.	Peaks (P)	TL-CBF	TL-SBL	U-Net-l_2	U-Net-l_{21}	TL-CBF	TL-SBL	U-Net-l_2	U-Net-l_{21}
1	3	37.96	48.76	48.43	**53.25**	61.35	**76.05**	74.92	74.77
2	4	39.33	55.90	61.10	**70.41**	67.57	**85.84**	85.78	86.57
3	5	40.05	59.10	64.28	**73.70**	71.42	89.0	88.64	**89.97**
4	6	44.40	60.61	64.66	**74.40**	72.79	**90.17**	88.94	89.97
5	7	50.31	61.31	64.85	**74.61**	73.68	93.28	94.67	**95.24**

Table 3.4: Accuracy (in $\%$) of various algorithms on different test datasets as detected peaks (P) varies from K to $2K + 1$, where K represents the number of sources present.

		TestData-1				TestData-2			
S.No.	Peaks (P)	TL-CBF	TL-SBL	U-Net-l_2	U-Net-l_{21}	TL-CBF	TL-SBL	U-Net-l_2	U-Net-l_{21}
1	3	0.629	**0.341**	0.568	0.553	0.361	**0.354**	0.369	0.360
2	4	0.577	**0.382**	0.616	0.586	0.386	0.384	0.389	**0.383**
3	5	0.582	**0.405**	0.627	0.579	0.405	0.392	0.391	**0.384**
4	6	0.605	**0.414**	0.627	0.574	0.412	0.395	0.389	**0.384**
5	7	0.632	**0.420**	0.628	0.573	0.415	0.395	0.389	**0.385**

Table 3.5: Average RMSE over detected sources of various algorithms on different test datasets as detected peaks (P) varies from K to $2K + 1$, where K represents the number of sources present.

TestData-1: In this experiment, the performance of different algorithms is analyzed as the SNR varies from -12 dB to 15 dB for $K = 3$ sources. U-Net with l_{21} loss has the highest average accuracy compared to other algorithms, while TL-CBF has the lowest as reported in Table 3.2. TL-SBL and U-Net with l_2 loss have approximately the same average accuracy. The average RMSE of TL-SBL is lowest compared to other methods for detected sources is reported in Table 3.3. The accuracy and average RMSE over detected sources of various algorithms are displayed in Figure 3.3. As the SNR increases, the accuracy of each algorithms increases but after further increment in SNR, the accuracy saturates. In parallel, the average RMSE over detected sources for all methods exhibits a decreasing trend as SNR increases.

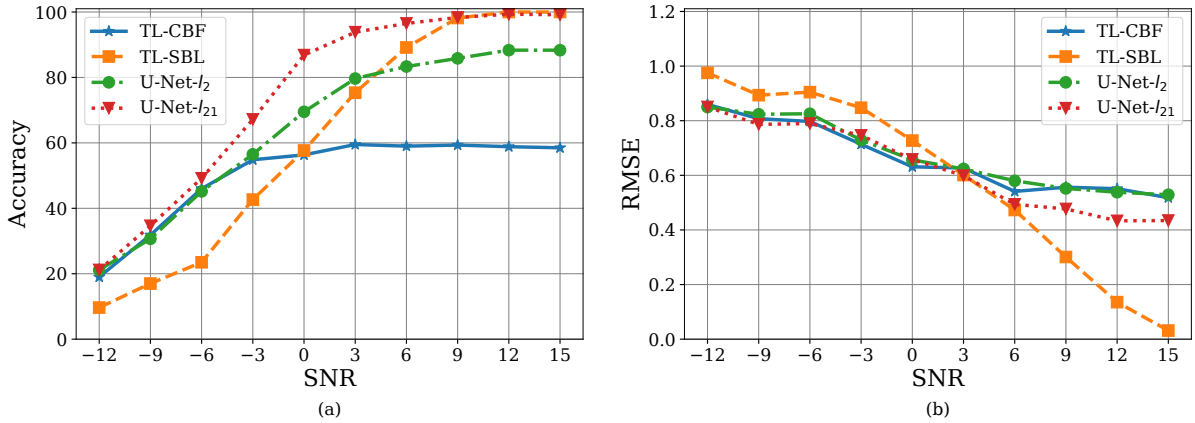

Figure 3.3: Accuracy (a) and average RMSE over detected sources (b) of various algorithms as SNR is changed.

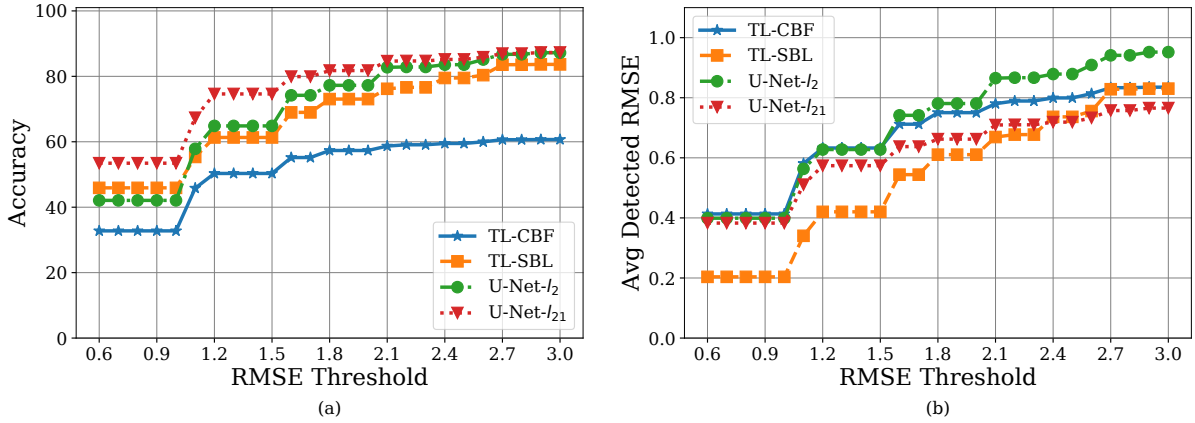

Figure 3.4: Accuracy (a) and average RMSE over detected sources (b) of various algorithms as RMSE threshold (η) varies.

The trajectories for the three sources, labeled S1, S2, and S3, are $(-4,-1)$, $(5,1)$ and $(60,2)$ respectively. TL-CBF's accuracy begins to saturates quite early, indicating that it struggles to resolve nearby sources S1 and S2. Both U-Net methods surpass TL-SBL in the low-to-mid SNR range. U-Net-l_2 reaches saturation at high SNR, while TL-SBL and U-Net-l_{21} approach 100% accuracy. Among the methods, TL-SBL exhibits the lowest accuracy for negative SNR values. As SNR increases, the average RMSE over detected sources for TL-CBF and U-Net variants tends to converge to similar values. TL-SBL have highest average RMSE over detected sources for SNR $\in [-12, 0]$ and have lowest value for SNR $\in [3, 15]$. Notably, this experiment demonstrates the ability of learned networks to generalize accurately to identify more sources ($K = 3$) than encountered during training ($K = 2$), without additional training. It's important to note that the deep learning-based algorithms undergo training using datasets featuring positive SNR. The superior performance of U-Net-l_{21} over U-Net-l_2 can be attributed to the l_1 regularization term in (3.4), which contributes to sharper output spectrum. In Figure

3.4, the accuracy and average RMSE of detected sources for various algorithms are depicted as a function of different threshold values (η), ranging from 0.6 to 3.0. As η increases, both the accuracy and the average RMSE over detected sources for all methods tend to increase. Figures 3.5, 3.6, 3.7, and 3.8 displays the output spectra of various methods at -3, 0, 3, and 6 dB SNR respectively, providing a visual representation of their performance. These findings highlight the robustness and generalization capabilities of U-Net-based methods, particularly U-Net-l_{21}, in resolving and accurately identifying trajectories even in challenging, low SNR scenarios.

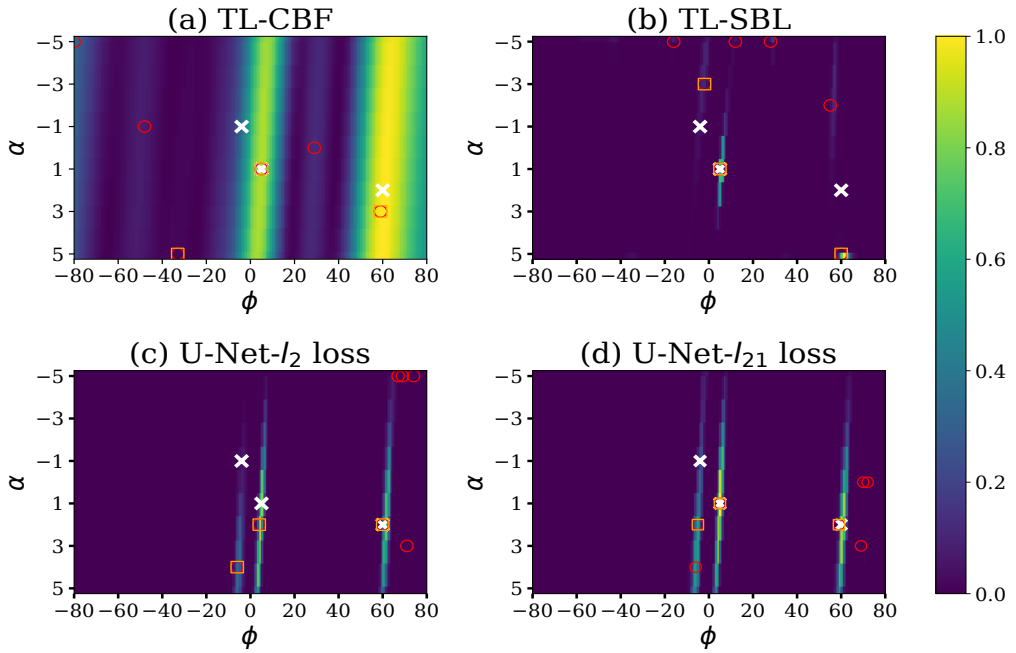

Figure 3.5: Spectrum from various algorithms for -3 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(-4, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively.

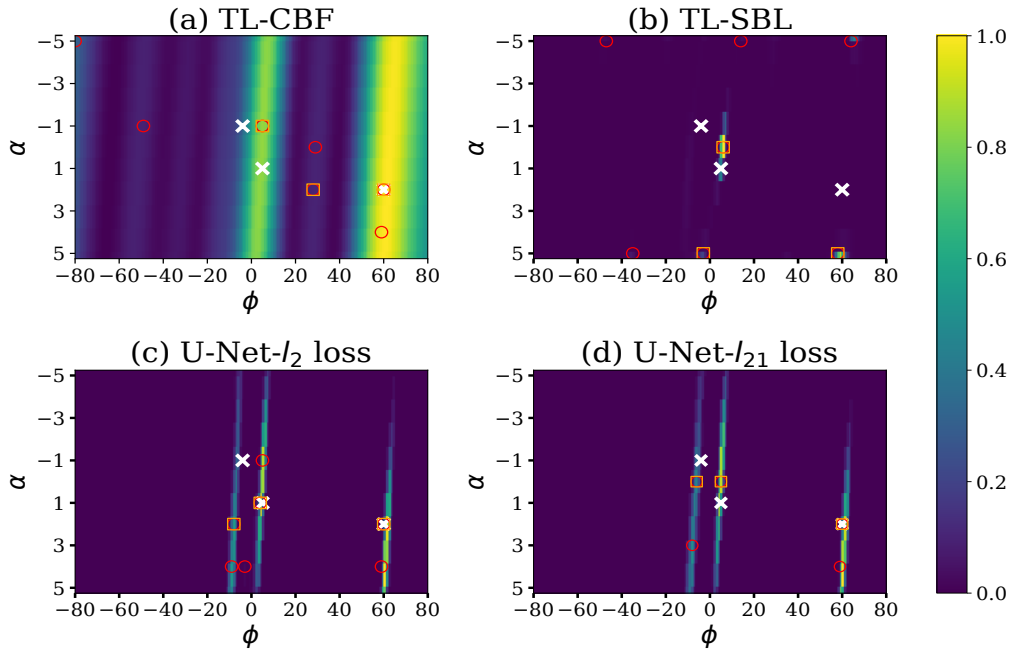

Figure 3.6: Spectrum from various algorithms for 0 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(-4, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively.

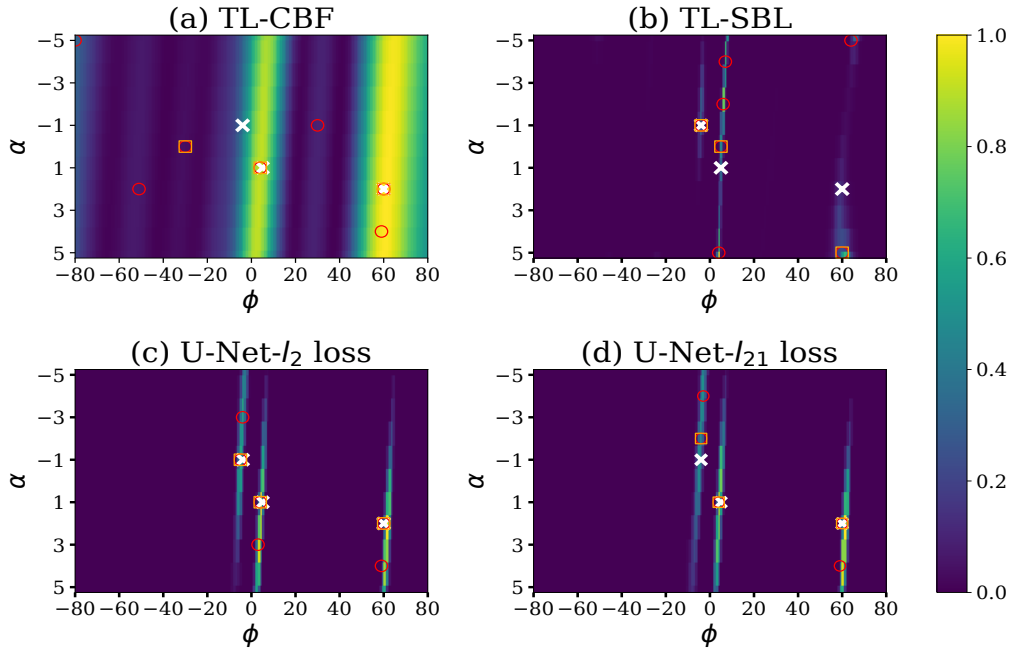

Figure 3.7: Spectrum from various algorithms for 3 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(-4, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively.

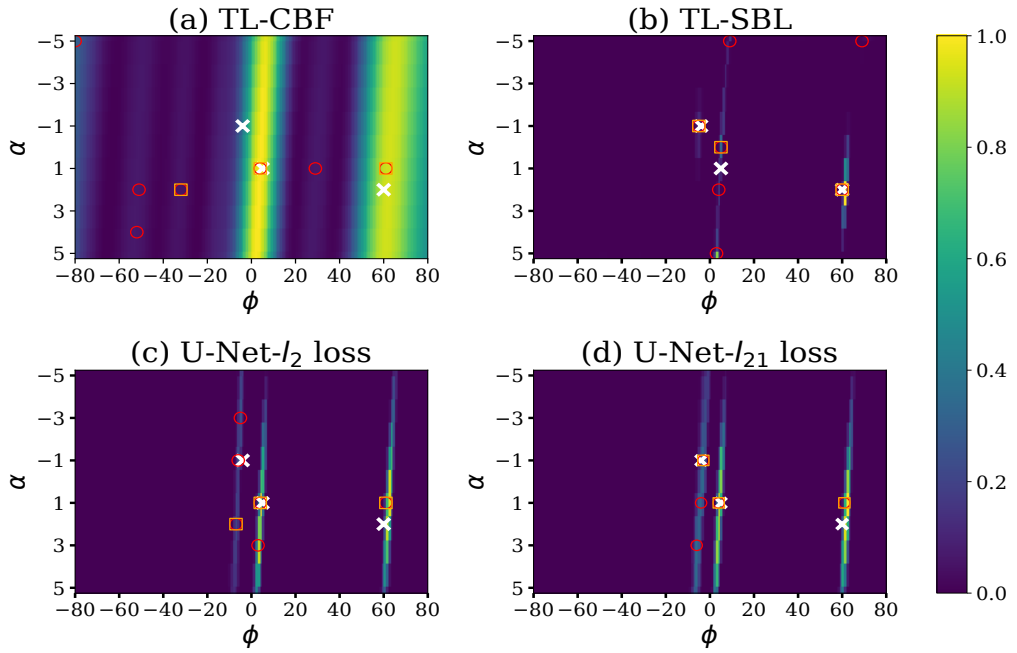

Figure 3.8: Spectrum from various algorithms for 6 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(-4, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively.

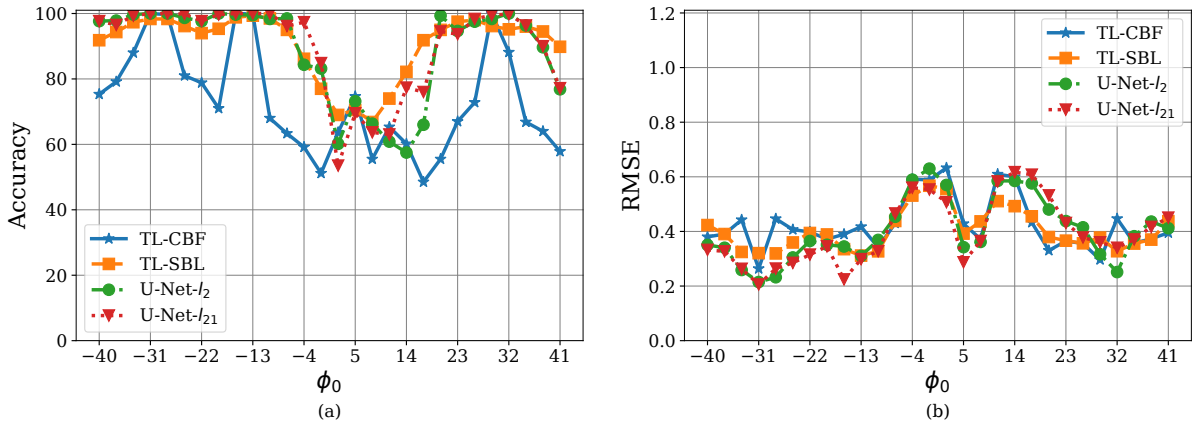

Figure 3.9: Accuracy (a) and average RMSE (b) vs ϕ_0. Two source trajectories are fixed to be $(60, 2)$ and $(5, 1)$ while the third source trajectory is $(\phi_0, -1)$.

TestData-2: In this experiment, we aim to assess the discriminatory capabilities of trajectory localization algorithms in scenarios where source trajectories are in close vicinity. The experimental setup involves two sources S1 and S2 having fixed source trajectories given by $(60, 2)$ and $(5, 1)$ respectively while the third source S3 has a trajectory $(\phi_0, -1)$ with ϕ_0 varying from -40 to 41. The average accuracy of TL-SBL and U-Net variants is approximately same, while TL-CBF have lowest accuracy. The U-Net with l_{21} loss has the lowest average RMSE compared to other methods. Accuracy and average RMSE over detected sources as a

function of ϕ_0 is shown in Figure 3.9, offers insightful observations regarding the algorithms' performance. As ϕ_0 increases, initially S3 comes closer to S2 and then moves away from it. When the three sources are well separated (for example, $\phi_0 \in [-40, -10] \cup [20, 41]$), TL-SBL and U-Net variants perform well by accurately detecting the distinct trajectories. The accuracy of TL-CBF varies considerably when the sources are farther apart. However, as the trajectory of S3 approaches that of S2 (for example, $\phi_0 \in [-10, 20]$), all methods show reduced accuracy with TL-CBF having the least accuracy exhibiting the least resilience to the challenging scenario of closely spaced trajectories. Remarkably, the TL-SBL, U-Net with l_2 loss, and U-Net with l_{21} loss are able to detect the sources with almost the same accuracy throughout. In Figure 3.9, average RMSE over detected sources of all algorithms follow the same trend. In Figure 3.4, the accuracy and average RMSE over detected sources for various algorithms are depicted as a function of different threshold values (η), ranging from 0.6 to 3.0. Both accuracy and average RMSE over detected sources increases as η increases. Figures 3.11, 3.12, 3.13, 3.14, 3.15, and 3.16 displays some of the output spectra of various methods as source S3 approaches to S2 and then moves away from source S2, offering a visual representation of their performance.

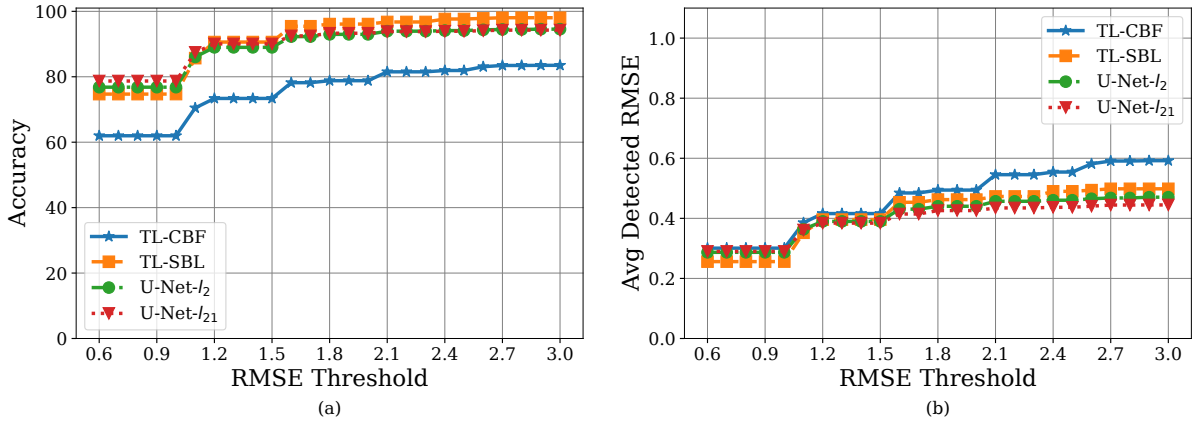

Figure 3.10: Accuracy (a) and average RMSE over detected sources (b) of various algorithms as RMSE threshold (η) varies.

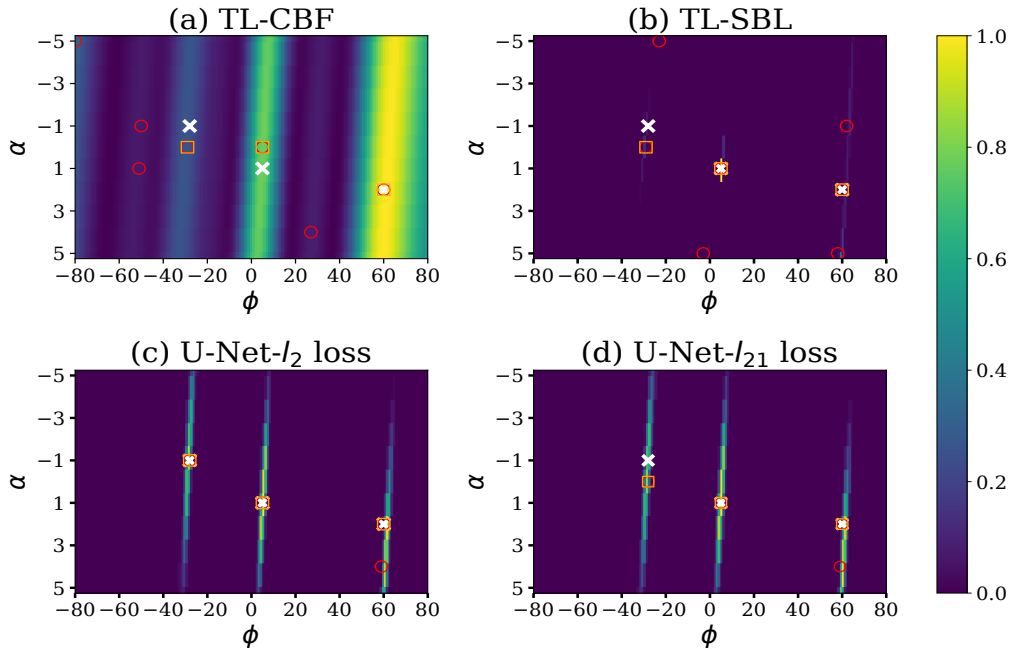

Figure 3.11: Spectrum from various algorithms for 5 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(-28, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively.

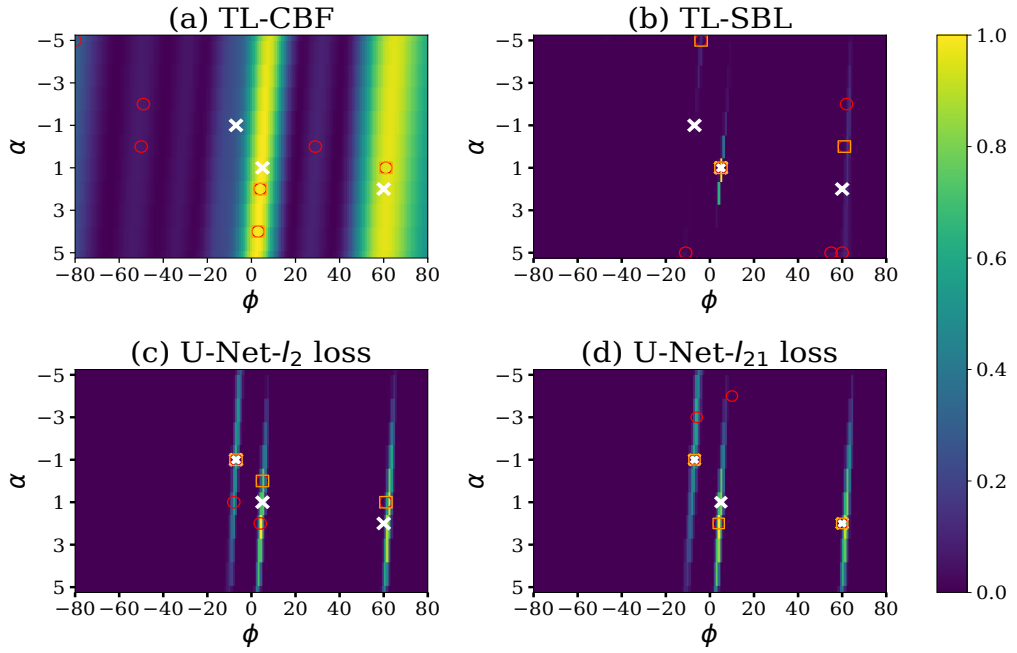

Figure 3.12: Spectrum from various algorithms for 5 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(-7, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively.

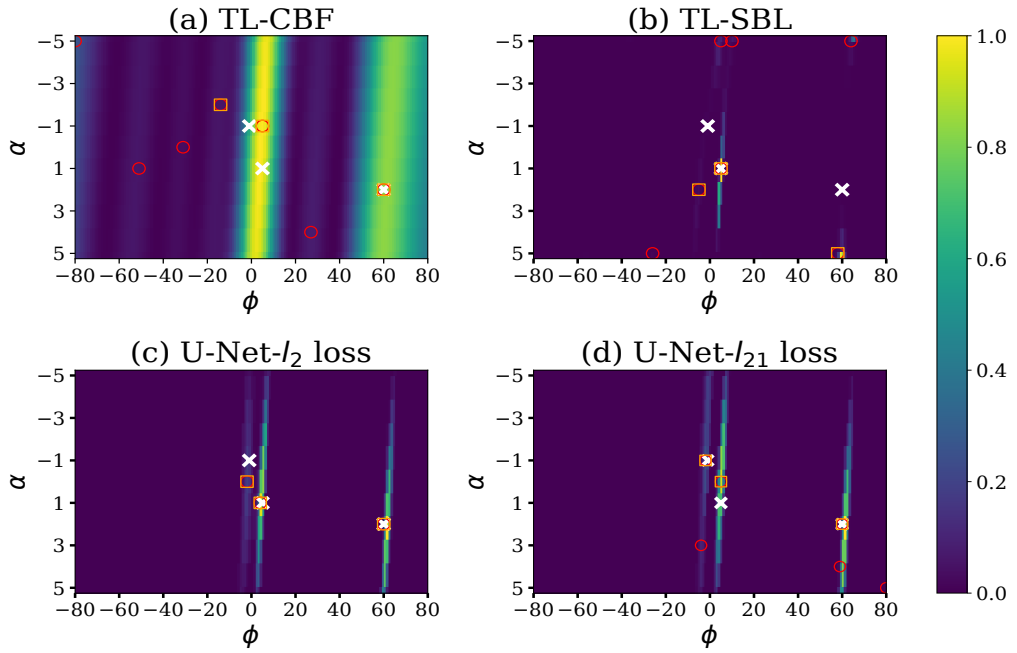

Figure 3.13: Spectrum from various algorithms for 5 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(-1, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively.

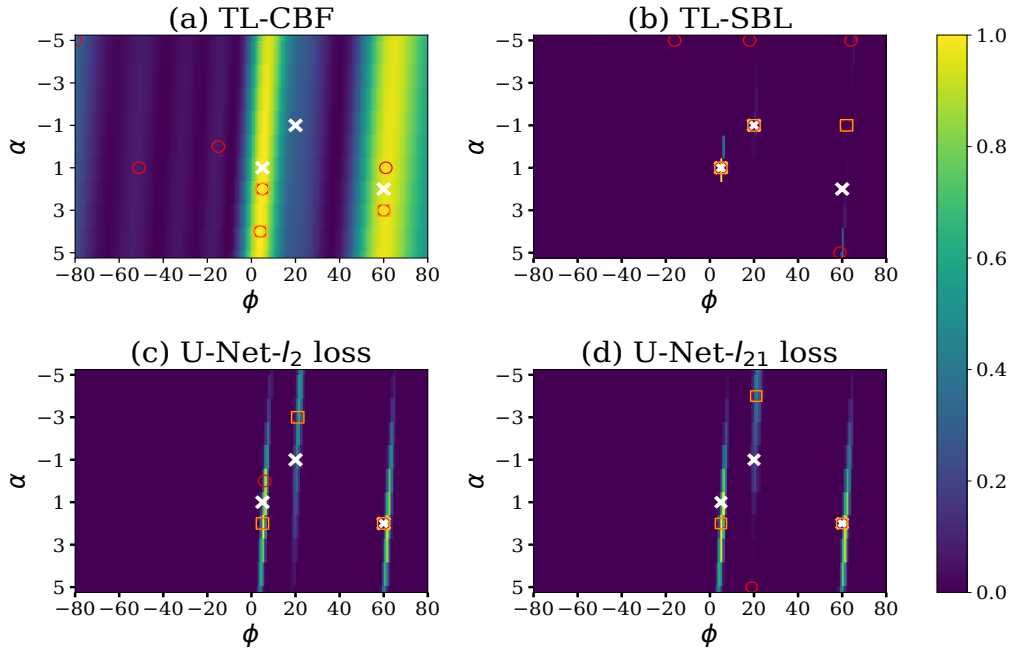

Figure 3.14: Spectrum from various algorithms for 5 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(20, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively.

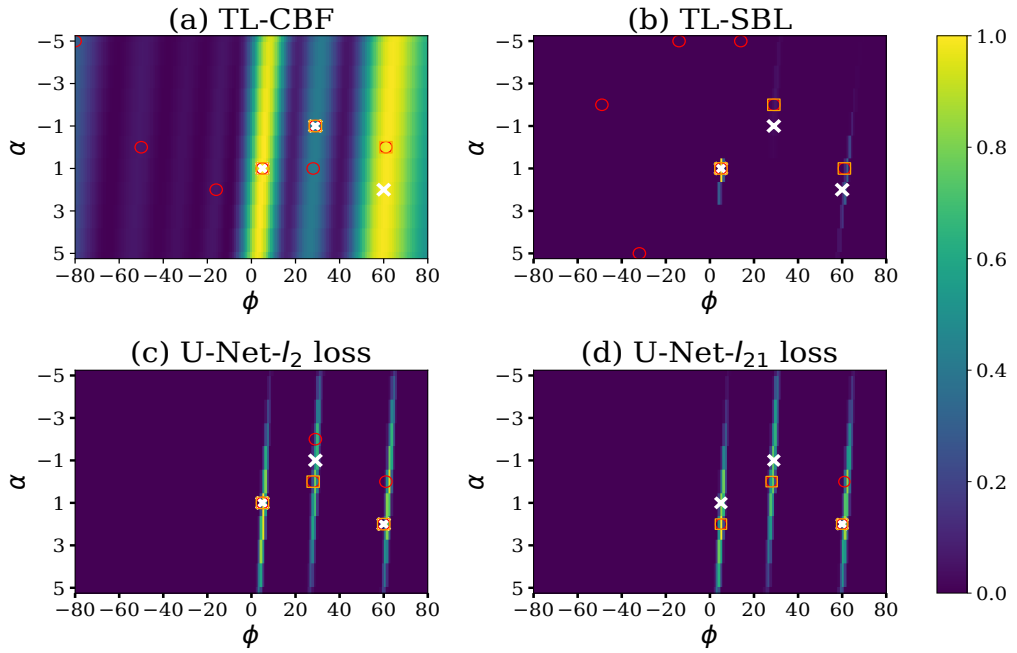

Figure 3.15: Spectrum from various algorithms for 5 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(29, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively.

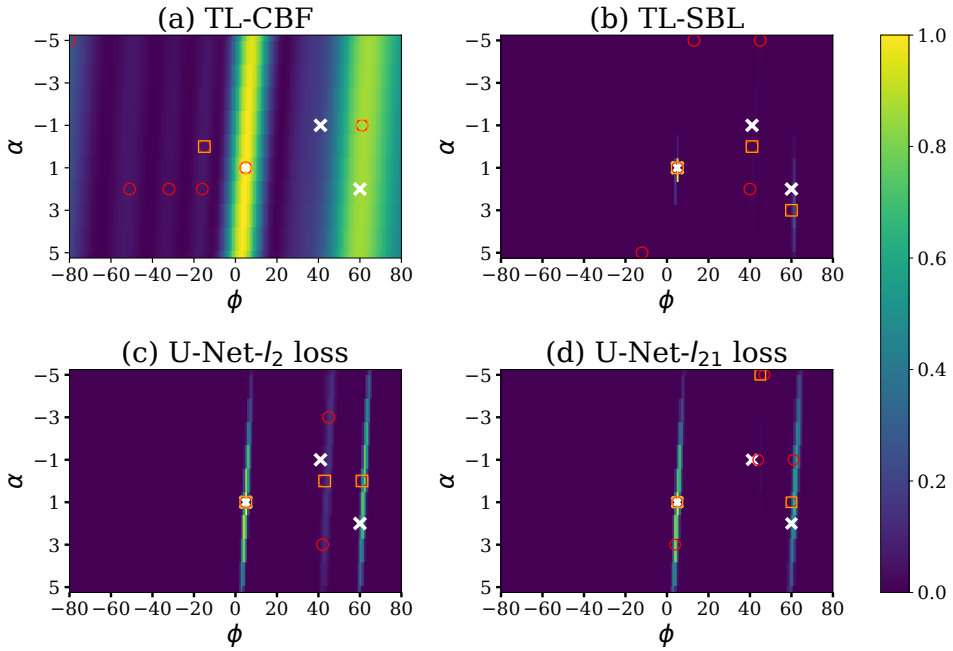

Figure 3.16: Spectrum from various algorithms for 5 dB SNR. Source trajectories are $(60, 2)$, $(5, 1)$, and $(41, -1)$ [white cross]. Detected and assigned peaks are shown by circle and square markers respectively.

3.4 Summary

In this chapter, we leap from previous conventional based methods to deep learning based methods for trajectory localization. We propose U-Net architecture for DOA trajectory estimation using RMSE-based target map which takes low resolution TL-CBF spectrum as input and outputs a refined spectrum allowing precise identification of source trajectories leading to better track estimates. The U-Net architecture demonstrates impressive generalization, adapting to scenarios with varying numbers of sources not encountered during training. We also report the comparison of our proposed method with other baseline methods.

Chapter 4

DOA Trajectory Localization using Complex Deep Nets : One Source at a time

In the pursuit of accurate DOA trajectory parameter estimation, researchers have explored diverse methodologies. Variants of Conventional Beamforming (CBF) [11] and Sparse Bayesian Learning [52] have been devised for this purpose, as highlighted in [39]. These methods represent classical approaches that leverage signal processing techniques for extracting crucial information about DOA. Additionally, a novel approach involves leveraging a U-Net architecture [26] for DOA trajectory parameter estimation task is discussed in the previous chapter. The U-Net architecture, known for its proficiency in image segmentation and feature extraction, is adapted to address the specific challenges associated with low-resolution TL-CBF. This adaptation capitalizes on the U-Net's inherent strengths in capturing intricate spatial patterns and contextual information from input data. This utilization of the U-Net architecture represents a significant advancement in the field, leveraging deep learning techniques to overcome limitations associated with conventional methods. By applying this deep learning technique, the refinement and enhancement of the TL-CBF spectrum contribute to the improvement of DOA trajectory parameter estimation.

But all these methods share a common limitation: they estimate parameters over a grid, which restricts their resolution capability. The grid-free methods proposed in [40] involve sequential identification and correction of source trajectory parameters. This departure from the grid-based approach allows grid-free methods to provide a more fine-grained and accurate estimation of parameters, offering superior resolution in scenarios where precision is critical. Acknowledging this, we have developed data-based methods that are both *grid-free* and *reusable*. A deep complex [41, 53] regression network is proposed to achieve grid-free DOA trajectory parameter estimation. The network outputs are complex signal amplitude (per snapshot) and trajectory parameters for a single source. After estimating parameters for one source, a residual signal is derived by subtracting the contributions of this estimated source from the input data. This residual signal is then fed back into the same network to identify additional sources iter-

atively. This iterative process continues until a predefined stopping criterion is met, allowing the network to be reusable and independent of the number of sources to be estimated. This unique characteristic enhances adaptability and efficiency in source estimation using the same network architecture. Notably, the number of network outputs does not depend on the number of sources, addressing a common issue when attempting to estimate all sources simultaneously. The concept of a reusable network draws inspiration from residual-based processing, a technique commonly employed in greedy methods like matching pursuit (MP) [42, 43]. It aims to decompose a given signal into a linear combination of elementary waveforms, typically called atoms, selected from a predefined dictionary. It is commonly used in applications such as signal denoising, compression, feature extraction, source parameter estimation and sparse representation of data.

4.1 Why complex-valued neural networks (CVNNs)?

Deep complex networks denote neural networks that employ complex-valued inputs, weights and activations, as opposed to real numbers, in their operations and representations. This approach expands the network's capability to handle complex-valued data, allowing for a more refined and expressive representation of information. By incorporating complex numbers, these networks can effectively model and process signals with both magnitude and phase information, making them particularly advantageous in applications involving waveforms, signals, and domains where the interplay of amplitude and phase is critical [54]. Thus, utilization of complex numbers in the network's architecture enhances its ability to capture and analyze intricate relationships within the data, offering a more comprehensive representation compared to networks solely relying on real numbers.

Complex-valued neural networks (CVNNs) [53] diverge from real-valued deep neural networks (DNNs) in several key aspects:

1. **Parameter representation:** CVNNs utilize complex numbers as parameters, which means that each parameter is represented by a pair of real numbers corresponding to real and imaginary parts. This doubles the number of parameters compared to a real-valued network with the same architecture. However, it also enhances the expressive capability of the network, allowing it to model and process complex-valued data more effectively.

2. **Complex multiplication:** The use of complex multiplication in CVNNs introduces a distinct impact on the network's degrees of freedom compared to real-valued multiplication.

3. **Activation functions:** Designing and implementing complex-valued activation functions pose a greater challenge as it must be both complex-differentiable and bounded, which is not a trivial task.

4. **Training complexity:** Training CVNNs can be more complex due to challenges in designing activation functions and dealing with the increased number of parameters. Additionally, the majority of deep learning libraries are optimized for real-valued operations, making the implementation of CVNNs more challenging [55].

5. **Domain-specific applications:** CVNNs excel in areas where complex numbers naturally appear or are intentionally introduced. They are particularly beneficial in fields like wireless communication, audio processing, and signal processing, where CVNNs can effectively capture the inherent correlation between the real and imaginary components of complex numbers [56].

In conclusion, while real-valued neural networks remain versatile and widely used, CVNNs offer a specialized and powerful alternative, providing enhanced capabilities for tasks that require handling complex data and exploiting the unique properties of complex numbers [57]. Their enhanced ability to understand complex-valued data structures can improve performance in certain applications. However, it's important to note that CVNNs come with challenges in training and implementation.

4.2 Network Architecture

Drawing inspiration from the achievements in deep complex networks [41, 53, 57] and their subsequent applications [54, 58, 59, 60], we propose a deep complex network architecture for grid-free DOA trajectory parameter estimation. It is designed to perform estimations of the source amplitudes and the DOA trajectory parameters, focusing on one source at a time. The proposed network architecture is build using Inception [61] and ResNet [62] as its fundamental building blocks. The motivation for using Inception and ResNet blocks lies in their proven effectiveness in handling complex features and mitigating common issues encountered in deep neural networks.

- **Inception blocks** facilitate efficient feature extraction across multiple scales by employing parallel convolutions with varying kernel sizes. This enables the network to capture both local and global features effectively, enhancing its capacity to learn hierarchical representations and improve overall model performance.

- **ResNet blocks** mitigate the vanishing gradient problem in deep neural networks by introducing skip connections that bypass certain layers. This promotes efficient gradient flow

during training, facilitating more effective learning, especially in deeper architectures. As a result, it achieve faster convergence and enable the training of deeper networks without performance degradation.

The overall network architecture is shown in Figure 4.1. It consists of three main parts – shared feature encoder (shaded pink), amplitude decoder (shaded blue), and trajectory decoder (shaded peach). The shared feature encoder generates an encoding for the input data $\mathbf{Y}$, serving as input for the subsequent decoder modules. The amplitude decoder estimates the source amplitudes $\mathbf{X}$, while the trajectory decoder estimates the trajectory parameters (ϕ, α). Notably, the trajectory decoder utilizes the long short-term memory (LSTM) block to effectively extract the sequential information [63]. The architectural design encompasses a total of 548,152 parameters. We conducted experiments with different configurations for skip connections in our architecture, including no skip connection, skip connection only to the trajectory decoder, and separate skip connections to both amplitude and trajectory decoders. However, these configurations did not perform well in comparison to the proposed architecture. Details of the network are discussed next.

Complex-valued convolution Blocks: Complex-valued convolution can be described as follows. Let $\mathbf{W} = \mathbf{A} + i\mathbf{B}$ be the complex-valued convolutional filter characterized by real-valued matrices $\mathbf{A}$ and $\mathbf{B}$. The input to the convolutional block is a complex matrix $\mathbf{H} = \mathbf{P} + i\mathbf{Q}$ is the input to the convolution block. The complex-valued convolution is mathematically formulated as $\mathbf{W} * \mathbf{H} = (\mathbf{A} * \mathbf{P} - \mathbf{B} * \mathbf{Q}) + i(\mathbf{A} * \mathbf{Q} + \mathbf{B} * \mathbf{P})$ where $*$ denotes the convolution operator. This formula unites real and imaginary elements, revealing the convolution operations that encompass the real components $\mathbf{A} * \mathbf{P} - \mathbf{B} * \mathbf{Q}$ and the imaginary components $\mathbf{A} * \mathbf{Q} + \mathbf{B} * \mathbf{P}$. Figure 4.2 provides a visual representation of the complex convolution operation, aiding in a clearer comprehension of the process.

4.2.1 Shared Feature Encoder

The shared encoder block, highlighted in pink in Figure 4.1, integrates complex convolutional blocks, followed by alternate sets of two complex inception blocks and complex convolutional downsampling. In each fundamental complex convolutional block, there is a combination of a complex convolutional layer, a complex batch normalization layer, and a real-valued ReLU acting as the non-linear activation function. To achieve comprehensive information processing across multiple scales, complex inception blocks are utilized. These blocks are designed using five unique complex convolutional blocks featuring distinct kernel sizes: (1, 1), (1, 3), (3, 1), (3, 3), and (5, 1). This ensures effective feature capturing and integration by enabling the model to adapt to a wide range of receptive field sizes. The output channel numbers for two different inception blocks are 6, and 8 for each kernel size (1, 1), (1, 3), (3,

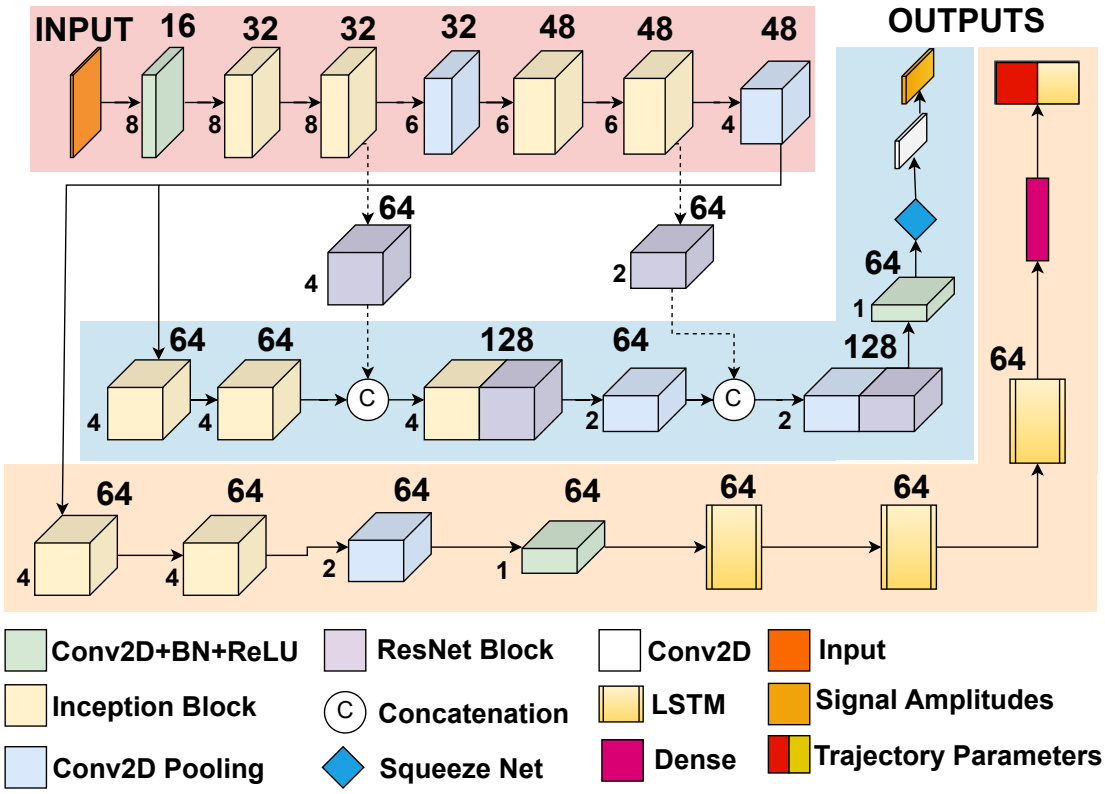

Figure 4.1: Deep Nets architecture. Number above the blocks represents the number of channels for convoluitional block and hidden size of LSTM. Additionally, the numbers beside each block signify one of the dimensions that decreases as the data progresses through each layer. Various components of the network are – shared feature encoder (shaded pink), amplitude decoder (shaded blue), and trajectory decoder (shaded peach).

1), and (5, 1), and 8 and 16, for the (3, 3) kernel size. The intermediate channel numbers for each inception block and for each kernel size are double of the respective output channel numbers. With a stride parameter set to 1, padding is adjusted according to the kernel size to maintain identical input and output sizes. This configuration ensures that information is processed thoroughly without altering the spatial dimensions during the convolutional operations [64]. Instead of traditional pooling layers, we employ dilated complex convolution layers for downsampling the input. This alternative approach is chosen to preserve essential information, as dilated convolutions allow for an expanded receptive field without compromising on the input's crucial details. Pooling layers conventionally decrease the dimensionality of feature maps by employing max or average operations over non-overlapping regions, often resulting in the loss of intricate details. For this operation, the specified parameters are as follows: kernel size

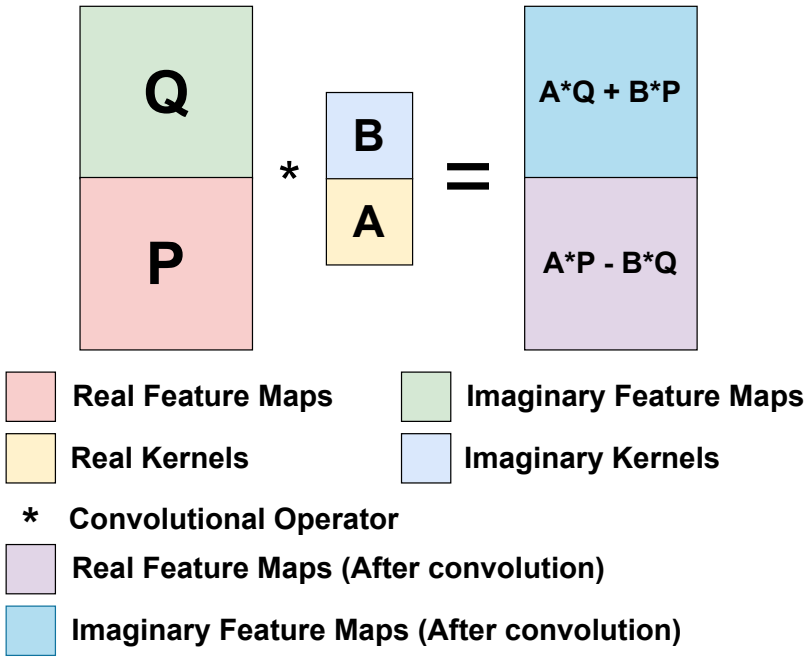

Figure 4.2: An illustration of the complex convolution operator.

(3, 1), dilation rate (2, 1), stride (1, 1), and padding (1, 0). These parameters govern the behavior of the dilated complex convolution layer, influencing the receptive field and downsampling characteristics.

4.2.2 Decoder

The output generated from the shared encoder is directed towards dedicated decoders, namely the *amplitude decoder* and the *trajectory decoder*. These decoders are designed for the purpose of estimating signal amplitudes and trajectory parameters, respectively. Within both the amplitude decoder and trajectory parameters decoder, the initial segment incorporates a set of two complex inception block, configured with hyperparameters similar to those described earlier. For these inception blocks, the output channel numbers for kernel sizes (1, 1), (1, 3), (3, 1), and (5, 1) are set to 10, while while the output channel number for the (3, 3) kernel size is 24. The intermediate channel numbers are exactly double of the corresponding output channel numbers for each kernel size. Following this, the output obtained from the inception block within the amplitude decoder is concatenated (via a first skip connection as shown in Figure 4.1) with the high-level features derived from the shared encoder by processing it using complex ResNet blocks. Subsequently, this fusion undergoes a down-sampling process enabled by dilated com-

plex convolutional blocks. The result of this final down-sampling stage is concatenated (via a second skip connection as shown in Figure 4.1) with an alternative set of high-level features that have been processed through another complex ResNet block. This combined output then undergoes further processing using the complex convolutional blocks with a kernel size of $(2, 1)$, a stride of 1 and zero padding. For amplitude decoder, this output is processed through Squeeze-and-Excitation (SE) [65] block, followed by final complex convolution having kernel size 1. The SE block encompasses two primary operations: squeezing and excitation. Squeezing aggregates feature maps across their spatial dimensions to generate a channel descriptor. Excitation takes this channel descriptor as input and generates a set of per-channel modulation weights. These weights are then applied to the feature maps to produce the SE block's output. Meanwhile, in the trajectory decoder depicted in Figure 4.1, there is no skip connection. This decoder integrates an LSTM block to extract sequential information. This information is then passed through a fully connected dense layer followed by the Tanh non-linear activation function. The LSTM layer is configured with a hidden size of 64, and a stack of three such layers is employed.

4.2.3 Loss functions

The following loss function is used for training of the network

$$L = \sum_{l=1}^{L} |\hat{x}_l - x_l|^2 + \sum_{l=1}^{L} |\hat{\theta}^l - \theta^l|^2 + \sum_{l=1}^{L} |\tilde{\theta}^l - \theta^l|^2 + \sum_{l=1}^{L} |\mathbf{a}(\hat{\theta}^l)\hat{x}_l - \mathbf{a}(\theta^l)x_l|^2 \tag{4.1}$$

where x_l and θ^l are the true source amplitude and DOA values of strongest source at the l-th snapshot, $\hat{x}_l$ is the source amplitude estimated by the amplitude decoder, $\hat{\theta}^l$ is the source DOA at the l-th snapshot obtained from parameters $(\hat{\phi}, \hat{\alpha})$ estimated by the trajectory decoder and, $\tilde{\theta}^l$ is the estimated DOA from the LSTM block. The loss is added across all the training examples. To ensure efficient network training, the first two terms of the loss function are adequate as network is designed to estimate source amplitudes per snapshot and the corresponding DOA parameters. However, for faster convergence, the third and fourth terms are also included in the loss function. The motivation behind the inclusion of term 3 in loss function is based on the assumption that obtaining accurate DOA trajectory parameter estimates is more attainable when the output of the LSTM block closely matches the actual DOA trajectory. This alignment is crucial as the dense layer utilizes the LSTM block's output for parameter estimation. Additionally, as the network comprises two separate decoders – the amplitude decoder (for estimating source amplitudes per snapshot) and the trajectory decoder (for estimating trajectory parameters) – adding term 4 to the loss function affects the weights of these decoders. It incorporates errors from both amplitude and trajectory parameter estimation simultaneously, leading to mutual influence between the weights of these two decoders. This mutual influence

accelerates training by enhancing the coordination between the decoders. An experiment excluding term 3 resulted in poor performance. Also training the network solely with the first two terms of the loss function and excluding skip connections highlighted a more pronounced error propagation. This effect arises because the residual signal is reintroduced to the same network iteratively to identify additional sources. Introducing the last two terms of the loss function enhanced precision. Additionally, incorporating skip connections as shown in Figure 4.1 further improved performance. Our network is trained to output the signal amplitudes and trajectory parameters associated with the strongest source. The strongest source is the one with highest energy, i.e., $\sum_{l=1}^{L} |x_l|^2$.

4.3 Experiments and Results

This section explores the methodologies used for generating data, specific implementation details, and performance metrics used. By delving into these aspects, we gain a comprehensive understanding of the experimental setup, how the model architecture was implemented, and how its performance was evaluated. The results are then discussed, providing an understanding of the model's effectiveness.

4.3.1 Data generation

We conducted a performance evaluation of the proposed deep learning model employing synthetically generated data. An 8-element Uniform Linear Array (ULA) with $\frac{\lambda}{2}$ spacing is used and we generate $L = 30$ snapshot data.

Training: We generated multiple two-source ($K = 2$) scenarios, ensuring a minimum separation of 5° in ϕ. The amplitudes of both sources are sampled independently and identically per snapshot. One source's amplitudes are sampled from $\mathcal{N}(0, 1)$, while the other source's amplitudes are sampled from $\mathcal{N}(0, u)$, where u is sampled from a uniform distribution $U(0.5, 1)$. To accommodate varying signal-to-noise ratios (SNR), we introduced 6 uniformly random SNR values in the range of $[0, 20]$ dB for every combination of two-source scenarios. The sensor array data $\mathbf{Y}$ as given in equation 2.4 serves as the network input, while the signal amplitudes and trajectory parameters are used for comparison with the predicted values. This diverse dataset enables the network to adapt to multi-source patterns with varying source trajectory separations and amplitude differences. In total, 72,5160 data points were generated. Subsequently, this dataset was divided into three subsets: the training set (*TrainData*), validation set (*ValData*), and test set (*TestData*). These subsets consisted of 580,128, 72,516, and 72,516 examples, respectively.

Testing: To specifically analyze the model's performance and trends under different conditions, we generate additional test data for two-source and three-source scenarios. Although

the model is trained solely on two-source scenarios, our objective is to showcase its ability to generalize to different scenarios.

1. Test data for two-source ($K = 2$) scenarios:

   (a) **TestData-1**: In this test dataset, we simulate two closely positioned sources, denoted as S1 and S2, with trajectories $(60.4, 3.5)$ and $(50.5, -2.5)$ respectively. The SNR values are varied from 0 dB to 19 dB with a step size of 1 dB. The S1 and S2 source have amplitudes sampled from $\mathcal{N}(0, 1)$ and $\mathcal{N}(0, 0.6)$ respectively. This allows the evaluation of the model's performance across different noise levels when two sources are present.

   (b) **TestData-2**: In this test dataset, we maintain a fixed SNR of 10 dB with two sources present. One source (S1) having a fixed trajectory $(20.4, 3.5)$ and has amplitudes sampled from $\mathcal{N}(0, 1)$. The other source (S2) has a variable trajectory $(\phi_0, -2.5)$, where ϕ_0 is varied in the range of $[-60, 54]$, and its amplitudes are sampled from $\mathcal{N}(0, 0.75)$. This dataset aims to evaluate the model's performance in scenarios where source trajectories are both close and distant from each other.

2. Test data for three-source ($K = 3$) scenarios:

   (a) **TestData-3**: In this test dataset, we include trajectories for two closely positioned sources (S1, S2) and one distant source (S3) with coordinates $(5.5, 1.2)$, $(-7.5, 3.5)$, and $(60.5, 2.5)$ respectively. We vary the SNR from 0 dB to 20 dB in 1 dB increments. The amplitudes for sources S1, S2, and S3 are sampled from $\mathcal{N}(0, 0.8)$, $\mathcal{N}(0, 0.5)$, and $\mathcal{N}(0, 1)$ respectively. This setup allows for the assessment of the model's performance across various noise levels when three sources are present, a scenario that the model has not encountered during training.

   (b) **TestData-4**: In this test dataset, we fix SNR at 5 dB and simulate three sources. Two sources, denoted as S1 and S2, have fixed trajectories $(60.5, 2.5)$ and $(5.5, 1.2)$ respectively, while the trajectory of the third source denoted as S3 is $(\phi_0, -3.8)$, with ϕ_0 varying within the range of $[-60, 54]$. The sources with fixed trajectories have amplitudes sampled from $\mathcal{N}(0, 1)$, while for the third source we sample from $\mathcal{N}(0, 0.5)$. This dataset aims to assess the model's performance in scenarios where source trajectories vary in proximity, ranging from closely positioned to distant. It includes three sources, a scenario not encountered by the model during training.

In each scenario, source amplitudes are sampled per snapshot in an IID manner from their respective distributions. A total of 4000 examples were generated for each test dataset, with 200 examples for each set of varied parameters.

4.3.2 Implementation details

The proposed architecture is implemented using the Pytorch framework [48] and utilizes Kaiming initialization [49] to initialize the network's weights. This initialization technique is particularly suited for deep networks that use asymmetric, non-linear activation functions like ReLU, which is the case in our network. Its purpose is to mitigate problems such as vanishing or exploding gradients during the training process, ensuring more stable and effective learning. The proposed model is trained for 50 epochs with a batch size of 512, a learning rate 10^{-3}, and using an Adam optimizer [50]. A multi-step learning rate scheduler is employed to enhance convergence, recognizing that a fixed learning rate may not be optimal throughout the entire training duration. This involves reducing the learning rate by half at designated milestones, which occurred at epochs 30, 35, 40, 43, 46, and 49. *ValData* is used to select best model, based on its accuracy score. We benchmark our results against the grid-based trajectory localization methods TL-CBF [39], U-Net-l_2, and U-Net-l_{21}. The baseline models U-Net-l_2 and U-Net-l_{21} were trained using the same dataset for 50 epochs. During training, a batch size of 256 examples was utilized, with a learning rate of 10^{-3}, and the Adam optimizer was employed. Additionally, the learning rate was reduced by a factor of 0.1 after 40 epochs to facilitate convergence.

4.3.3 Performance metrics

We evaluate the performance of various algorithms based on accuracy metrics and average Root Mean Square Error (RMSE) over detected sources. For the grid-based baseline methods, we have a low-resolution 2D power spectrum for TL-CBF, along with its high-resolution counterparts which are outputs from U-Net-l_2 and U-Net-l_{21}. The peaks in the 2D grid correspond to the trajectory parameters (ϕ, α) of each source. When K sources are present, $P = K$ peaks are identified in the output spectrum. The decision to select the first K peaks is based on their intensity. The Hungarian algorithm [51] is utilized to solve the assignment problem between the P peaks and K sources. As the proposed network estimates source amplitudes and trajectory parameters one source at a time, we can sequentially use the hungarian algorithm to solve the assignment problem among K sources. Once assigned (for grid-based methods) or estimated (for grid-free methods), a source is considered detected if the RMSE between the true θ_k^{*l} and the assigned/estimated track $\hat{\theta}_k^l$ is less than the threshold η. We choose $\eta = 2.4$. Accuracy is calculated as the ratio of total number of detected sources to the total number of sources present, over the complete dataset. Since the proposed method outputs signal amplitude, which is used to compute the residual, it's crucial to consider the amplitude relative error for the proposed method. Amplitude relative error is a metric used to quantify the difference between the estimated amplitude and the true amplitude, relative to the true amplitude.

4.3.4 Results and discussions

DOA trajectory localization accuracy and average detected RMSE over detected sources is reported in Table 4.1 and 4.2 respectively for the proposed grid-free method and other grid-based baselines.

In terms of accuracy, the proposed method demonstrates highest performance for *TestData*, *TestData-1* and *TestData-2*, all of which represent distinct scenarios for the two-source ($k = 2$) case. For *TestData*, both the proposed method and the U-Net variants exhibit nearly identical performance, while TL-CBF achieves the lowest accuracy. This test dataset comprises various combinations, including scenarios with low and high SNR, source trajectories both close and far apart, as well as varying amplitude levels. In *TestData-1*, where the SNR varies from 0 to 19 dB, TL-CBF demonstrates very poor performance with an accuracy of 16.17. Both the proposed method and U-Net with l_{21} loss achieve similar level of accuracy. U-Net with l_2 loss exhibits an accuracy approximately 10% lower compared to its l_{21} loss counterparts. For *TestData-3* and *TestData-4*, U-Net with l_2 loss and U-Net with l_{21} loss, respectively, exhibit the highest accuracy, with both achieving almost the same level of accuracy. These datasets represent distinct scenarios for the three-source $K = 3$ case. For *TestData-3*, TL-CBF achieves an accuracy of 48.49%, whereas the proposed method achieves nearly 15% higher accuracy but approximately 13% lower than the U-Net variants. In *TestData-4*, the proposed method achieves the accuracy of 65.97%, which is approximately 7% lower than that of U-Net with l_{21} loss (the best performer) and about 4% lower than TL-CBF. Both U-Net variants exhibit almost identical accuracy levels across all test datasets, with a notable 10% difference observed in *TestData-1*.

		Grid-based			Grid-free
S.No.	DatasetName	TL-CBF	U-Net-l_2	U-Net-l_{21}	Proposed
1	TestData	64.79	85.02	85.11	**88.93**
2	TestData-1	16.175	78.48	87.6	**90.87**
3	TestData-2	69.97	70.76	68.76	**95.22**
4	TestData-3	48.49	**76.47**	76.2	63.70
5	TestData-4	57.3	69.63	**72.7**	65.97

Table 4.1: Accuracy (in $\%$) of various algorithms on different test datasets ($\eta = 2.4$).

In terms of average RMSE over detected sources, the proposed method exhibits the highest values for each test dataset except for *TestData-1*. For *TestData-1*, TL-CBF has higest average RMSE of 1.7. U-Net with l_2 loss achieves the lowest average RMSE for *TestData-1* and *TestData-4*. On the other hand, TL-CBF demonstrates the lowest average RMSE for *TestData-*

2 and *TestData-3*. Both U-Net variants demonstrate almost the same average RMSE for each test dataset, with slight differences observed in *TestData-2*.

		Grid-based			Grid-free
S.No.	DatasetName	TL-CBF	U-Net-l_2	U-Net-l_{21}	Proposed
1	TestData	0.57	0.511	**0.510**	1.08
2	TestData-1	1.7	**0.90**	0.94	1.29
3	TestData-2	**0.53**	1.32	0.97	1.05
4	TestData-3	**0.766**	0.904	0.903	1.01
5	TestData-4	0.768	**0.69**	0.71	0.90

Table 4.2: Average RMSE over detected sources of various algorithms on different test datasets.

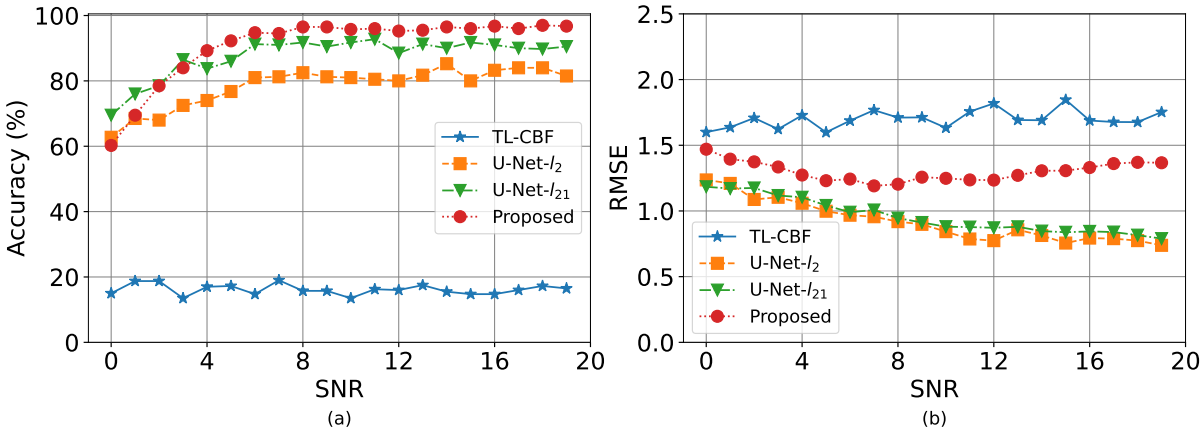

Figure 4.3: [**TestData-1**] Accuracy (a) and average RMSE over detected sources (b) of various algorithms as SNR is changed.

TestData-1: In this experiment, the primary focus is on analyzing the performance of different algorithms for estimating the trajectories of two closely positioned sources. This analysis is conducted across a range of SNR values spanning from 0 dB to 19 dB. Our proposed method exhibits performance similar to U-Net variants, while TL-CBF struggles with resolving nearby source trajectories, leading to lower average accuracy. The poor performance of TL-CBF is attributed to closely positioned source trajectories, where it has difficulty in accurately detecting peaks that are nearer to the actual ground truth source trajectories. It's worth noting that in terms of average RMSE over detected sources, the proposed method exhibits the higher average value compared to U-Net variants. In Figure 4.3, the accuracy and average RMSE over detected sources are plotted as a function of SNR for different algorithms. Both the proposed method and the U-Net variants demonstrate a similar trend of increasing accuracy and decreasing average RMSE over detected sources as SNR increases. The accuracy saturates as SNR reaches higher values for both the proposed method and the U-Net variants. However, for the

proposed method, there's a slight increase in average RMSE over detected sources after an SNR value of 8 dB, whereas for the U-Net variants, the average RMSE continues to decrease as SNR increases. TL-CBF's accuracy and average RMSE fluctuate consistently as SNR increases. Its accuracy consistently remains below 20%, while the average RMSE consistently stays above 1.6. In Figure 4.4, the accuracy and average RMSE over detected sources are plotted against the RMSE threshold (η) for various algorithms. The accuracy and average RMSE over detected sources for both the proposed method and U-Net variants show a smooth increment as η varies from 1.2 to 3.6. Within the range of $\eta \in [1.2, 2.0)$, U-Net with l_{21} loss exhibits the highest accuracy, while for $\eta \in [2.0, 3.6]$, the proposed method achieves higher accuracy. The average RMSE over detected sources remains almost the same for U-Net variants and is the lowest among all methods. For TL-CBF, the accuracy is consistently the lowest across all threshold values, while the average RMSE over detected sources is highest within the range of $\eta \in [2.0, 3.6]$.

Figure 4.5 (a) visually depicts the amplitude relative error for two-source $K = 2$ scenarios as SNR varies. As SNR increases, the amplitude error for both sources S1 and S2 decreases, which is reflected in their average error as well. However, it's worth noting that the relative error for the weaker signal source becomes more pronounced, primarily due to error propagation during residual calculations.

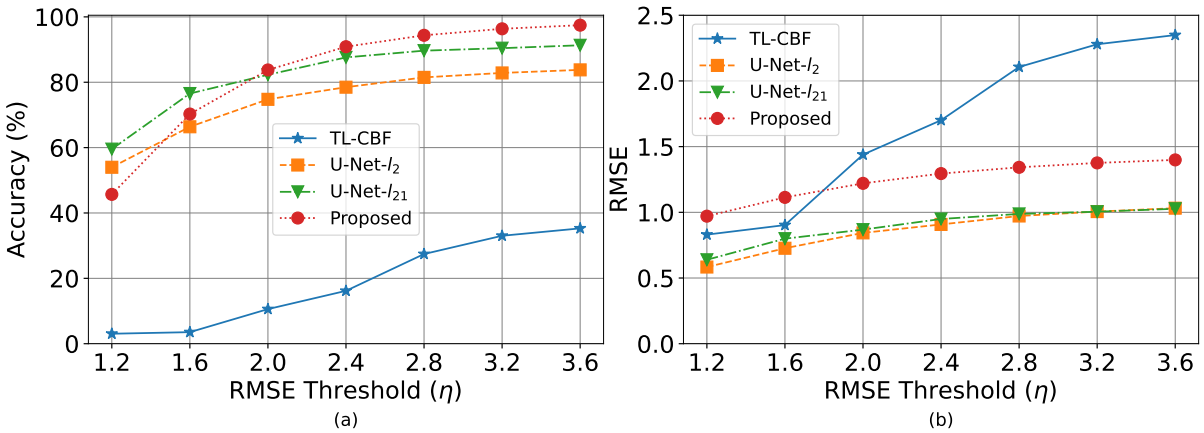

Figure 4.4: [**TestData-1**] Accuracy (a) and average RMSE over detected (b) sources of various algorithms as RMSE threshold (η) varies.

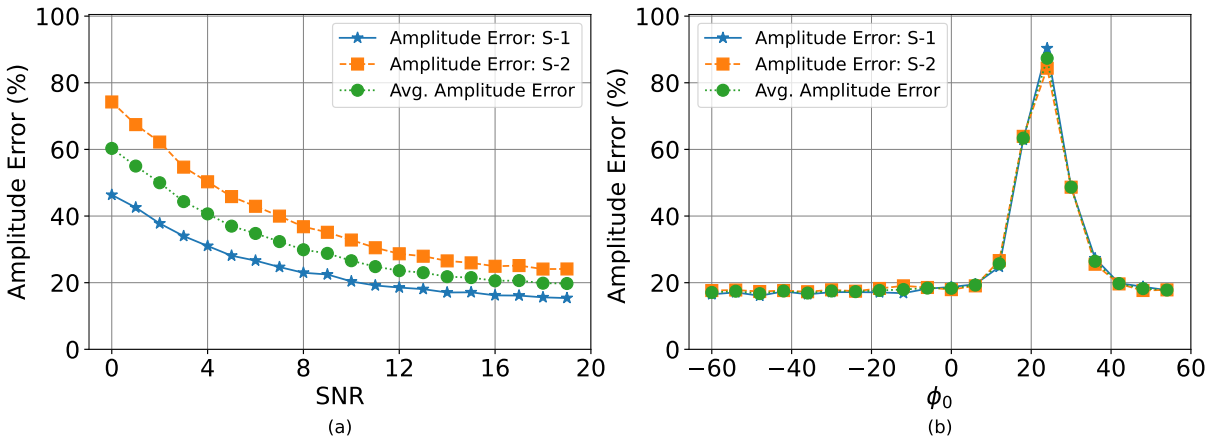

Figure 4.5: Amplitude relative error of the two-source scenarios as SNR (a) and ϕ_0 (b) varies for the proposed method.

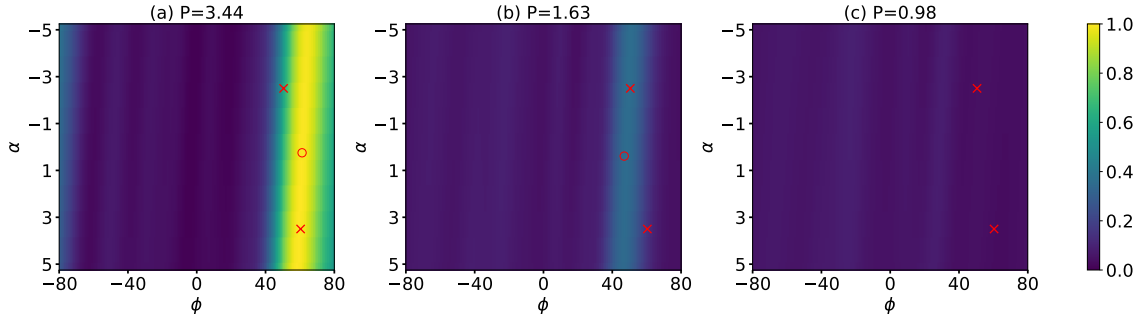

Figure 4.6: [**TestData-1**] TL-CBF spectrums: (a) original signal at 1 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (60.4, 3.5) and (50.5, -2.5) [red cross]. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

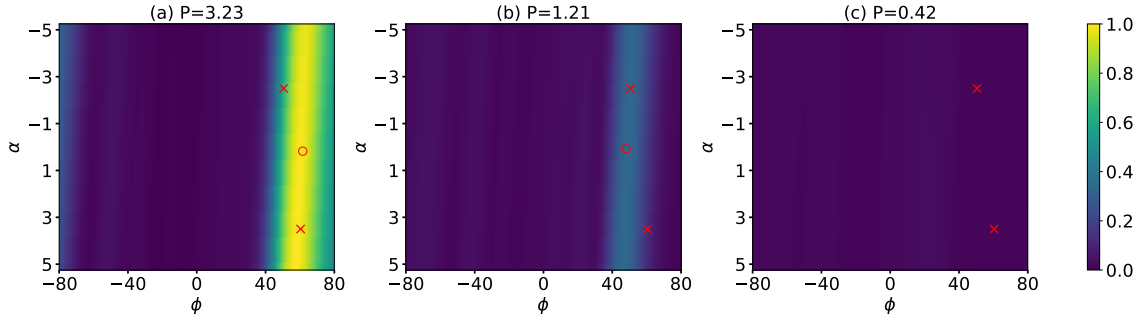

Figure 4.7: [**TestData-1**] TL-CBF spectrums: (a) original signal at 5 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (60.4, 3.5) and (50.5, -2.5) [red cross]. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

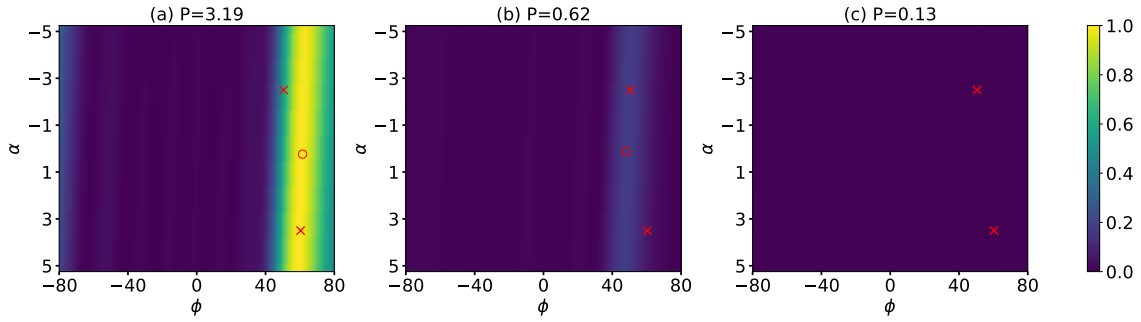

Figure 4.8: [**TestData-1**] TL-CBF spectrums: (a) original signal at 10 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (60.4, 3.5) and (50.5, -2.5) [red cross]. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

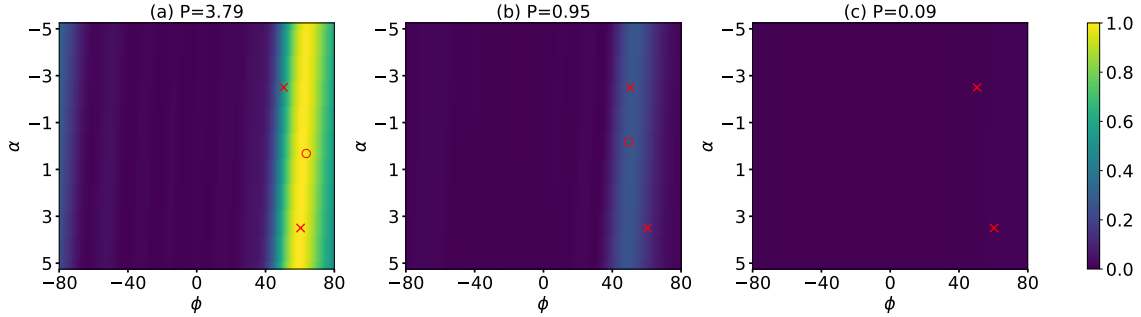

Figure 4.9: [**TestData-1**] TL-CBF spectrums: (a) original signal at 15 dB SNR, (b) after removing first source, (c) after removing second source at 15 dB SNR. Source trajectories are (60.4, 3.5) and (50.5, -2.5) [red cross]. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

Figure 4.6, 4.7, 4.8, and 4.9, displays the TL-CBF spectrums as sources are sequentially removed using proposed method at 1, 5, 10 and 15 dB SNR respectively. From these figures, it is evident that as sources are removed sequentially, the average power per sensor per snapshot (P) decreases. Moreover, as SNR increases, P after removing both sources demonstrates a decreasing trend.

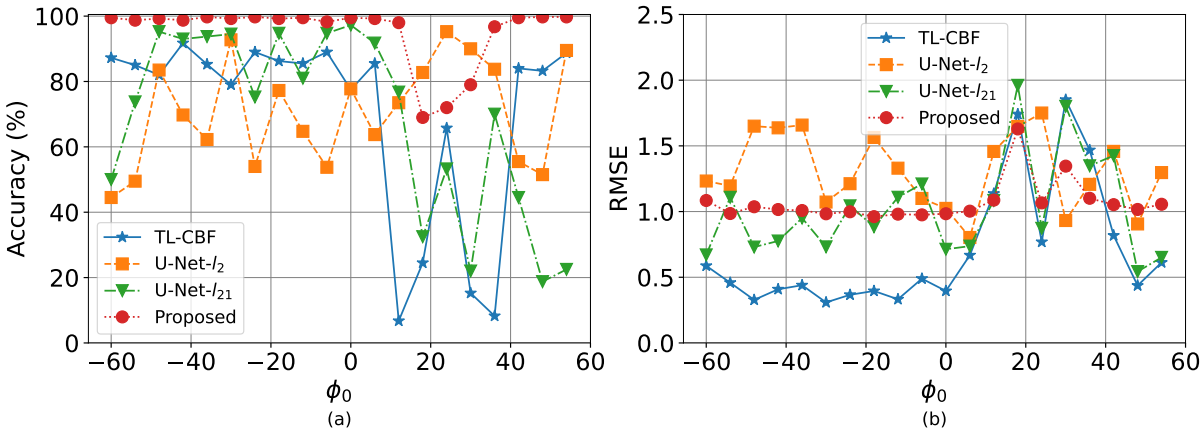

Figure 4.10: [**TestData-2**] Accuracy (a) and average RMSE over detected sources (b) of various algorithms vs ϕ_0. One source trajectory is fixed to be $(20.4, 3.5)$, while the second source trajectory is $(\phi_0, -2.5)$. The parameter ϕ_0 varies with a step size of 6, ranging from -60 to 54.

TestData-2: The primary goal of this experiment is to evaluate the performance of different algorithms under varying source trajectories, including scenarios where the sources are either closely positioned or far apart. Specifically, the experiment involves two sources, denoted as S1 and S2. S1 has a fixed trajectory $(20.4, 3.5)$, while S2 has a trajectory $(\phi_0, -2.5)$ with ϕ_0 varies from -60 to 54. As ϕ_0 increases, S2 moves closer to S1 and then moves away from it. The average accuracy of the proposed grid-free method is highest, while grid-based baseline has approximately same accuracy. The average RMSE over detected sources is minimum for TL-CBF while highest for U-Net with l_2 loss. Figure 4.10 illustrates the average accuracy and the average RMSE over detected sources for different algorithms as a function of ϕ_0. The proposed methods has highest accuracy compared to others when $\phi_0 \in [-60, 12] \cup [36, 54]$, while its performance degraded when $\phi_0 \in (12, 36)$. The U-Net with l_2 loss demonstrates superior accuracy compared to other methods when ϕ_0 ranges from 12 to 36. Within this range, the accuracy of other grid-based methods notably decreases. The average RMSE over detected sources of proposed methods remains relatively similar as ϕ_0 varies except when $\phi_0 \in [12, 36]$. The average RMSE over detected sources of other grid-based methods fluctuates significantly as ϕ_0 varies, with notably large values observed when $\phi_0 \in [12, 42]$. In Figure 4.11, the plot depicts the changes in accuracy and average RMSE over detected sources as η varies. Across different η values, the proposed method consistently achieves the highest accuracy, while TL-CBF consistently exhibits the lowest average RMSE over detected sources. For η values in the range $[1.2, 2.4)$, TL-CBF exhibits higher accuracy compared to U-Net variants, while beyond this range, U-Net variants demonstrate higher accuracy. However, all methods show saturation in accuracy beyond $\eta = 2.8$. Additionally, the average RMSE over detected sources of U-Net with l_{21} loss consistently remains lower than that of the proposed method and U-Net with l_2 loss.

Figure 4.5 (b) illustrates the amplitude relative error as a function of ϕ_0. The amplitude errors of both sources, S1 and S2, remain consistent except when $\phi_0 \in [12, 36]$. During this range, when S1 approaches closer to S2, the amplitude error of S1 escalates, leading to error propagation and hence, an increase in the amplitude error of S2 as well.

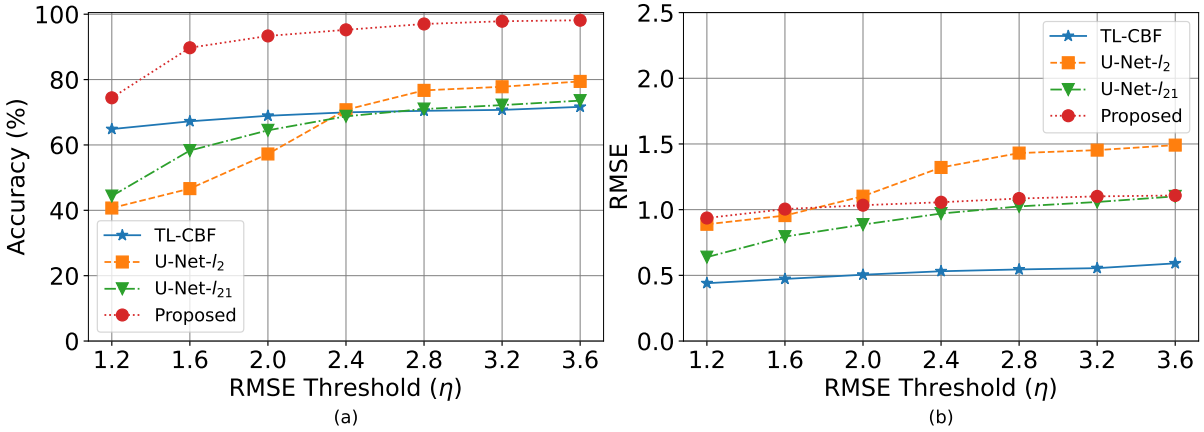

Figure 4.11: [**TestData-2**] Accuracy (a) and average RMSE over detected sources (b) of various algorithms as RMSE threshold varies (η). One source trajectory is fixed to be $(20.4, 3.5)$, while the second source trajectory is $(\phi_0, -2.5)$. The parameter ϕ_0 varies with a step size of 6, ranging from -60 to 54.

Figure 4.12, 4.13, 4.14 4.15, 4.16 and 4.17, illustrate the TL-CBF spectrum as sources are removed using the proposed method at 10 dB SNR, corresponding to ϕ_0 values of -24, 0, 12, 24, 36, and 48, respectively.

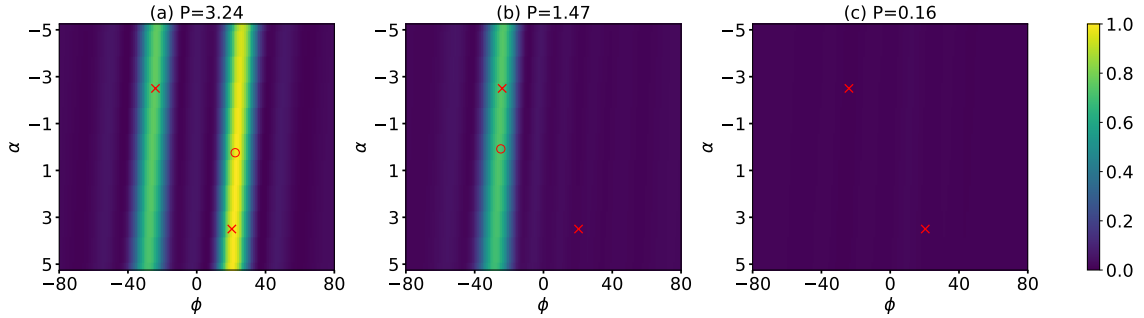

Figure 4.12: [**TestData-2**] TL-CBF spectrums: (a) original signal at 10 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (20.4, 3.5) and (ϕ_0, -2.5) [red cross] where $\phi_0 = -24$. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

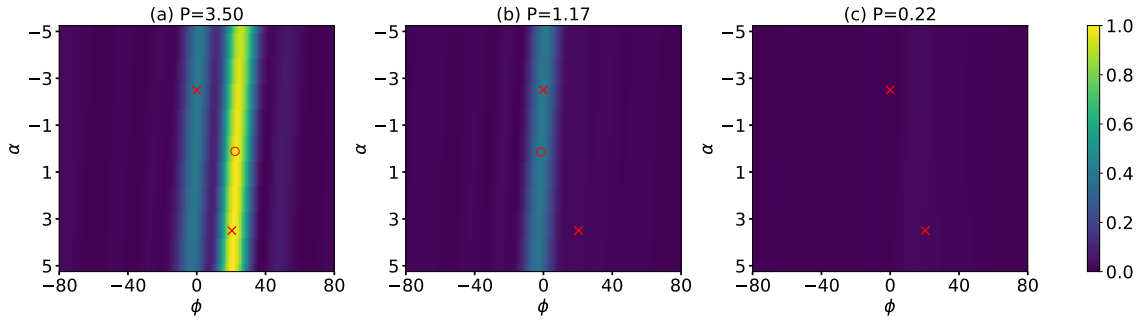

Figure 4.13: [**TestData-2**] TL-CBF spectrums: (a) original signal at 10 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (20.4, 3.5) and (ϕ_0, -2.5) [red cross] where $\phi_0 = 0$. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

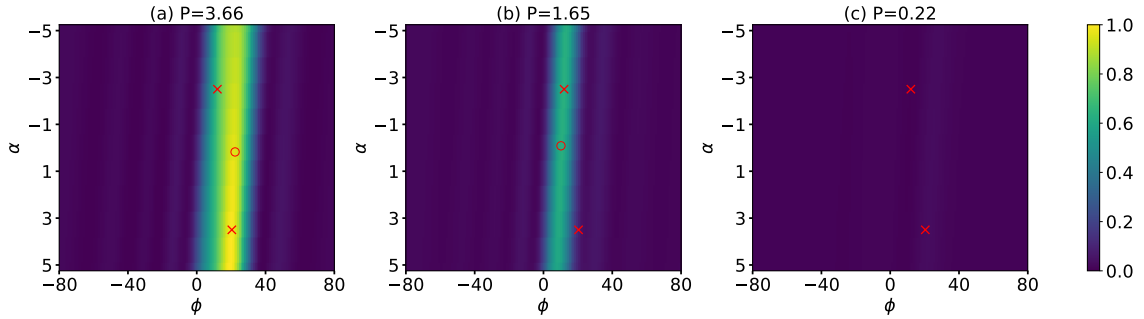

Figure 4.14: [**TestData-2**] TL-CBF spectrums: (a) original signal at 10 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (20.4, 3.5) and (ϕ_0, -2.5) [red cross] where $\phi_0 = 12$. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

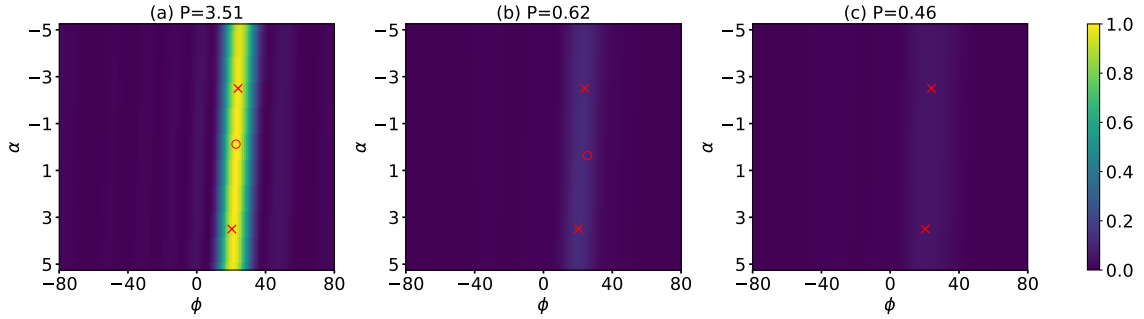

Figure 4.15: [**TestData-2**] TL-CBF spectrums: (a) original signal at 10 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (20.4, 3.5) and (ϕ_0, -2.5) [red cross] where $\phi_0 = 24$. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

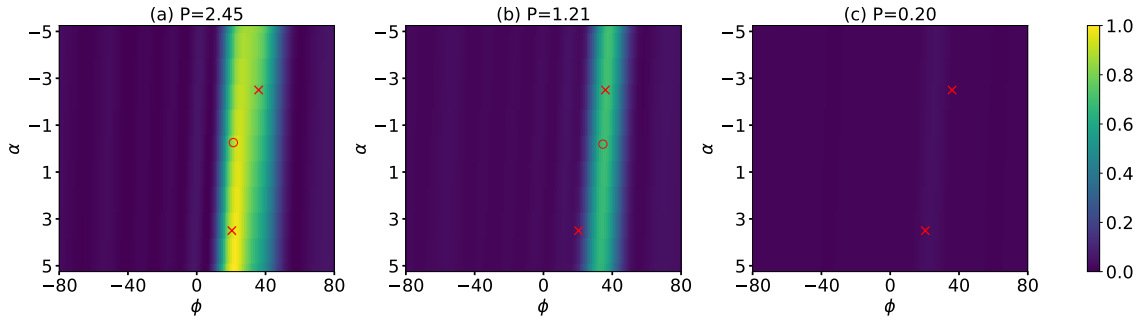

Figure 4.16: [**TestData-2**] TL-CBF spectrums: (a) original signal at 10 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (20.4, 3.5) and (ϕ_0, -2.5) [red cross] where $\phi_0 = 36$. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

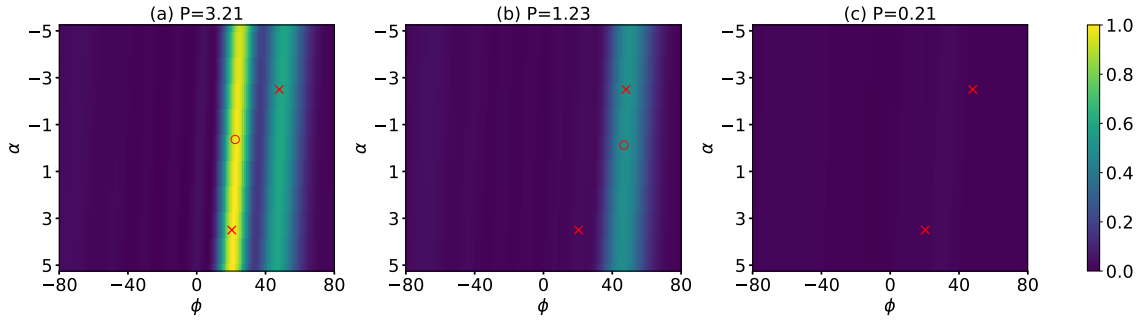

Figure 4.17: [**TestData-2**] TL-CBF spectrums: (a) original signal at 10 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are (20.4, 3.5) and (ϕ_0, -2.5) [red cross] where $\phi_0 = 48$. The estimated source trajectories are indicated by red circle in (a) and (b), which are partially or fully removed in the subsequent spectra (b) and (c) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

TestData-3: In this experiment, we aim to analyze the performance of different methods as the SNR varies from 0 to 19 dB for scenarios involving three sources. This scenario has not been encountered by both the proposed method and the U-Net variants during training. Two of the three sources, S1 and S2, have source trajectories of $(5.5, 1.2)$ and $(-7.5, 3.5)$, respectively, and are closely positioned. In contrast, the third source, S3, has a source trajectory of $(60.5, 2.5)$ and is located far from the other two sources. The U-Net variants exhibit almost identical average accuracy, with U-Net with l_2 loss showing the highest accuracy, while TL-CBF has the lowest accuracy among them. The average RMSE over detected sources is minimum for TL-CBF and maximum for proposed method. Figure 4.18 illustrates the accuracy and average RMSE over detected sources as SNR varies. While the accuracy of each method remains relatively consistent, the average RMSE decreases for all methods as SNR increases from 0 to 19 dB. Throughout this range, the accuracy of U-Net variants consistently

outperforms other methods, while TL-CBF consistently exhibits the lowest accuracy. Figure 4.19 illustrates the accuracy and average RMSE over detected sources as the RMSE threshold (η) varies from 1.2 to 3.6. As the RMSE threshold (η) increases, the accuracy and average RMSE over detected sources of each algorithm also increase. The accuracy and average RMSE over detected sources of U-Net variants remain relatively consistent across different η values. U-Net variants consistently achieve the highest accuracy, while TL-CBF consistently demonstrates the lowest accuracy. Additionally, as η varies, the proposed method consistently exhibits the maximum average RMSE over detected sources, while TL-CBF consistently demonstrates the minimum average RMSE.

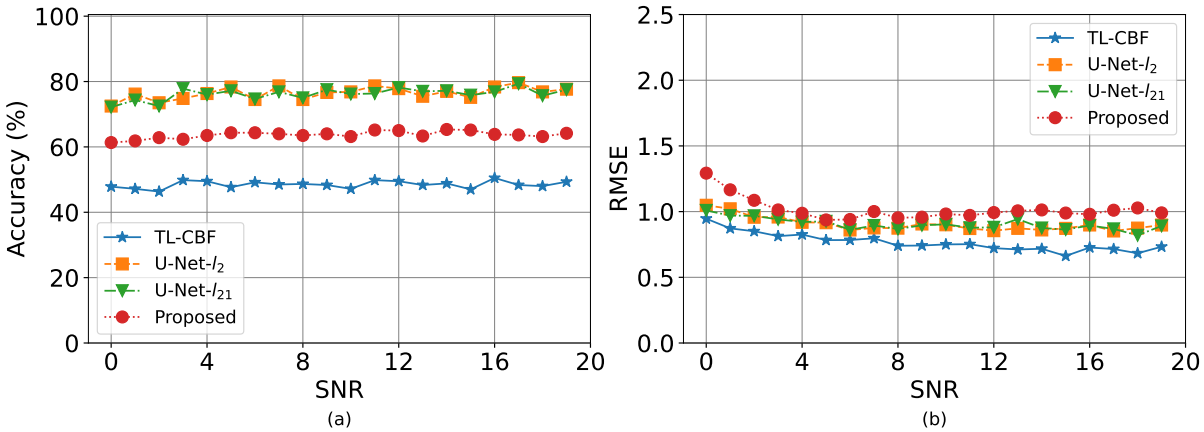

Figure 4.18: [**TestData-3**] Accuracy (a) and average RMSE over detected sources (b) of various algorithms as SNR is changed. Source trajectories are fixed to be $(-60.5, 2.5)$, $(-12.5, 3.5)$ and $(5.5, 1.2)$.

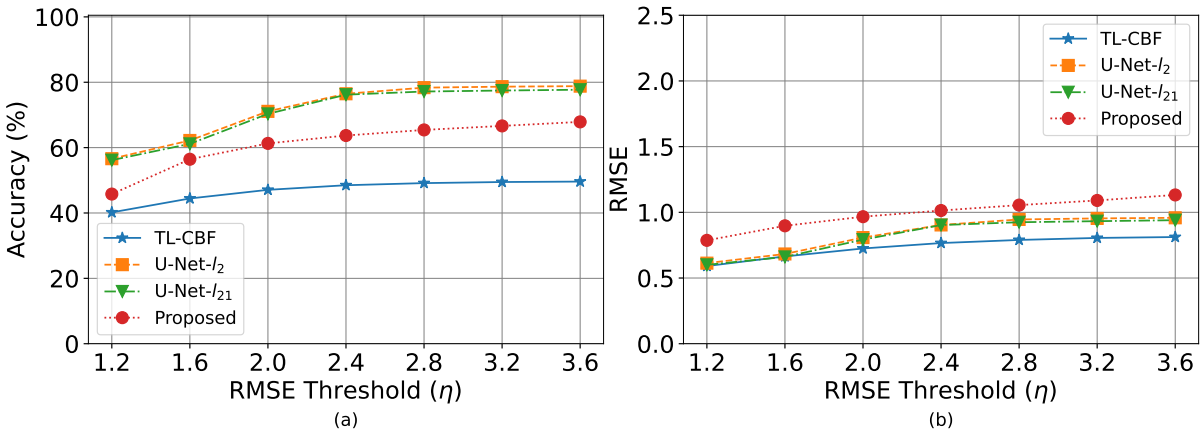

Figure 4.19: [**TestData-3**] Accuracy (a) and average RMSE over detected sources (b) of various algorithms as RMSE threshold varies (η). Source trajectories are fixed to be $(-60.5, 2.5)$, $(-12.5, 3.5)$ and $(5.5, 1.2)$.

In Figure 4.20 (a), the amplitude relative error for the three-source scenario is depicted as SNR varies. Notably, as SNR increases, the amplitude error for each source and its average decreases. It is observed that the proposed method exhibits poor accuracy at lower SNR due to the higher amplitude relative error of sources S1 and S2. This discrepancy leads to error propagation during residual computation, ultimately resulting in an inability to accurately estimate the source trajectory of S3.

Figure 4.21, 4.22, 4.23, 4.24 shows the TL-CBF spectrums for 1, 5, 10 and 15 dB SNR respectively. S1, with a source trajectory of $(-60.5, 2.5)$, is the strongest among all sources and is located far from the other two sources. Following S1, S3 with a source trajectory of $(-5.5, 1.2)$ is stronger than S2, which has a source trajectory of $(-12.5, 3.5)$. These figures illustrate that the proposed method first estimates the source trajectory and amplitudes of S1, followed by S3, and then S2. As sources are removed sequentially, the average power per sensor per snapshot also decreases. Additionally, as SNR increases, the average power per sensor per snapshot (P) decreases, indicating that at higher SNR levels, the method can more effectively remove the sources sequentially.

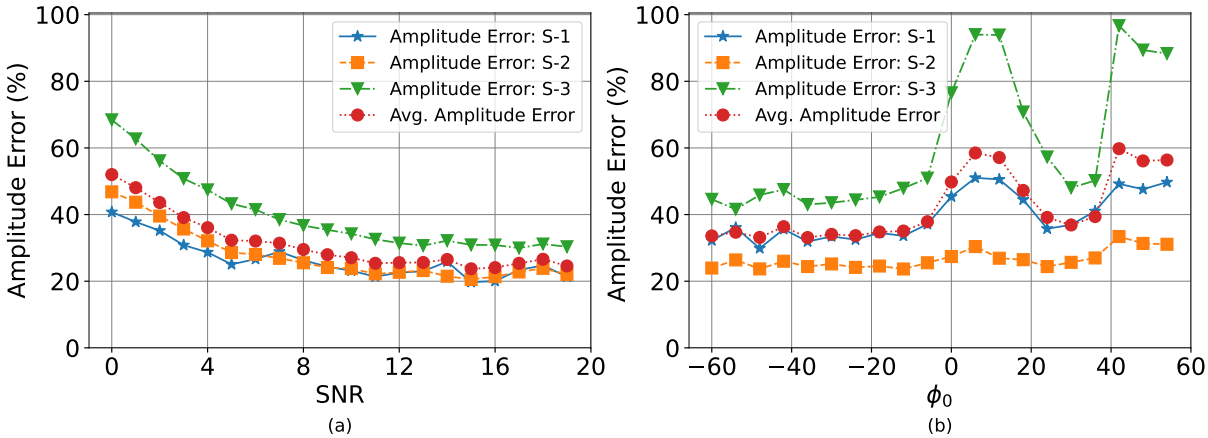

Figure 4.20: Amplitude relative error of the three sources scenarios as SNR (a) and ϕ_0 (b) varies for the proposed method.

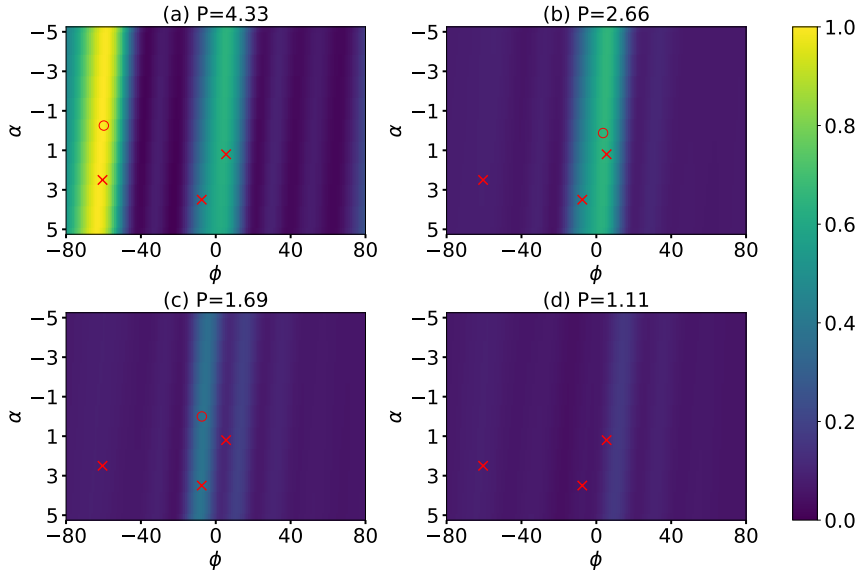

Figure 4.21: [**TestData-3**] TL-CBF spectrums: (a) original signal at 1 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are $(-60.5, 2.5)$, $(-12.5, 3.5)$ and $(5.5, 1.2)$ [red cross]. The estimated source trajectories are indicated by red circle in (a), (b) and (c), which are partially or fully removed in the subsequent spectra (b), (c) and (d) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

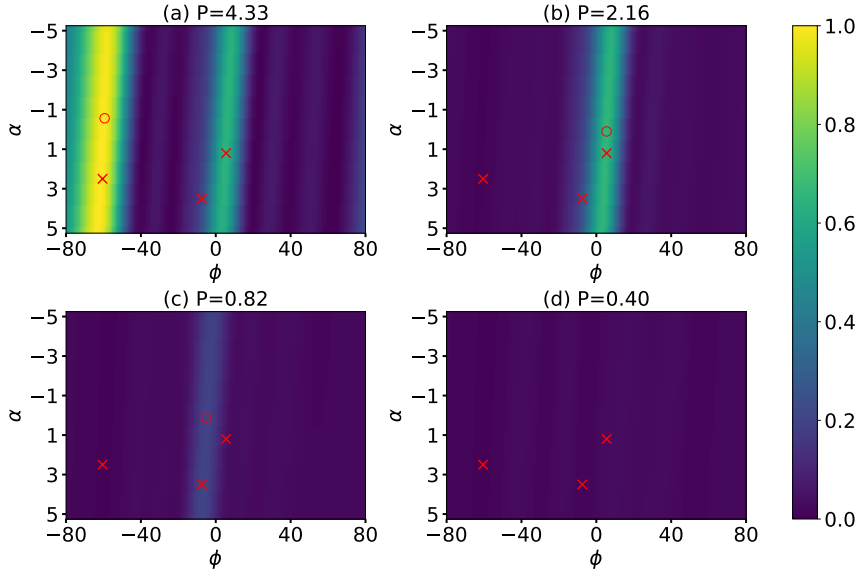

Figure 4.22: [**TestData-3**] TL-CBF spectrums: (a) original signal at 5 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are $(-60.5, 2.5)$, $(-12.5, 3.5)$ and $(5.5, 1.2)$ [red cross]. The estimated source trajectories are indicated by red circle in (a), (b) and (c), which are partially or fully removed in the subsequent spectra (b), (c) and (d) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

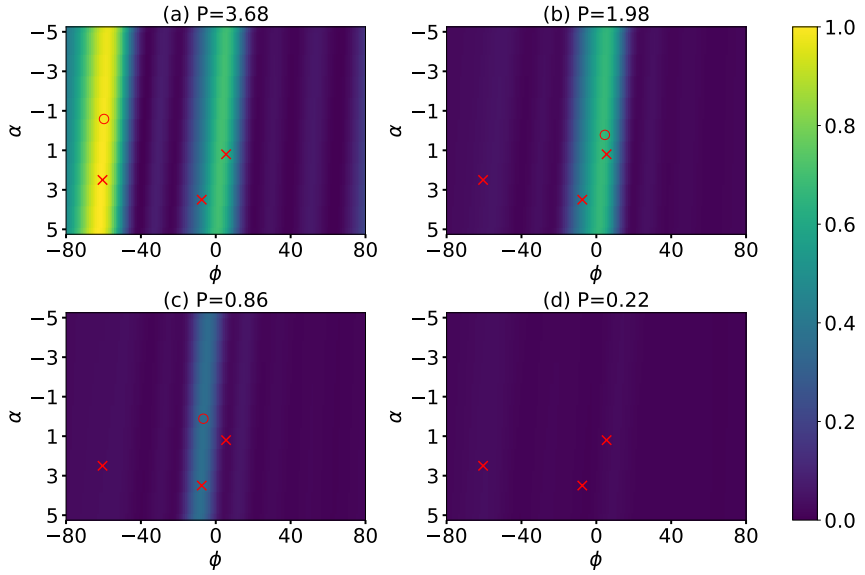

Figure 4.23: [**TestData-3**] TL-CBF spectrums: (a) original signal at 10 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are $(-60.5, 2.5)$, $(-12.5, 3.5)$ and $(5.5, 1.2)$ [red cross]. The estimated source trajectories are indicated by red circle in (a), (b) and (c), which are partially or fully removed in the subsequent spectra (b), (c) and (d) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

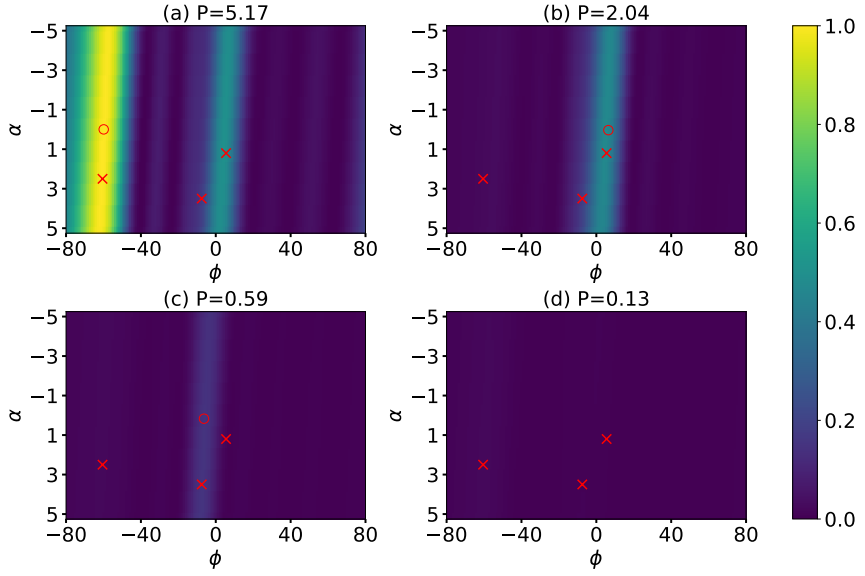

Figure 4.24: [**TestData-3**] TL-CBF spectrums: (a) original signal at 15 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are $(-60.5, 2.5)$, $(-12.5, 3.5)$ and $(5.5, 1.2)$ [red cross]. The estimated source trajectories are indicated by red circle in (a), (b) and (c), which are partially or fully removed in the subsequent spectra (b), (c) and (d) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

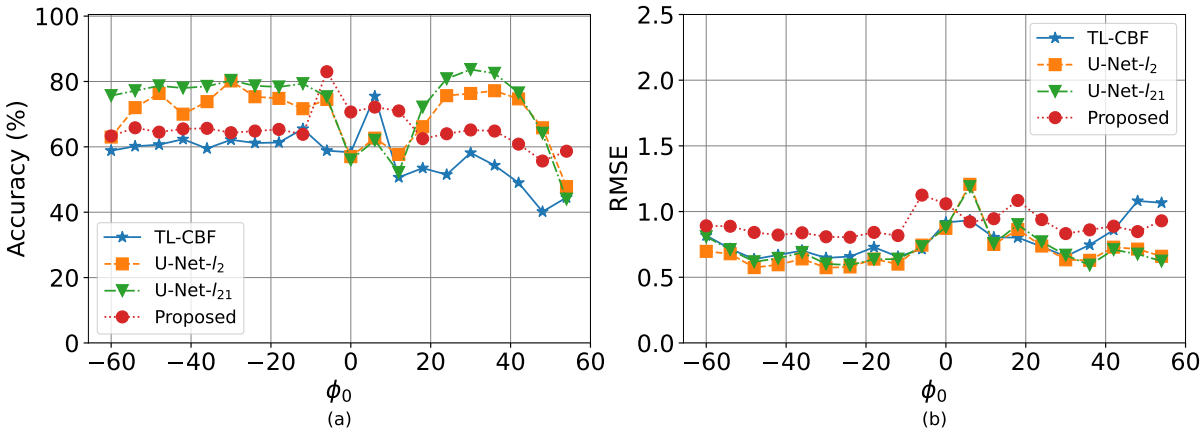

Figure 4.25: [**TestData-4**] Accuracy (a) and average RMSE over detected sources (b) of various algorithms vs ϕ_0. Two sources have fixed trajectories $(60.5, 2.5)$ and $(5.5, 1.2)$, while the third source trajectory is $(\phi_0, -3.8)$. The parameter ϕ_0 varies with a step size of 6, ranging from -60 to 54.

TestData-4: In this experiment, our goal is to assess how different methods perform as one of the source trajectories changes, while the trajectories of the other two sources remain constant. Both the proposed method and U-Net variants have not encountered this three-source scenario during training. Two sources, S1 and S2, have fixed trajectories $(60.5, 2.5)$ and $(5.5, 1.2)$, respectively, with similar energy levels. The source trajectory of S3 varies as $(\phi_0, -3.8)$, where ϕ_0 ranges from -60 to 54. As ϕ_0 increases, initially, S3 moves closer to S2, then moves away from it, and finally approaches near to S1. The proposed methods demonstrate superior performance compared to TL-CBF, yet fall short when compared to the U-Net variants in terms of accuracy. U-Net with l_{21} loss have highest accuracy, while U-Net with l_{21} loss have lowest average RMSE over detected sources. The proposed method have the highest average RMSE over detected sources. In Figure 4.25, the accuracy and average RMSE over detected sources are depicted as a function of ϕ_0. Within the range of $\phi_0 \in [-60, -12] \cup [24, 42]$, where all sources are relatively distant from each other, all algorithms exhibit approximately consistent accuracy with small fluctuations. In this interval, U-Net with l_{21} loss achieves the highest accuracy, while the TL-CBF demonstrates the lowest accuracy. When ϕ_0 falls within the range $[-6, 12]$, S3 is closer to S2. During this interval, the proposed method achieves the highest accuracy, while the accuracy of the grid-based methods exhibits considerable fluctuations. When ϕ_0 is in the range $(42, 54]$, S3 comes closer to S1 and the accuracy of all algorithms decreases. The average RMSE over detected sources is highest for the proposed method as ϕ_0 varies.

In Figure 4.26, the average accuracy and average RMSE over detected sources are plotted against varying values of η. Throughout the range of η, the TL-CBF consistently exhibits the lowest accuracy and while the proposed method exhibits the highest average RMSE over detected sources. The U-Net variants display comparable accuracy and average RMSE across

the range. In Figure 4.20 (b), the amplitude relative error of the three sources is depicted as a function of ϕ_0. When all sources are far apart, the relative error of each source remains small compared to scenarios where S3 approaches either S1 or S2. Notably, the relative error of source S3 consistently remains higher compared to S1 and S2 throughout, attributed to error propagation during residual computation resulting from estimation errors.

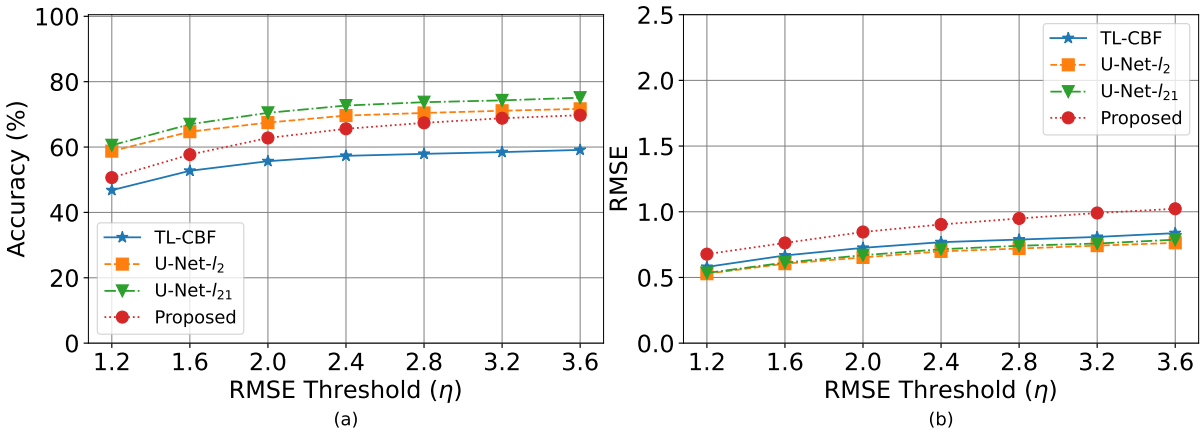

Figure 4.26: [**TestData-4**] Accuracy (a) and average RMSE over detected sources (b) of various algorithms as RMSE threshold varies (η). Two sources have fixed trajectories $(60.5, 2.5)$ and $(5.5, 1.2)$, while the third source trajectory is $(\phi_0, -3.8)$. The parameter ϕ_0 varies with a step size of 6, ranging from -60 to 54.

Figure 4.27, 4.28, 4.29 and 4.30 illustrates the TL-CBF spectrum as sources are removed using the proposed method at 5 dB SNR, corresponding to ϕ_0 values of -12, 0, 48, and 54, respectively.

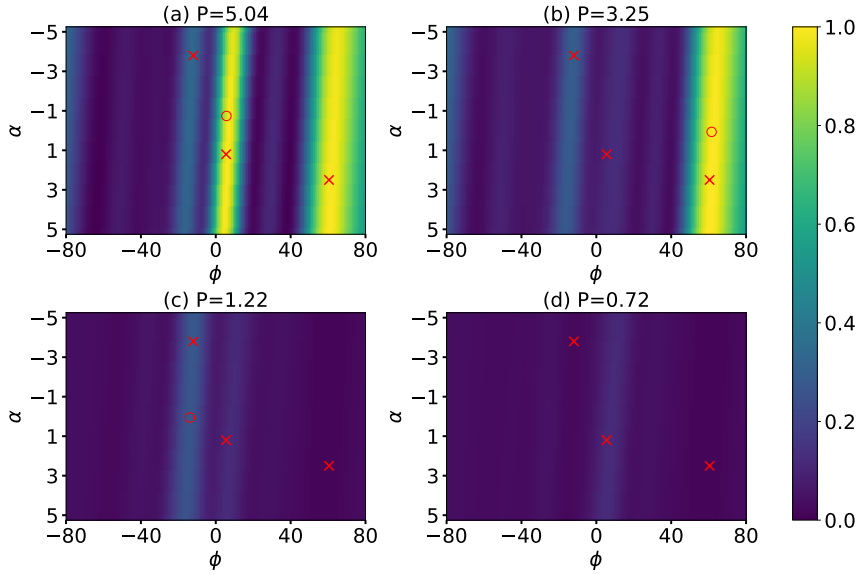

Figure 4.27: [**TestData-4**] TL-CBF spectrums: (a) original signal at 5 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are $(60.5, 2.5)$, $(5.5, 1.2)$ and (ϕ_0, -2.5) [red cross] where $\phi_0 = -12$. The estimated source trajectories are indicated by red circle in (a), (b) and (c), which are partially or fully removed in the subsequent spectra (b), (c) and (d) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

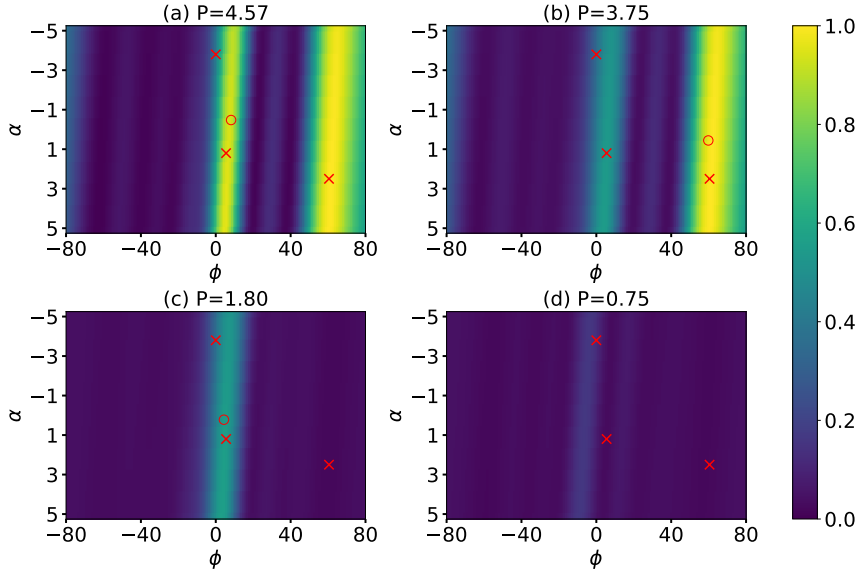

Figure 4.28: [**TestData-4**] TL-CBF spectrums: (a) original signal at 5 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are $(60.5, 2.5)$, $(5.5, 1.2)$ and (ϕ_0, -2.5) [red cross] where $\phi_0 = 0$. The estimated source trajectories are indicated by red circle in (a), (b) and (c), which are partially or fully removed in the subsequent spectra (b), (c) and (d) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

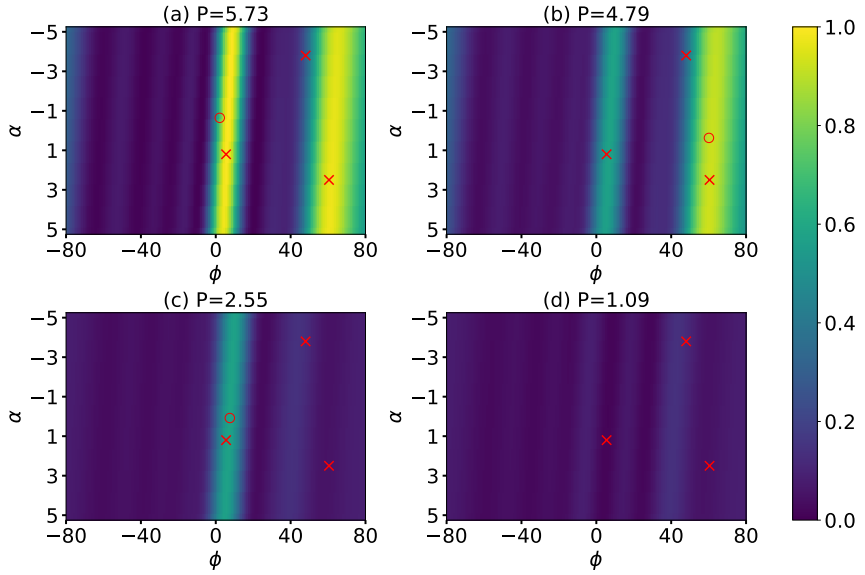

Figure 4.29: [**TestData-4**] TL-CBF spectrums: (a) original signal at 5 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are $(60.5, 2.5)$, $(5.5, 1.2)$ and (ϕ_0, -2.5) [red cross] where $\phi_0 = 48$. The estimated source trajectories are indicated by red circle in (a), (b) and (c), which are partially or fully removed in the subsequent spectra (b), (c) and (d) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

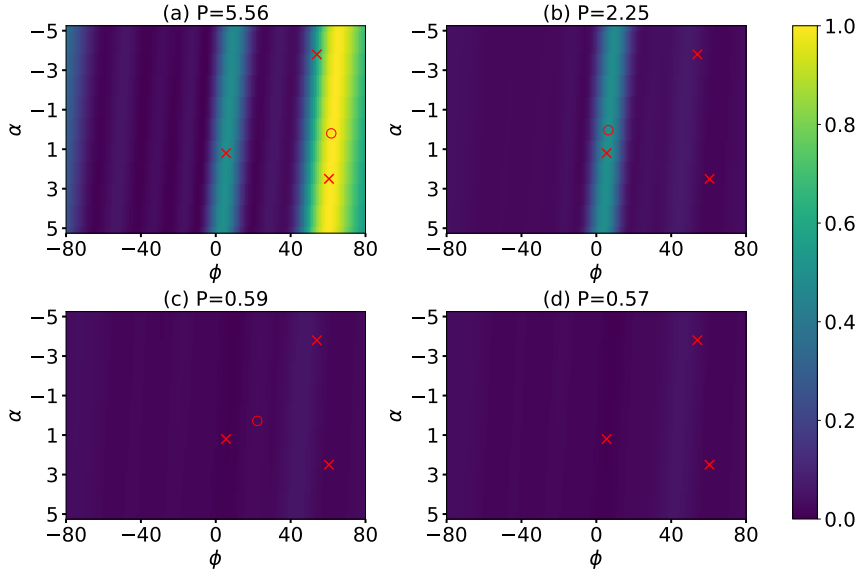

Figure 4.30: [**TestData-4**] TL-CBF spectrums: (a) original signal at 5 dB SNR, (b) after removing first source, (c) after removing second source. Source trajectories are $(60.5, 2.5)$, $(5.5, 1.2)$ and (ϕ_0, -2.5) [red cross] where $\phi_0 = 54$. The estimated source trajectories are indicated by red circle in (a), (b) and (c), which are partially or fully removed in the subsequent spectra (b), (c) and (d) respectively. Additionally, the average power per sensor per snapshot (P) is provided as sources are sequentially removed.

4.4 Summary

In this chapter, we discuss about two primary contributions aimed at addressing the problem of DOA trajectory estimation. Firstly, we introduce a deep complex network designed to estimate source amplitudes and trajectory parameters without relying on a grid-based approach. Secondly, this network is engineered to specifically estimate parameters for the single (strongest) source which makes the network re-useable and allows for sequential source estimation. Simulation results indicate that the proposed method is able to estimate multiple sources and future refinements in the network architecture or training strategies may be explored to mitigate the effects of error propagation and enhance accuracy in such scenarios.

Chapter 5

Conclusions

The thesis begins by addressing a critical issue in existing methods, which often struggle to generalize well to non-ideal behaviors such as uncertain noise characteristics. Additionally, most methods predominantly employ block-level processing assuming static DOA within each block (multiple measurements). This is effective for slow-moving targets, but may not be optimal for scenarios involving fast motion. To address the dynamic behavior of sources within a block, we acknowledge the work by authors in [39] which incorporates the linear DOA motion across snapshots within a block and devise grid-based trajectory localization (TL) algorithms, namely conventional beamforming (TL-CBF) and Sparse Bayesian learning (TL-SBL).

We propose two distinct solutions to address this issue: one employs a *grid-based* approach, while the other adopts a *grid-free* strategy. Firstly, for the *grid-based* solution, we introduce the utilization of the U-Net architecture for DOA trajectory estimation using an RMSE-based target map. This involves developing a neural network specifically designed to enhance low-resolution TL-CBF spectra, transforming them into high-resolution counterparts. The neural network, by harnessing the latent information, aims to enhance peak resolution and, consequently, improve the localization accuracy by allowing precise identification of source trajectories. This showcases remarkable generalization capabilities, effectively adapting to scenarios with differing numbers of sources, even those not encountered during the training phase. Secondly, we develop a deep complex-valued neural network to estimate source amplitudes (per snapshot) and trajectory parameters in a *grid-free* manner which eliminates the need for predefined grids. This network estimates the parameters of the strongest source, making the network reusable and enabling sequential source estimation. The choice of a complex-valued neural network is deliberate, as it offers enhanced capabilities for handling complex data. This approach represents a departure from traditional grid-based methods, offering a more scalable and versatile solution for DOA estimation.

5.1 Future Work

For the *grid-based* solution, further exploration could involve the development of novel architectures integrating attention mechanisms. Additionally, leveraging vision transformers could be investigated to transform low-resolution spectra into high-resolution counterparts, facilitating precise peak identification even in low SNR scenarios. Extending this solution to handle multi-frequency TL-CBF spectrum would significantly broaden the applicability and robustness of the grid-based approach. This expansion would enable the system to adapt to diverse environmental conditions and effectively capture the complexities of real-world scenarios, thereby enhancing its utility in practical applications such as surveillance, communication, and navigation systems. For the *grid-free* approach, efforts can be made to mitigate the effects of error propagation during the residual calculation process. This could involve refining the network architecture or enhancing training strategies to achieve better error handling and minimize inaccuracies. By optimizing these aspects, the grid-free approach has the potential to achieve improved accuracy, potentially surpassing the performance of the grid-based approach. Another avenue for research lies in addressing non-linear trajectories.

Related Publications

1. **Deep Architecture for DOA Trajectory Localization**, Shreyas Jaiswal, Ruchi Pandey, Santosh Nannuru. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2023.

2. **DOA Trajectory Localization using Deep Nets : One Source at a time**, Shreyas Jaiswal, Santosh Nannuru. IEEE European Signal Processing Conference (EUSIPCO), 2024. (Under Review).

Publications Not Included In The Thesis:

3. **Improving audio event localization accuracy via derivative prediction**, Ruchi Pandey, Shreyas Jaiswal, Huy Phan, Santosh Nannuru. IEEE European Signal Processing Conference (EUSIPCO), 2023.

Bibliography

[1] D. Salvati, C. Drioli, G. Ferrin, and G. L. Foresti, "Acoustic source localization from multirotor UAVs," *IEEE Trans. on Indus. Elec.*, vol. 67, no. 10, pp. 8618–8628, 2019.

[2] G. Han, L. Wan, L. Shu, and N. Feng, "Two novel DOA estimation approaches for real-time assistant calibration systems in future vehicle industrial," *IEEE Systems Journal*, vol. 11, no. 3, pp. 1361–1372, 2015.

[3] K. Nakadai, T. Lourens, G.H. Okuno, and H. Kitano, "Active audition for humanoid," in *AAAI/IAAI*, 2000, pp. 832–839.

[4] K. Nakamura, K. Nakadai, F. Asano, and G. Ince., "Intelligent sound source localization and its application to multimodal human tracking," in *IEEE Inter. Conf. on Intel. Rob. and Sys.*, 2011, pp. 143–148.

[5] M. Farmani, M. S. Pedersen, Z. H. Tan, and J. Jensen, "Maximum likelihood approach to "informed" sound source localization for hearing aid applications," in *Inter. Conf. on Acous, Spe. and Sig. Process. (ICASSP)*. IEEE, 2015, pp. 16–20.

[6] B. Alenljung, J. Lindblom, R. Andreasson, and T. Ziemke, "User experience in social human-robot interaction," in *Rapid automation: Concepts, methodologies, tools, and applications*, pp. 1468–1490. IGI Global, 2019.

[7] J-M Valin, François Michaud, Brahim Hadjou, and Jean Rouat, "Localization of simultaneous moving sound sources for mobile robot using a frequency-domain steered beamformer approach," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*. IEEE, 2004, vol. 1, pp. 1033–1038.

[8] Justin Salamon and Juan Pablo Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal processing letters*, vol. 24, no. 3, pp. 279–283, 2017.

[9] Dan Stowell, Mike Wood, Yannis Stylianou, and Hervé Glotin, "Bird detection in audio: a survey and a challenge," in *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2016, pp. 1–6.

[10] Pasquale Foggia, Nicolai Petkov, Alessia Saggese, Nicola Strisciuglio, and Mario Vento, "Audio surveillance of roads: A system for detecting anomalous sounds," *IEEE transactions on intelligent transportation systems*, vol. 17, no. 1, pp. 279–288, 2015.

[11] H. L. Van Trees, *Optimum array processing: Part IV of detection, estimation, and modulation theory*, John Wiley & Sons, 2004.

[12] R. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Trans. Antennas Propag.*, vol. 34, no. 3, pp. 276–280, 1986.

[13] Richard Roy, A Paulraj, and Thomas Kailath, "Estimation of signal parameters via rotational invariance techniques-esprit," in *MILCOM 1986-IEEE Military Communications Conference: Communications-Computers: Teamed for the 90's*. IEEE, 1986, vol. 3, pp. 41–6.

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

[15] V Krishnaveni, T Kesavamurthy, and B Aparna, "Beamforming for direction-of-arrival (doa) estimation-a survey," *International Journal of Computer Applications*, vol. 61, no. 11, 2013.

[16] X. Xiao, S. Zhao, X. Zhong, D.L. Jones, E.S. Chng, and H. Li, "A learning-based approach to direction of arrival estimation in noisy and reverberant environments," in *ICASSP*, 2015, pp. 2814–2818.

[17] Charles Knapp and Glifford Carter, "The generalized correlation method for estimation of time delay," *IEEE transactions on acoustics, speech, and signal processing*, vol. 24, no. 4, pp. 320–327, 1976.

[18] Fabio Vesperini, Paolo Vecchiotti, Emanuele Principi, Stefano Squartini, and Francesco Piazza, "A neural network based algorithm for speaker localization in a multi-room environment," in *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2016, pp. 1–6.

[19] Weipeng He, Petr Motlicek, and Jean-Marc Odobez, "Deep neural networks for multiple speaker detection and localization," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 74–79.

[20] Paolo Vecchiotti, Emanuele Principi, Stefano Squartini, and Francesco Piazza, "Deep neural networks for joint voice activity detection and speaker localization," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 1567–1571.

[21] S. Chakrabarty and E. Habets, "Broadband doa estimation using convolutional neural networks trained with noise signals," in *IEEE WASPAA*, 2017, pp. 136–140.

[22] S. Chakrabarty and E. Habets, "Multi-speaker doa estimation using deep convolutional networks trained with noise signals," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 1, pp. 8–21, 2019.

[23] Scott Rickard and Ozgiir Yilmaz, "On the approximate w-disjoint orthogonality of speech," in *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*. IEEE, 2002, vol. 1, pp. I–529.

[24] P. A. Grumiaux, S. Kitić, L. Girin, and A. Guérin, "A survey of sound source localization with deep learning methods," *J. Acoust. Soc. Am.*, vol. 152, no. 1, pp. 107–151, 2022.

[25] Shlomo E Chazan, Hodaya Hammer, Gershon Hazan, Jacob Goldberger, and Sharon Gannot, "Multi-microphone speaker separation based on deep doa estimation," in *2019 27th European Signal Processing Conference (EUSIPCO)*. IEEE, 2019, pp. 1–5.

[26] O. Ronneberger and T. Fischer, P.and Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[27] Hodaya Hammer, Shlomo E Chazan, Jacob Goldberger, and Sharon Gannot, "Dynamically localizing multiple speakers based on the time-frequency domain," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2021, no. 1, pp. 16, 2021.

[28] B. Laufer-Goldshtein, R. Talmon, and S. Gannot, "Semi-supervised sound source localization based on manifold regularization," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 8, pp. 1393–1407, 2016.

[29] M. J. Bianco, S. Gannot, and P. Gerstoft, "Semi-supervised source localization with deep generative modeling," in *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2020, pp. 1–6.

[30] Diederik P Kingma and Max Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[31] R. Takeda and K. Komatani, "Unsupervised adaptation of deep neural networks for sound source localization using entropy minimization," in *Inter. Conf. on Acous, Spe. and Sig. Process. (ICASSP)*. IEEE, 2017, pp. 2217–2221.

[32] D. Diaz-Guerra, A. Miguel, and J. R. Beltran, "Robust sound source tracking using SRP-PHAT and 3D convolutional neural networks," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 29, pp. 300–311, 2020.

[33] Y. Wu, R. Ayyalasomayajula, M.J. Bianco, D. Bharadia, and P. Gerstoft, "Sslide: Sound source localization for indoors based on deep learning," in *Inter. Conf. on Acous, Spe. and Sig. Process. (ICASSP)*. IEEE, 2021, pp. 4680–4684.

[34] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, "Sound event localization and detection of overlapping sources using convolutional recurrent neural networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 1, pp. 34–48, 2018.

[35] Thi Ngoc Tho Nguyen, Douglas L Jones, Karn N Watcharasupat, Huy Phan, and Woon-Seng Gan, "Salsa-lite: A fast and effective feature for polyphonic sound event localization and detection with microphone arrays," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 716–720.

[36] Georgios K Papageorgiou, Mathini Sellathurai, and Yonina C Eldar, "Deep networks for direction-of-arrival estimation in low snr," *IEEE Transactions on Signal Processing*, vol. 69, pp. 3714–3729, 2021.

[37] Y. Park, F. Meyer, and P. Gerstoft, "Sequential sparse Bayesian learning for time-varying direction of arrival," *J. Acoust. Soc. Am.*, vol. 149, no. 3, pp. 2089–2099, 2021.

[38] R. Opochinsky, G. Chechik, and S. Gannot, "Deep ranking-based DOA tracking algorithm," in *29th European Sig. Process. Conf. (EUSIPCO)*. IEEE, 2021, pp. 1020–1024.

[39] R. Pandey and S. Nannuru, "Parametric models for DOA trajectory localization," in *IEEE Inter. Conf. Acous., Spe., Sig. Proces.* IEEE, 2022, pp. 5118–5122.

[40] Ruchi Pandey and Santosh Nannuru, "Grid-free algorithms for direction-of-arrival trajectory localization," *The Journal of the Acoustical Society of America*, vol. 155, no. 2, pp. 1379–1390, 2024.

[41] C Trabelsi, O Bilaniuk, Y Zhang, D Serdyuk, S Subramanian, JF Santos, S Mehri, N Rostamzadeh, Y Bengio, and CJ Pal, "Deep complex networks," in *Inter. Conf. Learning Representations*, 2017.

[42] S. G Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. on sig. proces.*, vol. 41, no. 12, pp. 3397–3415, 1993.

[43] Y. C. Pati, R. Rezaiifar, and P.S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *IEEE Asilomar conf. on sig., sys. and comput.*, 1993, pp. 40–44.

[44] S. Nannuru, K. L. Gemba, P. Gerstoft, W. S. Hodgkiss, and C. F. Mecklenbräuker, "Sparse Bayesian learning with multiple dictionaries," *Sig. Process.*, vol. 159, pp. 159–170, 2019.

[45] P. Gerstoft, C. F. Mecklenbräuker, A. Xenaki, and S. Nannuru, "Multisnapshot sparse Bayesian learning for DOA," *IEEE Sig. Process. Lett.*, vol. 23, no. 10, pp. 1469–1473, Oct. 2016.

[46] Xinkai Wei, Ioan Andrei Bârsan, Shenlong Wang, Julieta Martinez, and Raquel Urtasun, "Learning to localize through compressed binary maps," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10316–10324.

[47] Robert 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.

[48] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[49] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

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

[51] H.W. Kuhn, "The Hungarian method for the assignment problem," *Naval research logistics quarterly*, pp. 83–97, 1955.

[52] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, Jun. 2001.

[53] Joshua Bassey, Lijun Qian, and Xianfang Li, "A survey of complex-valued neural networks," *arXiv preprint arXiv:2101.12249*, 2021.

[54] Yuheng Zhang, Rong Zeng, Shuchen Zhang, Jiachen Wang, and Yifeng Wu, "Complex-valued neural network with multi-step training for single-snapshot doa estimation," *IEEE Geoscience and Remote Sensing Letters*, 2024.

[55] Nils Mönning and Suresh Manandhar, "Evaluation of complex-valued neural networks on real-valued classification tasks," *arXiv preprint arXiv:1811.12351*, 2018.

[56] Mark Tygert, Joan Bruna, Soumith Chintala, Yann LeCun, Serkan Piantino, and Arthur Szlam, "A mathematical motivation for complex-valued convolutional networks," *Neural computation*, vol. 28, no. 5, pp. 815–825, 2016.

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

[58] Hyeong-Seok Choi, Jang-Hyun Kim, Jaesung Huh, Adrian Kim, Jung-Woo Ha, and Kyogu Lee, "Phase-aware speech enhancement with deep complex u-net," in *Inter. Conf. on Learning Representations*, 2018.

[59] Yanxin Hu, Yun Liu, Shubo Lv, Mengtao Xing, Shimin Zhang, Yihui Fu, Jian Wu, Bihong Zhang, and Lei Xie, "Dccrn: Deep complex convolution recurrent network for phase-aware speech enhancement," *arXiv preprint arXiv:2008.00264*, 2020.

[60] Mhd Modar Halimeh and Walter Kellermann, "Complex-valued spatial autoencoders for multichannel speech enhancement," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 261–265.

[61] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[62] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[63] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[64] Vincent Dumoulin and Francesco Visin, "A guide to convolution arithmetic for deep learning," *arXiv preprint arXiv:1603.07285*, 2016.

[65] Jie Hu, Li Shen, and Gang Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.