# PERFORMANCE OF A NON-LINEAR ADAPTIVE BEAMFORMER ALGORITHM FOR SIGNAL-OF-INTEREST EXTRACTION

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

BY

ÖZKAN OĞUZ

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

FEBRUARY 2015

Approval of the thesis:

**PERFORMANCE OF A NON-LINEAR ADAPTIVE BEAMFORMER ALGORITHM FOR SIGNAL-OF-INTEREST EXTRACTION**

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

Prof. Dr. Gülbin Dural Ünver
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Gönül Turhan Sayan
Head of Department, **Electrical and Electronics Engineering**

Prof. Dr. T. Engin Tuncer
Supervisor, **Electrical and Electronics Engineering Dept., METU**

**Examining Committee Members:**

Prof. Dr. Kemal Leblebicioğlu
Electrical and Electronics Engineering Dept., METU _____

Prof. Dr. T. Engin Tuncer
Electrical and Electronics Engineering Dept., METU _____

Prof. Dr. Buyurman Baykal
Electrical and Electronics Engineering Dept., METU _____

Asst. Prof. Dr. Fatih Kamışlı
Electrical and Electronics Engineering Dept., METU _____

Dr. Özgür Oruç
ASELSAN Inc. _____

Date: 05/02/2015

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

Name, Last Name: Özkan OĞUZ

Signature: _____

# ABSTRACT


## PERFORMANCE OF A NON-LINEAR ADAPTIVE BEAMFORMER ALGORITHM FOR SIGNAL-OF-INTEREST EXTRACTION

Oğuz, Özkan

M.S., Department of Electrical and Electronics Engineering

Supervisor: Prof. Dr. T. Engin Tuncer

February 2015, 96 pages

In this thesis a non-linear adaptive beamforming technique, Adaptive Projections Subgradient Method [1] (APSM) is considered. This method uses projections over convex sets in Reproducing Kernel Hilbert Space. Main advantage of this method is observed if the signal-of-interest is due to digital modulation and when there are more jammers than the number of antennas. The performance of this non-linear beamforming technique is compared with well-known methods including Minimum Variance Distortionless Response [2] (MVDR) Beamformer, Robust Capon Beamformer [3] (RCB), Covariance Matrix Reconstruction [4] (CMR) Beamformer and Recursive Least Squares [5] (RLS) Capon Beamformer. A new beamformer is proposed in order to fill the gap between the non-linear and classical beamformer methods. It is shown that this new beamformer performs better than well-known methods when the number of jammers are larger than the number of antennas like APSM. However, the computational complexity of this method is significantly lower than APSM. The beamformer methods are compared in a variety of scenario in order to outline the advantages and disadvantages clearly.

Keywords: Adaptive beamforming, linear beamforming, non-linear beamforming, Reproducing Kernel Hilbert Space, sensor array signal processing.

# ÖZ

## HEDEF SİNYAL ÇIKARIMI İÇİN DOĞRUSAL OLMAYAN UYARLAMALI BİR HÜZME ŞEKİLLENDİRME ALGORİTMASININ PERFORMANSI

Oğuz, Özkan
Yüksek Lisans, Elektrik Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. T. Engin Tuncer

Şubat 2015, 96 sayfa

Bu yüksek lisans tezinde, Uyarlamalı İzdüşüm Alt Eğim Metodu [1] (APSM) adlı doğrusal olmayan uyarlamalı bir hüzme şekillendirme tekniği değerlendirilmiştir. Bu yöntemde Üretken Çekirdek Hilbert Uzayındaki dışbükey kümelere izdüşümler kullanılır. Bu metodun temel avantajı hedef sinyal sayısal kiplenmiş ve anten sayısı karıştırıcı sinyal sayısından daha az ise gözlemlenmektedir. Bu doğrusal olmayan hüzme şekillendirme tekniğinin başarımı, en küçük değişinti bozulmasız cevap hüzme [2] (MVDR) şekillendirici, gürbüz Capon hüzme şekillendirici [3] (RCB), ortak değişken matris yeniden yapılandırma [4] (CMR) hüzme şekillendirici, yinelemeli en düşük kare [5] (RLS) Capon hüzme şekillendirici gibi bilinen hüzme şekillendirici algoritmaların başarımları ile karşılaştırılmıştır. Doğrusal olmayan ve klasik hüzme şekillendirici metotları arasındaki boşluğu doldurmak amacıyla yeni bir hüzme şekillendirici önerilmiştir. Bu yeni hüzme şekillendiricinin, APSM metodunda olduğu gibi, karıştırıcı sinyal sayısının anten sayısından yüksek olduğu durumda, bilinen metotlardan daha iyi işlediği gösterildi. Ancak, bu metodun işlem karmaşıklığı APSM' nin hatırı sayılır biçimde altındadır. Hüzme şekillendirici

metotları avantaj ve dezavantajlarının açık biçimde belirtilmesi için farklı senaryolarda karşılaştırılmıştır.

Anahtar kelimeler: Uyarlamalı hüzme şekillendirme, doğrusal hüzme şekillendirme, doğrusal olmayan hüzme şekillendirme, Üretken Çekirdek Hilbert Uzayı, alıcı dizini sinyal işleme.

***To my grandmother***

# ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor **Prof. Temel Engin TUNCER** for his progressive guidance, advice and supervision throughout my study. I owe special thanks to my friend and colleague **Dr. Özgür ORUÇ** for his immense and crucial support in a very educative, patient and supportive manner with his telecommunications and teaching background. I would like to thank **Dr. Serkan KARAKÜTÜK** and **Dr. Handan AĞIRMAN** for their help during my studies.

Finally yet importantly, I would like to express that I am grateful to my parents, **Hatice OĞUZ** and **Tahir OĞUZ** for their life-long affection and support.

# TABLE OF CONTENTS

# LIST OF FIGURES

FIGURES

# LIST OF TABLES

TABLES

# LIST OF ABBREVATIONS

AM: Amplitude Modulated

APSM: Adaptive Projections Sub-gradient Method

ASK: Amplitude Shift Keying

AWCM: Adaptive Weighted Covariance Matrix

BPSK: Binary Phase Shift Keying

CMR: Covariance Matrix Reconstruction

DOA: Direction of Arrival

FM: Frequency Modulated

MI: Matrix Inversion

MVDR: Minimum Variance Distortionless Response

PSK: Phase Shift Keying

QPSK: Quadrature Phase Shift Keying

RCB: Robust Capon Beamformer

SER: Symbol Error Rate

SINR: Signal to Interference plus Noise Ratio

SNR: Signal to Noise Ratio

SOI: Signal of Interest

UCA: Uniform Circular Array

ULA: Uniform Linear Array

# CHAPTER 1

# INTRODUCTION

In this chapter, motivation of the academic study, which lead to this thesis, has been stated and organization of thesis is given. Then brief theoretic information on beamforming and estimation use of beamforming that is used in the thesis have been given in the following sections on the chapter.

## 1.1. Motivation of Thesis

Sensor arrays have a history over a century of practical usage in practical world. Sensor Array Signal Processing applications are used in RADARs, SONARs, Telecommunications, Medical Applications, Speech Processing and Space Sensor Applications. Objectives of Array Signal Processing may vary as:

- Detection of a particular Signal of Interest (SOI) in a noisy signal environment,
- De-modulating/classifying impinged signals for the original transmitted signal,
- Estimating the Direction of Arrival (DOA) of SOI in a noisy environment,
- Directing the beam of transmitted signal to a specific location/direction,
- Finding the location of the SOI.

In [1] a non-linear beamforming algorithm (Adaptive Projections Sub-gradient Method) has been explained theoretically and its performance has been briefly presented in a single beamforming scenario. Five signal sources transmit binary symbols to an array of three antennas in uniform linear array geometry, where the array is tasked to estimate the SOI. In contrast to classical beamforming methods,

where the number of signals exceeds the number of antennas, its performance has been reported to be good. Announcement of such a good performance has been a motivation for building this master thesis.

The objective of this thesis is to understand working mechanism and performance of Adaptive Projections Sub-gradient Method. In this thesis, in order to understand how this algorithm (APSM) behaves in different beamforming scenarios, it has been implemented along with other well-known beamformers. Those other beamformers are Minimum Variance Distortionless Response (MVDR) beamformer, Robust Capon Beamformer (RCB), Recursive Least Squares (RLS) Beamformer and Covariance Matrix Reconstruction (CMR) beamformer. All abovementioned beamformers have been implemented as signal of interest estimators and their performances have been compared using input schemes of Analog Modulated (AM), Binary Phase Shift Keying (BPSK) and Quadrature Phase Shift Keying (QPSK). Array geometry is taken as uniform linear array for default case. However, additional comparisons have been made for uniform circular array geometry. Performance criteria have been Signal-to-Interference-plus-Noise Ratio (SINR) for real number inputs and Symbol Error Rate (SER) for integer input cases. Comparisons have been made with sweeping input SNR, DOA mismatch, number of interferences, number of array antennas, interference-to-SOI signal angle, number of snapshots, SOI-to-Jammer Power Ratio and carrier frequency. Additional comparisons have been made on output power, Beampattern and processing time. Interesting findings on SOI a priori DOA error have been made and presented. Having implemented and compared all the algorithms mentioned above, an additional method, Adaptive Weighted Covariance Matrix Method (AWCM), has been proposed and its performance has been compared with former methods.

### 1.2. Organization of Thesis

The organization of this thesis is as follows: in chapter 1, motivation of thesis is stated with some background information on beamforming, which is used in the rest

of the text. In chapter 2, a number of well-known beamformers have been introduced. In chapter 3, APSM algorithm has been explained in detail. In chapter 4, performance criteria for performance evaluation of beamformers have explained and performance comparison results have been presented in relative sections. In chapter 5, thesis work has been summarized and conclusions on compared beamformers have been made.

Throughout this text, matrices will be represented in **bold** and UPPERCASE letters whereas vectors will be represented in **bold** and lowercase letters. **I** will denote the identity matrix of appropriate size.

### 1.3. Array Signal Model

A number of sensors placed in space are required for Array Signal Processing in order to receive or transmit signal(s) of Interest with a space-filter generated by the array. Depending on the geometry and placement of the sensors that constitute an array, arrays are classified as [6]:

1. Array Geometry
    i. Linear
    ii. Planar
    iii. 3-D

2. Sensor Placement
    i. Uniform
    ii. Non-Uniform (Based on careful design on particular scenario)
    iii. Randomly Spaced (No rule of placement)

In this thesis, spherical coordinate system shown in Figure 1 has been used, where $\theta$ denotes azimuth angle and $90°-\varphi$ denotes elevation angle for sources.

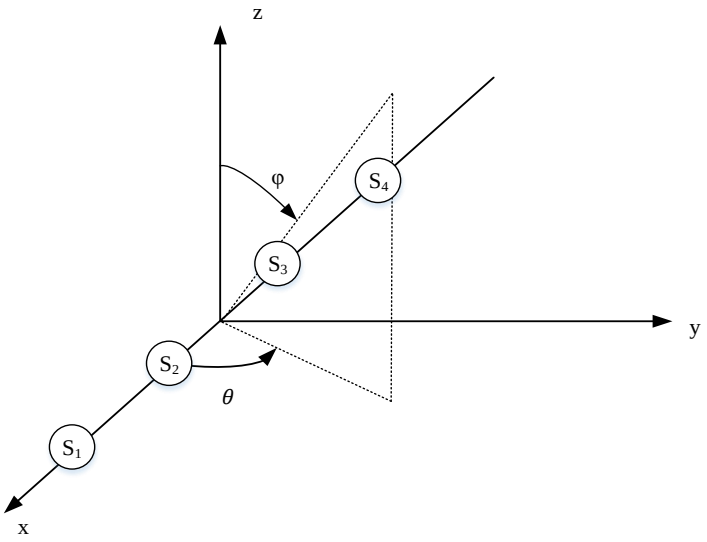


Figure 1. Spherical Coordinates

2-D Array geometries for uniform linear array (ULA) and uniform circular array (UCA), which have been used for this thesis work, are shown in Figure 2 and Figure 3 with sensors denoted as $S_i$, where i denotes the number of the sensor in the array. All the arrays and signal sources used in this thesis have been placed on φ=90° plane.

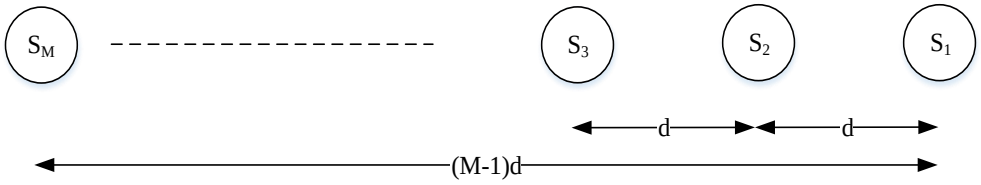


Figure 2. Uniform Linear Array of M elements

In this thesis, uniform linear arrays have been placed on x-axis of coordinate system as shown in Figure 1.

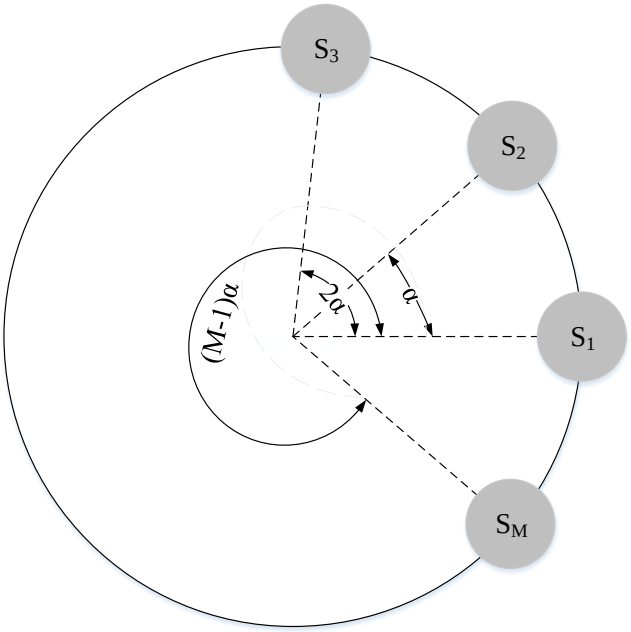


Figure 3. Uniform Circular Array

ULAs and UCAs are very commonly used in beamforming applications. Their performance has been comparatively evaluated in [7]. For ULAs, θ=0° and θ=180° directions are called array endfire, where steered main lobe is wide and array spatial resolution is low. θ=90° and θ=270° directions are called to array boresight and main lobes steered to these directions are narrow. Therefore, array spatial resolution is high for these DOA values. In UCAs, there is no concept of endfire or boresight because array is geometrically symmetrical over the whole azimuth.

Array aperture is the maximum length occupied by any two elements of array and has a particular effect on array performance that will be observed in the following chapters. As a rule, inter-elemental spacing between two consecutive array elements cannot be over 0.5λ not to cause spatial aliasing. The idea behind this rule will be covered in this section and this rule is known as “Spatial Nyquist Principle”.

In a scenario that a single source signal to impinge on an array of M elements, the signals received on each array element are represented as **x**(t) where **x**(t) is a 1xM complex valued vector. **x**(t) would be expressed as:

$$\mathbf{x}(t)=\mathbf{a}(\theta)s(t)+\mathbf{n}(t) \text{ for } t=1,\ldots,N \quad (1)$$

where **a**(θ) is the array steering vector of size Mx1, s(t) is signal value of source for a specified time instant of "t" which is a 1x1 value and **n**(t) is the environment and sensor measurement noise vector of size 1xM with N number of snapshots. Steering vector **a**(θ) can be expressed as:

$$\mathbf{a}(\theta)=\begin{bmatrix} e^{-jwT_1(\theta)} \\ e^{-jwT_2(\theta)} \\ \vdots \\ e^{-jwT_M(\theta)} \end{bmatrix} \quad (2)$$

where w is the angular carrier frequency and $T_i(\theta)$ is the time lag for the signal to travel from a reference point to $i^{th}$ sensor. As one would easily understand, the steering vector **a**(θ) is nothing but complex-phase values for each sensor at a particular angle of arrival. The illustration of the scenario and phase values are seen in Figure 4.

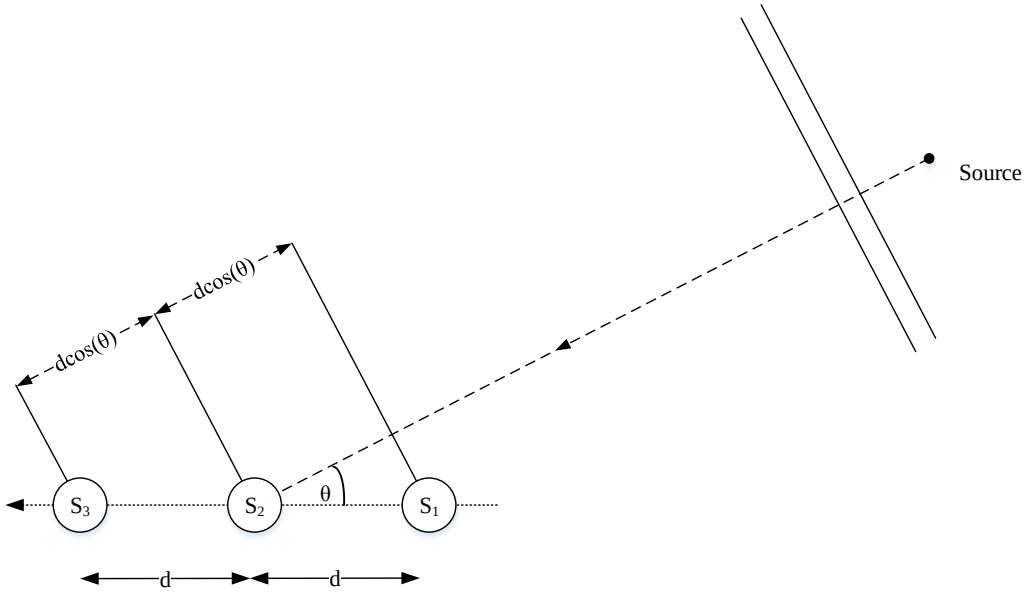


Figure 4. A Single Source Signal impinging on ULA

Taking $S_1$ as the reference point, the phase value of $S_1$ is "1", the time lag for the signal to reach $S_2$ and $S_3$ are:

$$T_2(\theta)= \frac{d\cos(\theta)}{c} \text{ , } T_3(\theta)= \frac{2d\cos(\theta)}{c} \tag{3}$$

where "c" denotes the speed of light. Taking into account that $w=2\pi f$ and $\frac{f}{c} = \frac{1}{\lambda}$, steering vector becomes:

$$\mathbf{a(\theta)} = \begin{bmatrix} 1 \\ e^{-j\frac{2\pi d\cos(\theta)}{\lambda}} \\ \vdots \\ e^{-j\frac{(M-1)\pi d\cos(\theta)}{\lambda}} \end{bmatrix} \tag{4}$$

The idea behind the above-mentioned "Spatial Nyquist Principle" is revealed here: for the phase value to be unambiguous between two consecutive sensor signals the exponent $-\frac{2\pi d\cos(\theta)}{\lambda}$ shall take values between $[-\pi, \pi]$. This is mathematically

guaranteed only in the case that $d<\frac{\lambda}{2}$ . Expanding one-signal scenario to a multi-signal scenario, compound signal impinging from S signals (one signal of interest and S-1 number of jammers) on a uniform linear array of M elements (with minimum inter-elemental spacing = d) be called again **x**(t) where **x**(t) is a 1xM complex valued vector. **x**(t) here would be expressed as:

$$\mathbf{x}(t)=\mathbf{A}(\boldsymbol{\theta})\mathbf{s}(t)+\mathbf{n}(t) \tag{5}$$

where **A**(**θ**) is the array steering matrix of size MxS for signal of interest and jammers. **s**(t) is the vector for signal values of signal of interest and jammers for a specified time instant of "t" which is a Sx1 vector and **n**(t) is the vector for environment and sensor measurement noise of length M. Steering matrices **A**(**θ**) can be expressed as:

$$\mathbf{A}(\boldsymbol{\theta}) =[\mathbf{a}(\theta_1) \quad \mathbf{a}(\theta_2) \quad \dots \quad \mathbf{a}(\theta_S)] \tag{6}$$

where $\theta_i$ is the angle of arrival for $i^{th}$ signal on the array. The columns of steering matrix belong to each individual signal and steering vectors for each signal whose closed form expression is given in Eq. (4). As mentioned above, for most cases in Uniform Linear Arrays, the reference point is taken as first sensor of the array, thus first row elements of the steering matrix are all 1, which is called Vandermonde Matrix. Whereas for Uniform Circular Arrays the reference point is taken as array center (without any sensor on it). Therefore, steering matrix rows that correspond to sensors, which are symmetrical along the reference point, are complex conjugate of each other. Eq. (5) can be expressed in summation form as:

$$\mathbf{x}(t)=\mathbf{a}(\theta_1)s_1(t)+\sum_{i=2}^{S}\mathbf{a}(\theta_i)s_i(t)+\mathbf{n}(t) \tag{7}$$

where **e**(t) is called as interference part and can be expressed as:

$$\mathbf{e}(t)=\sum_{i=2}^{S}\mathbf{a}(\theta_i)s_i(t) \tag{8}$$

### 1.4. Linear Beamforming vs. Non-Linear Beamforming

In order to transmit to or receive from a specific direction of interest, different beamforming techniques are used in literature with different complex weight factors are applied multiplicatively to the elements of an array signals and resultant signals are added to produce a single output at a given time, this beamforming scheme is known as linear beamforming [8]. A simple linear beamforming application is illustrated Figure 5.

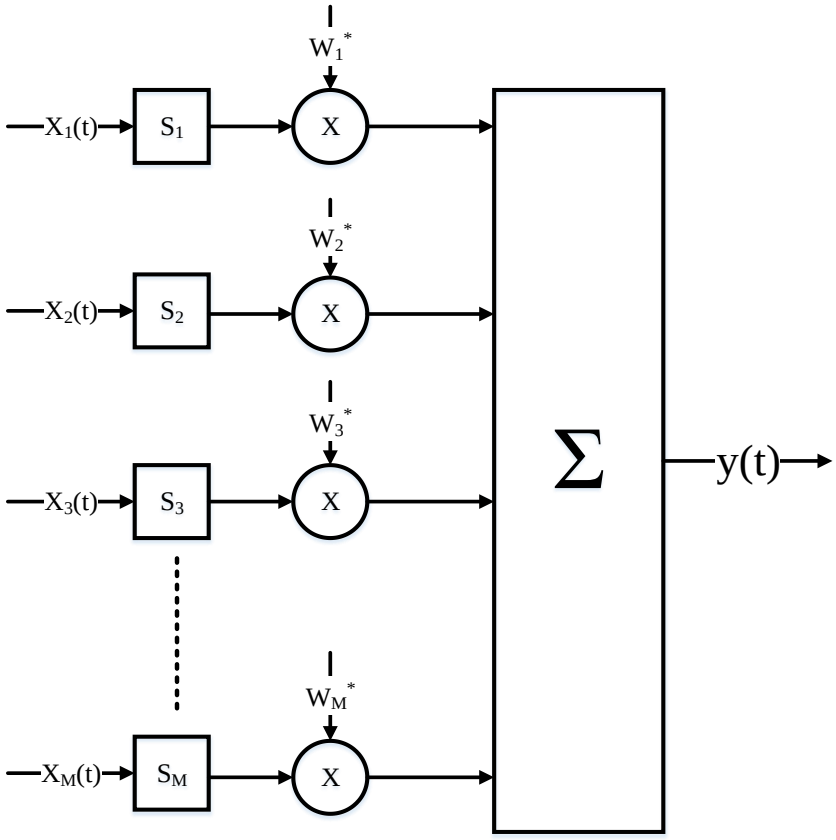


Figure 5. Linear Beamforming Example

Representing individual weight factors in a vector form $\mathbf{w}(t) = [w_1(t)\ w_2(t)\ \ldots\ w_M(t)]^T$, the beamformer output y(t) can be expressed as:

$$y(t)=\mathbf{w}^H(t)\mathbf{x}(t) \tag{9}$$

That is

$$y(t)=\mathbf{w}^H(t)[\mathbf{a}(\theta_1)s_1(t)+\mathbf{e}(t)+\mathbf{n}(t)] \tag{10}$$

which can be partitioned as:

$$y(t)=\mathbf{w}^H(t)\mathbf{a}(\theta_1)s_1(t)+\mathbf{w}^H(t)\mathbf{e}(t)+\mathbf{w}^H(t)\mathbf{n}(t) \tag{11}$$

where the first part of the summation is the signals of interest's contribution to the beamformer output and second part is that of interferences and the third part belongs to the noise.

In linear beamforming applications weight factor "**w**" is found in such a way to achieve certain goals for problem at hand such as maximizing signal-to-noise plus interference ratio at the output or having minimum noise-plus-interference at the output.

In non-linear beamforming, beamformer output is not expressed directly as a linear function [1] such that $y(t)=\mathbf{w}^H\mathbf{x}(t)$ as in the case of linear beamforming. In non-linear beamforming applications seen in literature [1] [9] [10] [11], the beamformer output is expressed as:

$$y(t_N)=\sum_{i=1}^{N-1}\gamma_i\langle\mathbf{k}(\mathbf{x}(t_i)),\mathbf{k}(\mathbf{x}(t_N))\rangle \tag{12}$$

where $\mathbf{k}(\mathbf{x}(t_i))$ and $\gamma_i$ respectively denote special functions called kernel and kernel coefficients computed based on beamformer input signals $x(t_i)$ for a particular instant $t_i$. Eq. (12) means that beamformer output is summation of weighted inner products of kernel functions of beamformer inputs. More information on generation and applications of kernels has been given in chapter 3.

.

Some assumptions have been made in this work for simplification of beamforming process such as [12]:

1. Far field assumption: The radiating sources are assumed to be in the far field of the arrays receiving them. Under far field assumption, the propagating signals are planar waves.

2. Narrowband assumption: The signals are assumed narrowband. Therefore, time-bandwidth product is small ($f_{BW}\tau \ll 1$) for the signal to travel through array. $f_{BW}$ denotes signal frequency bandwidth and $\tau$ denotes maximum time for the signal travel through the whole array.

3. Homogeneity of Transmission Media assumption: Transmission medium is assumed homogenous and non-dispersive so that the signal does not change as it propagates.

4. Independent - Identical - Normal Distribution of Noise assumption: The noise implemented in the simulations are assumed zero mean complex-valued additive white Gaussian noise.

5. Zero Elevation assumption: For computational simplicity, the elevation of all signal sources are assumed zero with respect to arrays' plane, that is $\varphi=90°$. Therefore, arrays are co-planar with signal sources.

6. Uncorrelatedness of the Signals: The signals and noise are assumed uncorrelated in the building of this work.

7. Multipath: The signals are assumed to be directly impinging on the arrays; that is, no multipath effect is induced in the simulations.

### 1.5. Beamformers for SOI Estimation

Beamformer applications in communications may be used to determine the signal of interest's value. Communication signals may be analog FM, AM or digital modulations such as BPSK, QPSK, QAM, etc. In this master thesis, different robust adaptive beamforming algorithms have been implemented and their performances in estimating SOI have been compared using several input types. Input signal generation and beamformer output generation for different I/O schemes has been made as follows:

1. BPSK case: Symbols for each signal are randomly selected from [-1, +1] with equal probability. Real part of beamformer output have been designated as final beamformer output and beamformer outputs have been estimated as ±1 according to their distance to the input signal constellation. Sensor and process noise are additive and complex valued.

2. QPSK case: Symbols for each signal are randomly selected from [-1, +1, -i, +i] with equal probability. Complex valued beamformer outputs have been estimated as [-1, +1, -i, +i] according to their distance to input signal constellation. Sensor and process noise are additive and complex valued.

3. AM case: Signal are randomly generated real numbers with zero mean and variance of "1". Beamformer outputs are used as is for estimation. Sensor and process noise are additive and complex valued.

Illustration of beamformer output generation for BPSK and QPSK are seen in Figure 6.

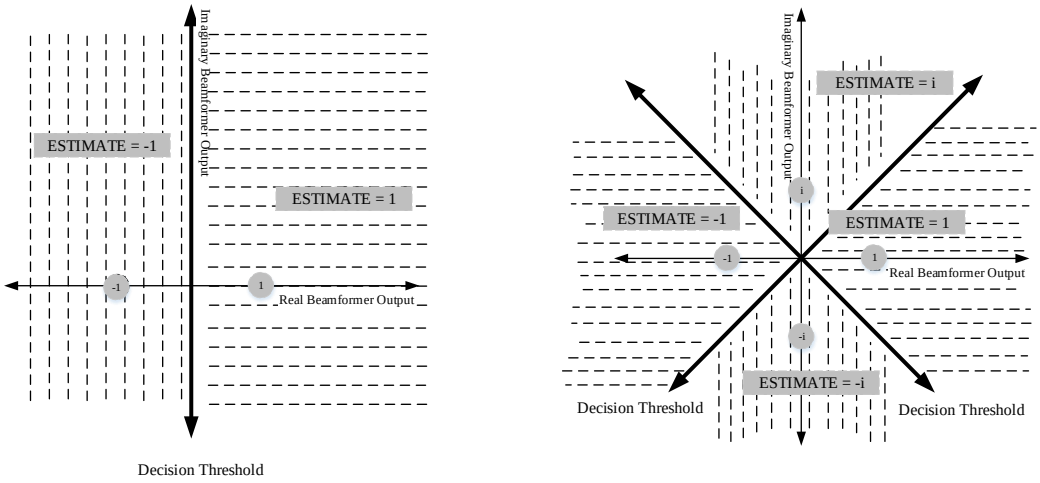


Figure 6. Beamformer Output Generation Mechanisms for BPSK and QPSK I/O Schemes

It shall be noted that due to a threshold mechanism used in the algorithms for BPSK and QPSK schemes, operating error levels are quite small.

# CHAPTER 2

# CLASSICAL BEAMFORMERS

In this chapter, several linear beamformers have been mentioned with their background information, design goals, properties and algorithm design steps. The organization of this chapter is as follows: first beamformer mentioned is Minimum Variance Distortionless Response beamformer with the intention of maximizing SINR at the output. However, in practical beamforming applications, the signal-of-interest is frequently received from a direction with some level of error. Robust Capon Beamformer has been presented as the second beamformer of this chapter with its advantage of maximizing SINR at the output in case of SOI DOA errors. Another imperfection encountered in beamforming is self-nulling which results in suppression of SOI. Third algorithm of the chapter is Covariance Matrix Reconstruction, which can deal with self-nulling while proving powerful against DOA mismatches. Finally, Recursive Least Squares algorithm has been presented as a beamformer to use its adaptive nature in beamforming scenarios.

## 2.1. Minimum Variance Distortionless Response

Minimum Variance Distortionless Response Beamformer (MVDR), sometimes referred to as Capon Beamformer [2], has been proposed by Capon in 1969 and been widely used in beamforming applications. The idea behind the algorithm is to maximize Signal-to-Interference-plus-Noise Ratio (SINR) by minimizing interference-plus-noise contribution to the beamformer output while keeping the beamformer gain as unity for the direction assumed to be that of Signal of Interest. That is:

$$\max_{\mathbf{w}} \text{SINR} = \max_{\mathbf{w}} \frac{\mathbf{w}^H \mathbf{R}_s \mathbf{w}}{\mathbf{w}^H \mathbf{R}_{e+n} \mathbf{w}} \tag{13}$$

where $\mathbf{R}_s$ is the covariance matrix of signal of interest and $\mathbf{R}_{e+n}$ is the covariance matrix of interference plus noise. For one signal of interest, the mathematical driving expression becomes:

$$\mathbf{w}_{MVDR} = \arg \min_{\mathbf{w}} \mathbf{w}^H \mathbf{R}_{e+n} \mathbf{w} \text{ such that } \mathbf{w}^H \mathbf{a}(\theta_{Ap}) = 1 \tag{14}$$

where $\theta_{Ap}$ is the direction of arrival angle for signal of interest while $\mathbf{R}_s$ and $\mathbf{R}_{e+n}$ are given by:

$$\mathbf{R}_s = E\left\{\left[\mathbf{a}(\theta_{Ap})s(t)\right]\left[\mathbf{a}(\theta_{Ap})s(t)\right]^H\right\} \tag{15}$$

$$\mathbf{R}_{e+n} = E\{[\mathbf{e}(t)+\mathbf{n}(t)][\mathbf{e}(t)+\mathbf{n}(t)]^H\} \tag{16}$$

Considering the independence and orthogonality of noise and interferences with each other $\mathbf{R}_{e+n}$ becomes:

$$\mathbf{R}_{e+n} = \mathbf{R}_e + \mathbf{R}_n \tag{17}$$

$$\mathbf{R}_{e+n} = \mathbf{R}_e + \mathbf{I}\sigma_n^2 \tag{18}$$

where $\sigma_n^2$ is the variance of the noise.

Structure of $\mathbf{R}_{e+n}$ will be discussed and exploited in the following sections in order to modify training data and covariance matrix for our particular goal of robust beamforming. A careful look on Eq. (20) reveals that $\mathbf{w}^H \mathbf{R}_{e+n} \mathbf{w}$ is nothing but covariance matrix for interference plus noise at the beamformer output. Hence, the minimization process guarantees that noise plus interference, which constitute $\mathbf{R}_{e+n}$, seen at the output is minimum. Minimization constraint ensures that signal coming

from expected angle of arrival is undistorted. Beamformer weight vector through solution of Eq. (14) by method of Lagrange multiplier yields [13]:

$$\mathbf{w}_{MVDR} = \frac{\mathbf{R}_{e+n}^{-1}\mathbf{a}(\theta_{Ap})}{\mathbf{a}^{H}(\theta_{Ap})\,\mathbf{R}_{e+n}^{-1}\,\mathbf{a}(\theta_{Ap})} \tag{19}$$

MVDR method is statistically optimum as it uses true covariance matrix of the noise and interferences. However for practical cases the exact noise plus interference covariance matrix is not known perfectly and must be estimated through a special "training" period. During training period, noise and interference signals are impinged on the array and samples are taken to generate sample covariance matrix as follows:

$$\widehat{\mathbf{R}}_{e+n} = \frac{1}{N}\sum_{t=1}^{N}[\mathbf{e}(t)+\mathbf{n}(t)][\mathbf{e}(t)+\mathbf{n}(t)]^{H} \tag{20}$$

Beamformers designed using Eq. (19) are known are *Matrix Inverse* or *Matrix Inversion* algorithms. Several MI beamformers exist with some differences in between. However, MI algorithms that use sample covariance matrix have a major shortcoming: "Self-Nulling", that is SOI signal components or snapshots are included in $\widehat{\mathbf{R}}_{e+n}$ and beamformer weight vector is designed in a way that SOI is treated as interference. In this thesis, MVDR is implemented using true Noise-plus-Interference covariance matrix whereas other methods use Sample covariance matrices generated using training period samples. Using an exact covariance matrix of interference signals makes MVDR method the best of all MI algorithms. However, as imperfections arise such as a priori DOA mismatch, having antenna number less than total number of signals, MVDR performance depreciates. Some modifications to Eq. (19) adds nature of robustness and adaptiveness to beamforming application at hand. In the following sections some of those algorithms, which have been implemented in this thesis work, will be presented.

## 2.2. Robust Capon Beamformer

A priori DOA knowledge is most of the time is not correct due to inaccurate angle knowledge or slow motion of platforms in the scenarios. Angular mismatch of a few degrees is very common in beamforming applications, that is, the assumed DOA and true DOA of SOI are different. In order to deal with imperfections like angular mismatch or self –nulling, rigorous studies have been made in the literature, most common of those algorithms are based on covariance matrix modifications of some sense. A well-known and widely used technique is diagonal loading. Examining Eq. (18), intuition leads that interference components are distributed to covariance matrix elements according to the angle they impinge and white noise component is seen directly at the diagonal elements of the covariance matrix. So using diagonal loading alters the covariance matrix to be suppressed in the beamforming application thus modifies the beamformer. The modification is as follows:

$$\widetilde{\mathbf{R}}_{e+n} = \widehat{\mathbf{R}}_{e+n} + \mathbf{\Lambda} \tag{21}$$

where $\mathbf{\Lambda}$ is a real valued diagonal matrix of the same size as of $\widehat{\mathbf{R}}_{e+n}$. In the case that $\mathbf{\Lambda}$ is in a scaled form of identity matrix, that is $\mathbf{\Lambda} = \alpha\mathbf{I}$, an additional effort in suppressing white noise depending how big the scale factor "α".

In [3], Stoica et al. propose a method of Robust Capon Beamformer such that the true DOA of SOI belongs to a robustness range where steering vectors of assumed DOA and worst case possible true DOA for SOI have the following relation:

$$\left\| \mathbf{a_0} - \mathbf{a}(\theta_{Ap}) \right\|^2 \leq \epsilon \tag{22}$$

where $\mathbf{a_0}$ is the steering vector belongs the true DOA and $\mathbf{a}(\theta_{Ap})$ is that of assumed DOA. Thinking about the implication of norm operation, Eq. (22) suggests that worst case steering vector imperfection is smaller than some scalar "ϵ".

Following the related text, the modification that RCB algorithm brings to MVDR algorithm (Eq. (19)) to estimate beamformer weight vector is as follows:

$$\mathbf{w}_{RCB} = \frac{(\widehat{\mathbf{R}}_{e+n}+\frac{1}{v}\mathbf{I})^{-1}\mathbf{a}(\theta_{Ap})}{\mathbf{a}^{H}(\theta_{Ap})(\widehat{\mathbf{R}}_{e+n}+\frac{1}{v}\mathbf{I})^{-1}\widehat{\mathbf{R}}_{e+n}(\widehat{\mathbf{R}}_{e+n}+\frac{1}{v}\mathbf{I})^{-1}\mathbf{a}(\theta_{Ap})} \text{ such that} \quad (23)$$

$$\sum_{m=1}^{M}\frac{\|\mathbf{z}_m\|^2}{(1+v\gamma_m)^2}=\epsilon \quad (24)$$

where v takes values in the interval:

$$\frac{\|\mathbf{a}(\theta_{Ap})\|-\sqrt{\epsilon}}{\gamma_1\sqrt{\epsilon}}\leq v \leq \min\left\{\sqrt{\frac{1}{\epsilon}\sum_{m=1}^{M}\frac{\|\mathbf{z_m}\|^2}{\gamma_m^2}}\ ,\ \frac{\|\mathbf{a}(\theta_{Ap})\|-\sqrt{\epsilon}}{\gamma_M\sqrt{\epsilon}}\right\} \quad (25)$$

where $\mathbf{z}_{i=1}^{M}$ and $\gamma_{i=1}^{M}$ are eigenvectors and eigenvalues of $\widehat{\mathbf{R}}_{e+n}$ respectively in decreasing order. It is seen that selection of "$\epsilon$" is an issue to start with, and to go on $\epsilon$ is selected as the upper limit of Eq. (22), that is $\epsilon=\|\mathbf{a_0}-\mathbf{a}(\theta_{Ap})\|^2$ . RCB provides a robustness into MVDR algorithm that is already adaptive in its nature in case of no angular mismatch. However, it shall also be noted that RCB uses sample covariance matrix of noise plus interference whereas MVDR uses a theoretical and true covariance matrix. Beamformer SINR comparison of MVDR and RCB implemented for this thesis work can be observed in Figure 7. Note that in order to emphasize the improvement that RCB algorithm brings, sample covariance matrix has been used in both algorithms.

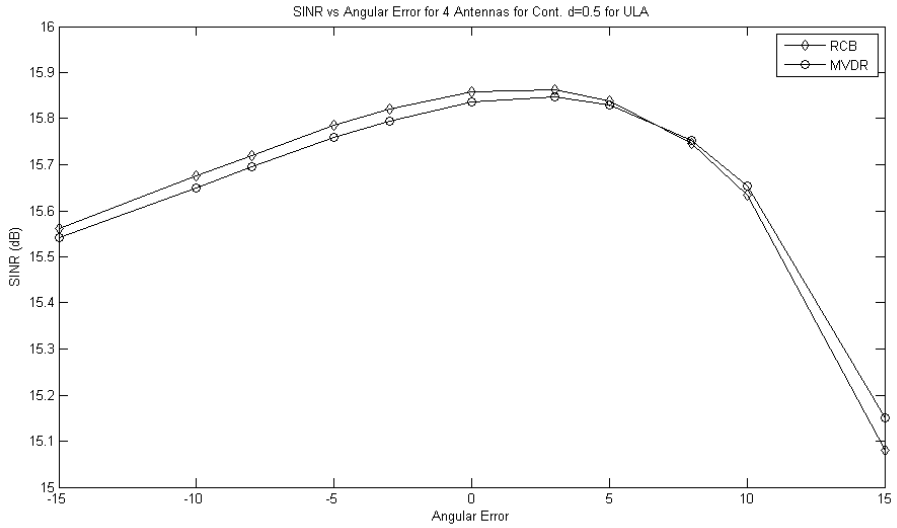


Figure 7. SINR vs. SOI Angular Mismatch Performances of MVDR and RCB for ULA of 4 Antennas and AM Signals Input Scheme with SOI Angle 30° and Jammer Angle 135° both with SNR of 10 dB

RCB algorithm steps are as follows:

1. Find a maximum steering vector error related parameter "$\epsilon$", using robustness range and a priori SOI DOA data according to Eq. (22).

2. Find sample covariance matrix $\widehat{\mathbf{R}}_{e+n}$ during the training session according to Eq. (20)

3. Eigendecompose $\widehat{\mathbf{R}}_{e+n}$ and find related eigenvalues ($\gamma_i$) and eigenvectors ($\mathbf{z}_i$), sorted in descending order as i increases.

4. Find an interval for inverse loading factor (v) according to Eq. (25) using values found in step 1 and 3.

5. Search for v in the interval found in previous step according to Eq. (24).

6. Using the inverse loading factor found in the previous step and a priori DOA of SOI, find beamformer weight vector according to Eq. (23).

## 2.3. Covariance Matrix Reconstruction

An algorithm has been proposed in [4] to overcome effects of inaccurate SOI DOA knowledge or self-nulling due to inclusion of SOI in the training data.

An examination of Eq. (19) reveals that calculation of beamformer weight vector depends on training data covariance matrix and steering vector for SOI, which depends on array geometry, and DOA of SOI. Hence, inaccuracies of any of those would degrade beamforming performance with severity depending on the level of deviation from the correct values.

Due to possible angular mismatch or possibility of self-nulling, a beamformer weight vector is calculated using sample covariance matrix in Eq. (20) and Eq. (19). Then using the roughly estimated beamformer vector, beampattern minima are found. Angular positions of beampattern local minima are marked as interference DOA values. Using the eigendecomposition of sample covariance matrix in conjunction with those marked interferer DOA values, a new covariance matrix of interference-plus-noise is constructed. Using reconstructed covariance matrix a new beamformer weight vector is calculated using Eq. (19). After implementation of this algorithm in a scenario, that one jammer exists for one SOI, which impinge on a ULA of four antennas, a comparison on beamformer output SINR has been made with that of MVDR algorithm. It shall be noted that for this comparison, MVDR method uses sample covariance matrix directly, with result seen in Figure 8. It has been found that due to angular mismatch induced in SOI and jammer proximity, CMR method performs slightly better than MVDR does.

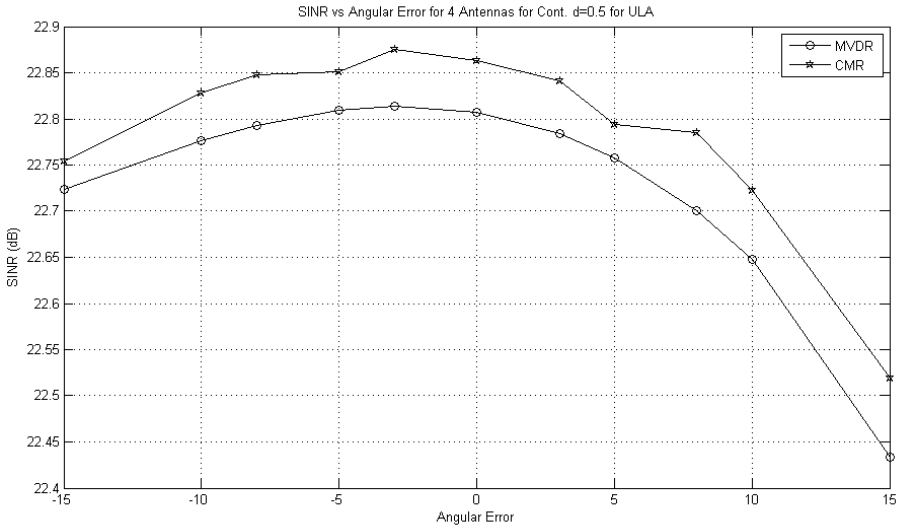


Figure 8. SINR vs. SOI Angular Mismatch Performances of MVDR and CMR for ULA of 4 Antennas and AM Signals Input Scheme with SOI Angle 30° and Jammer Angle 50° both with SNR of 20 dB

The algorithm steps are as follows:

1. Generate sample covariance matrix $\widehat{\mathbf{R}}_{\text{ini}}$ using Eq. (20)

2. Calculate an initial beamformer weight vector $\mathbf{w}_{\text{ini}}$ using Eq. (19)

3. Plot beampattern over [0°, 180°] with angular resolution of interest using the weight vector $\mathbf{w}_{\text{ini}}$

4. Find all nulls of beampattern ($\theta_{i=1}^{K}$) where K < M (Number of array elements)

5. Eigendecompose $\widehat{\mathbf{R}}_{\text{ini}}$ and find minimum and maximum eigenvalues, $\gamma_{\min}$ and $\gamma_{\max}$

6. Generate $\widehat{\mathbf{R}}_{\text{CMR}}$ using:

$$\widehat{\mathbf{R}}_{\text{CMR}} = \gamma_{\max} \sum_{i=1}^{K} \mathbf{a}(\theta_i)\mathbf{a}^{H}(\theta_i) + \gamma_{\min}\mathbf{I} \qquad (26)$$

7. Calculate CMR beamformer weight vector using:

$$\mathbf{w}_{CMR} = \frac{\hat{\mathbf{R}}_{CMR}^{-1}\mathbf{a}(\theta_{Ap})}{\mathbf{a}^{H}(\theta_{Ap})\,\hat{\mathbf{R}}_{CMR}^{-1}\,\mathbf{a}(\theta_{Ap})} \tag{27}$$

The idea behind the algorithm is to find the interference angles by a search rationale and designing a beamformer that equally suppresses them. Addition of a scaled identity matrix includes a suppression for a pseudo-white noise and it also ensures the sample covariance matrix, $\hat{\mathbf{R}}_{CMR}$ to be full rank in the case that number of interferences are smaller than number of antennas. The interference suppression effect of CMR algorithm on MVDR algorithm beampattern is seen in Figure 9.

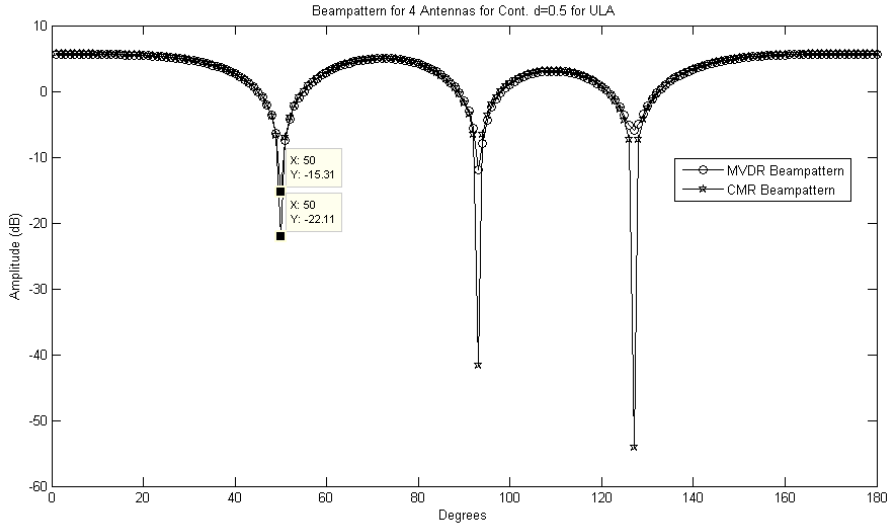


Figure 9. Beampatterns of MVDR and CMR for ULA of 4 Antennas and AM Signals Input Scheme with SOI Angle 30° and Jammer Angle 50° both with SNR of 20 dB

It can be seen that beampatterns of both algorithms are almost the same with the difference that CMR algorithm puts deeper nulls on jammer angles.

## 2.4. Beamforming with Recursive Least Squares Algorithm

Recursive Least Squares algorithm, originally a work of Gauss, has been rediscovered by Plackett [14] and Kalman [15] and has been used for almost a century for adaptive filtering applications ranging from control theory to adaptive signal processing. As beamforming may be viewed as a spatial filtering operation, it has found a suitable place in that area too.

Original RLS algorithm can be qualitatively summarized [16] as: Starting with an initial assumption for covariance matrix and adaptive filter coefficient vector, a next state estimation is made for the signal of interest. Using the error between next state estimate and true next state value of the signal, filter coefficients and covariance matrix is updated with a memory mechanism such that more statistical weight is placed on fresh samples rather than old samples. Then using the updated filter coefficients and covariance matrix the estimation-and-update loop starts again. The step-by-step algorithm is as follows:

**1. Initialization**

1.1. $\mathbf{w}_0=\mathbf{0}$, initial filter coefficients

1.2. $\mathbf{P}(0)=\delta^{-1}\mathbf{I}$, initial covariance matrix inverse

**2. Computation and Update** (for n=1, 2, …)

2.1. $\mathbf{z}(n)=\mathbf{P}(n-1)\mathbf{x}^H(n)$

2.2. $\mathbf{g}(n)=\dfrac{\mathbf{z}(n)}{\lambda+\mathbf{x}^T(n)\mathbf{z}(n)}$

2.3. $\alpha(n)=d(n)-\mathbf{w}_{n-1}^T\mathbf{x}(n)$

2.4. $\mathbf{w}_n=\mathbf{w}_{n-1}+\alpha(n)\mathbf{g}(n)$

2.5. $\mathbf{P}(n) = \frac{1}{\lambda}[\mathbf{P}(n-1) - \mathbf{g}(n)\mathbf{z}^H(n)]$

where $\lambda$ is the forgetting factor and d(n) is the true value of SOI for time=n.

In this work, using RLS update of sample covariance matrix for MI algorithm and modifying the steps covered in [5], an RLS beamformer design has been made with the following steps:

**1. Initialization**

1.1. $\mathbf{P}(1) = \mathbf{I}$, initial covariance matrix inverse

**2. Computation and Update** (for n=1, 2, …)

2.1. Compute $\mathbf{w}(n)$ and $\hat{\mathbf{s}}(n)$ by:

$$\mathbf{w}(n) = \frac{\mathbf{P}(n)\mathbf{a}(\theta_{Ap})}{\mathbf{a}^H(\theta_{Ap})\,\mathbf{P}(n)\,\mathbf{a}(\theta_{Ap})} \tag{28}$$

$$\hat{\mathbf{s}}(n) = \mathbf{w}^H(n)\mathbf{x}(n) \tag{29}$$

where $\mathbf{x}(n)$ is generated by Eq. (5).

2.2. Find estimation error by:

$$\Delta(n) = s(n) - \hat{s}(n) \tag{30}$$

2.3. Find gain vector, $\mathbf{g}(n)$ by:

$$\mathbf{g}(n) = \frac{\mathbf{P}(n)\mathbf{x}(n)}{\lambda + \mathbf{x}^H(n)\mathbf{P}(n)\mathbf{x}(n)} \tag{31}$$

2.4. Find covariance matrix update $\mathbf{P}(n+1)$ by:

$$\mathbf{P}(n+1) = \lambda^{-1}\mathbf{P}(n) - \lambda^{-1}\mathbf{g}(n)\mathbf{x}^H(n)\mathbf{P}(n) \tag{32}$$

As seen in the steps above a beamformer weight vector different than **0** has been selected for initial weight vector and instead of using RLS update for weight vector update, matrix inversion method has been used due to observing that performance is better for the latter one. A very important point to note in the RLS beamforming algorithm is that there is no matrix inversion in the whole algorithm, advantage of which shall be observed in the computational complexity.

# CHAPTER 3

# NON-LINEAR BEAMFORMERS

In the first and large part of this chapter, a non-linear beamformer algorithm (Adaptive Projections Sub-gradient Method) has been explained in detail. This method has been claimed to outperform classical beamformers as an SOI estimator where the number of antennas in the array is less than the number of signals in the scenario. Understanding the basics of this algorithm, another beamforming algorithm (Adaptive Weighted Covariance Matrix) has been proposed in the second part of this chapter. This method uses several mechanisms of previous beamformers explained.

## 3.1. Adaptive Projections Sub-gradient Method

### 3.1.1. Background and Introduction

In [1], a non-linear method of adaptive learning algorithm that might have a use in many areas in engineering has been presented and its performance has been shown for adaptive robust beamforming. It is claimed that APSM algorithm allows an array with elements less than the number of interferers to estimate SOI values in a more successful rate than Minimum Variance beamformers. This algorithm has been a source of attraction for this master thesis work based on that assertion.

Using the following assumptions:

- A priori knowledge of specific scenario that the algorithm will run, e.g.: presumed SOI DOA or ambiguity region for SOI DOA to be expected within,
- Training data of sensor impinged signals and relevant source (of SOI or Interferers) signals (**x**(n), s(n)) to generate a working base for the algorithms,
- A given set of parametric functions.

The algorithm's result is a function **f**(.) of non-linear nature such that $\hat{y} = \mathbf{f}(\mathbf{x})$ is the estimate, which minimizes a specific loss function for the estimation error. The idea summarized above is illustrated in Figure 10.

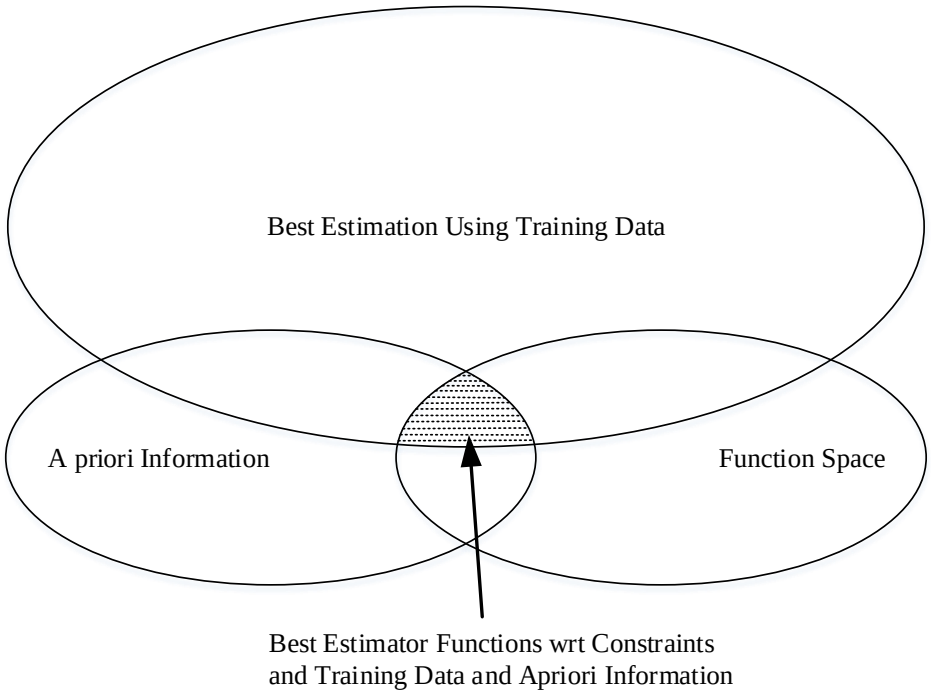


Figure 10. Region of Best Estimators

The random parameters in the specific scenario such as noise or deviation of SOI DOA from presumed DOA determines the exact position of best estimator in the shaded region seen in Figure 10. The relative size of the intersection areas is primarily determined by the relative magnitude of that randomness.

In order to determine the performance of the estimator, a non-negative loss function with quadratic nature can be employed, in the form:

$$\Psi\big(s(n),\mathbf{f}(\mathbf{x}(n))\big)=(s(n)-\mathbf{f}(\mathbf{x}(n)))^2 \tag{33}$$

The selection of a suitable loss function is very important in order to be suitable for optimization and resemble the nature of errors and data that will be encountered in the modeled system.

Incorporating a priori information of the scenario in the form of constraints for loss function minimization and ensuring that the solution is robust against scenario parameter deviations to a degree makes the problem quite complex.

Some further complications may be encountered, such as the necessity to keep the complexity low while avoiding the algorithm to adapt to training data but fail under general scenario. This second phenomenon is known as "overfitting" as stated in [17] make the selection of a dynamic/living algorithm as well as an appropriate loss function very important. The first aim is to make the size of training data small, which is achieved by "sparsification". Overfitting is dealt with a solution of adding a strictly increasing monotonic function, with an upper limit depending on its argument, to the loss function as in [18], the method being called "regularization". The loss function minimization, which gives a solution, then becomes:

$$\min_{\mathbf{f}} \sum_{n=1}^{N} \Psi\big(s(n),\mathbf{f}(\mathbf{x}(n))\big) + \lambda\Phi(\|\mathbf{f}\|) \text{ s.t } \Phi(\|\mathbf{f}\|)\leq\delta \tag{34}$$

where $\lambda$ is a scaling term and $\Phi$ is the regularization term depending on the norm of its argument and $\delta$ is a constant.

One must note that sparsification which leads to limiting training data size, if taken too far, may cause the system to fail in a dynamic scenario. Therefore, a smart system/algorithm design shall incorporate those mechanisms in a way that an optimum trade-off point is reached.

The idea that APSM algorithm relies on is to treat each individual a priori scenario information and training data sample as constraints in form of a convex set in a suitable space; and to find a suitable function in that space which minimizes a specific loss function using all those constraints. APSM method employs convex sets in order to create a mechanic method of projection onto each constraint in order to guarantee that a solution in created in a way that considers all constraints during its generation.

### 3.1.2. Projections onto Convex Sets and Classification

#### 3.1.2.1. Projections onto Convex Sets and Some Projection Definitions

In this section, brief information and definitions about projections will be presented which are used in the APSM algorithm.

Convex Sets

A set C is a convex set if and only if a line connecting any two points in the set is included in the set as a whole, that is:

$$\lambda \mathbf{f_1} + (1-\lambda)\mathbf{f_2} \in C \text{ for } \forall \mathbf{f_1}, \mathbf{f_2} \in C \text{ and } \lambda \in [0,1] \qquad (35)$$

Projection onto Convex Set

For a closed convex set $C \in \Omega$ and any element $\mathbf{f} \in \Omega$, there exists a unique point $\mathbf{f}^* = P_C(\mathbf{f})$ such that $\mathbf{f}^*$ is the closest element of C to $\mathbf{f}$. In this context, $P_C: \Omega \rightarrow C$ is

called “projection mapping” or “projector”. Ω denotes linear vector space, which is a real Hilbert Space. Definition of Hilbert Space has been given in the following sections.

Projection onto Hyperplane

A hyperplane H in Hilbert Space (Ω) is defined as:

$$H \triangleq \{\mathbf{g} \in \Omega: \langle \mathbf{g}, \mathbf{h} \rangle = c\} \text{ for a given set } \mathbf{h} \in \Omega \text{ and } c \in R \tag{36}$$

where projection onto a hyperplane is given as in [19] and [20]:

$$P_H(\mathbf{f}) = \mathbf{f} - \frac{\langle \mathbf{g}, \mathbf{h} \rangle - c}{\|\mathbf{h}\|^2} h, \ \forall \mathbf{f} \in \Omega \tag{37}$$

Illustrations of hyperplane and projection mapping onto hyperplanes are given in Figure 11.

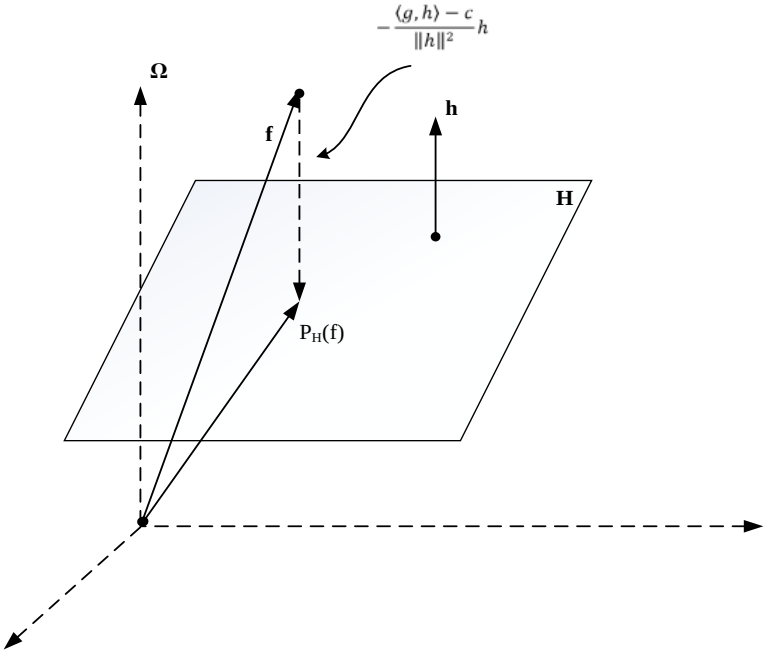


Figure 11. Projection onto a Hyperplane

Projection onto Closed Half Space

A closed half space $H^+$ in Hilbert Space ($\Omega$) is defined as:

$$H^+ \triangleq \{\mathbf{g} \in \Omega: \langle \mathbf{g},\mathbf{h} \rangle \geq c\} \text{ for a given set } \mathbf{h} \in \Omega \text{ and } c \in R \tag{38}$$

where projection onto a closed half space is given in [19] as:

$$P_{H^+}(\mathbf{f}) = \mathbf{f} - \frac{\min\{0, \langle \mathbf{g},\mathbf{h} \rangle - c\}}{\|\mathbf{h}\|^2}\mathbf{h}, \ \forall \mathbf{f} \in \Omega \tag{39}$$

Note that if the inequality turns into an equality in Eq. (38) the set becomes a hyperplane. Illustrations of hyperplane and projection mapping onto closed half spaces are given in Figure 12.

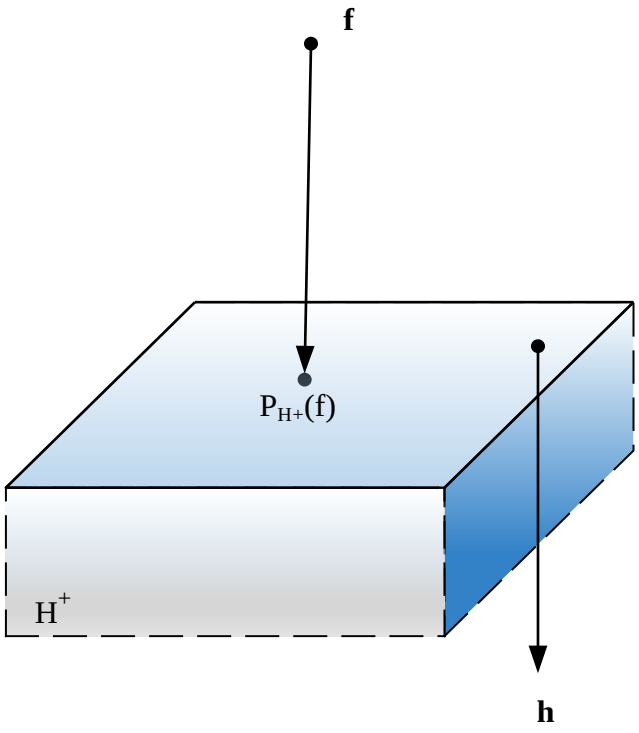


Figure 12. Projection onto a Closed Half Space

Projection onto Hyperslab

A hyperslab S in Hilbert Space (Ω), for an error margin, ε is defined as:

$$S \triangleq \{\mathbf{g} \in \Omega: |\langle \mathbf{g},\mathbf{h} \rangle - c| \leq \varepsilon\} \text{ for a given set } \mathbf{h} \in \Omega \text{ and } c, \varepsilon \in R \tag{40}$$

where projection of a point $\mathbf{f} \in \Omega$ onto hyperslab (S) is given in [19] as:

$$P_S(\mathbf{f}) = \mathbf{f} + \beta_{\mathbf{f}} \mathbf{h}, \forall \mathbf{f} \in \Omega \tag{41}$$

where $\beta_{\mathbf{f}}$ is found by:

$$\beta_{\mathbf{f}} = \begin{cases} \dfrac{c - \langle \mathbf{f},\mathbf{h} \rangle - \varepsilon}{\langle \mathbf{h},\mathbf{h} \rangle}, & \text{if } \langle \mathbf{f},\mathbf{h} \rangle - c < -\varepsilon \\ 0, & \text{if } |\langle \mathbf{f},\mathbf{h} \rangle - c| \leq \varepsilon \\ \dfrac{c - \langle \mathbf{f},\mathbf{h} \rangle + \varepsilon}{\langle \mathbf{h},\mathbf{h} \rangle}, & \text{if } \langle \mathbf{f},\mathbf{h} \rangle - c > \varepsilon \end{cases} \tag{42}$$

Note that a hyperslab is nothing but two intersecting closed half spaces. Illustrations of hyperslabs and projection mapping onto hyperslabs are given in Figure 13.

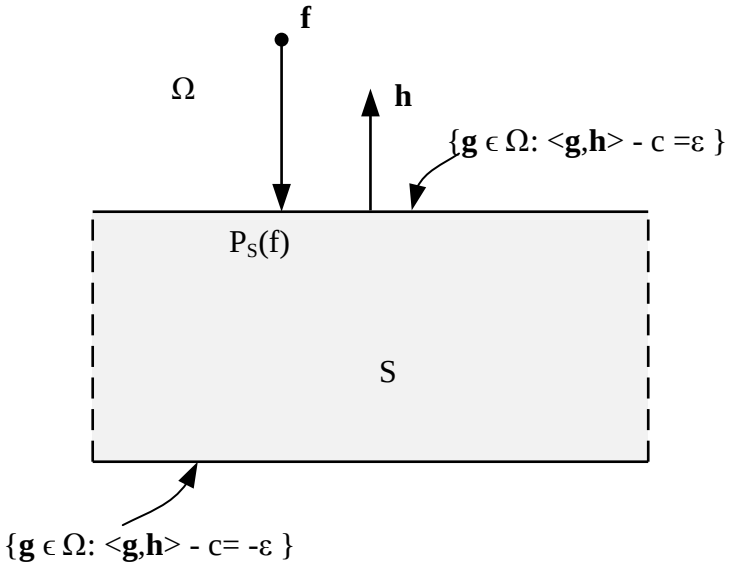


Figure 13.Projection onto a Hyperslab

Projection onto Closed Ball

A closed ball of radius "r" and center as origin of Ω, B(0,r), is defined as:

$$B(0,r) \triangleq \{\mathbf{g} \in \Omega: \|\mathbf{g}\| \leq r\} \text{ for a given } r \in R \tag{43}$$

where projection onto a closed ball is given in [19] as:

$$P_{B(0,r)}(\mathbf{f}) = \begin{cases} \mathbf{f}, & \text{if } \|\mathbf{f}\| \leq r \\ \dfrac{r}{\|\mathbf{f}\|}\mathbf{f}, & \text{if } \|\mathbf{f}\| \geq r \end{cases} \tag{44}$$

Illustration of closed ball and projection mapping onto closed ball is given in Figure 14.

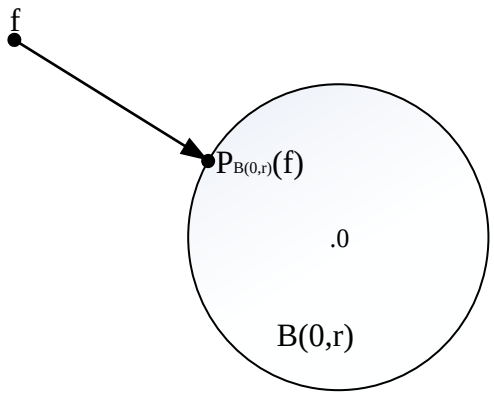


Figure 14. Projection onto Closed Ball

Projection onto Ice-Cream Cone

For cartesian pairs of $(\mathbf{g},r) \in \Omega \times R$ of dimensions seen in Figure 15, an ice-cream cone, K is defined by:

$$K \triangleq \{(\mathbf{g},r) \in \Omega \times R: \|\mathbf{g}\| \leq r\} \tag{45}$$

where projection of a point $(\mathbf{f},\tau) \in \Omega$ x R onto an ice-cream cone (K) is given as in [21] and [22]:

$$P_K(\mathbf{f},\tau)=\begin{cases} (\mathbf{f},\tau), & \text{if } \|\mathbf{f}\|\le\tau \\ (0,0), & \text{if } \|\mathbf{f}\|\le-\tau \\ \frac{\|\mathbf{f}\|+\tau}{2}\left(\frac{\mathbf{f}}{\|\mathbf{f}\|},1\right), & \text{otherwise} \end{cases} \tag{46}$$

Illustration of ice-cream cone and projection mapping onto ice-cream cone is given in Figure 15.

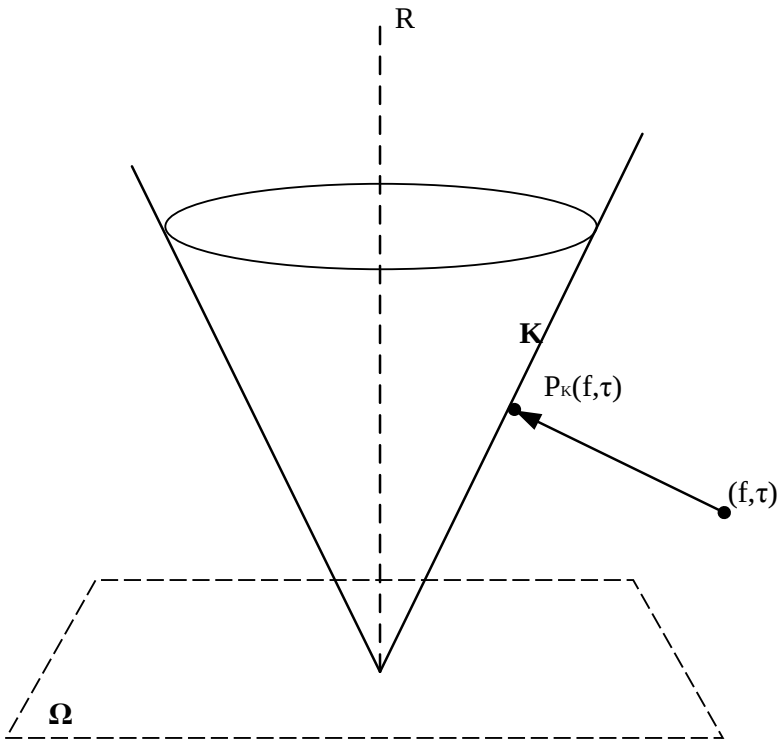


Figure 15.Projection onto Ice-Cream Cone

Fundamental Theory of Projections onto Convex Sets

For $M_i$, i=1, 2,…, n, being finite number of closed convex sets in Hilbert space Ω, the intersection of consecutive projections of a point $f_0$ onto $M_i$ converges to a weak intersection point/region of $M_{i=1}^{n}$. Figure 16 illustrates the idea behind this theory. The dotted lines indicate all possible sets of projections in between $M_1$ and $M_2$ in Hilbert space.

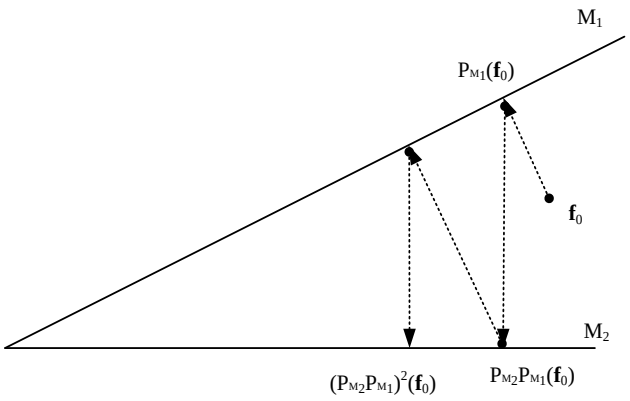


Figure 16. Consecutive Projections onto Closed Sets

Rather than being in a consecutive manner, projections onto convex sets can be performed in a parallel manner too. Theorem in [23] states that for a point $f_0$ in Hilbert Space Ω, and a set of choice of non-negative weights of $w_i$ such that $\sum_{i=1}^{p} w_i=1$, the sequence of summation of parallel projection points is:

$$\mathbf{f}_{n+1}=\sum_{i=1}^{p} w_i P_{Mi}(\mathbf{f}_n) \text{ such that } \sum_{i=1}^{p} w_i=1 \qquad (47)$$

Figure 17 illustrates the idea behind this theory. The dotted lines again indicate all possible sets of projections in between $M_1$ and $M_2$ in Hilbert space.

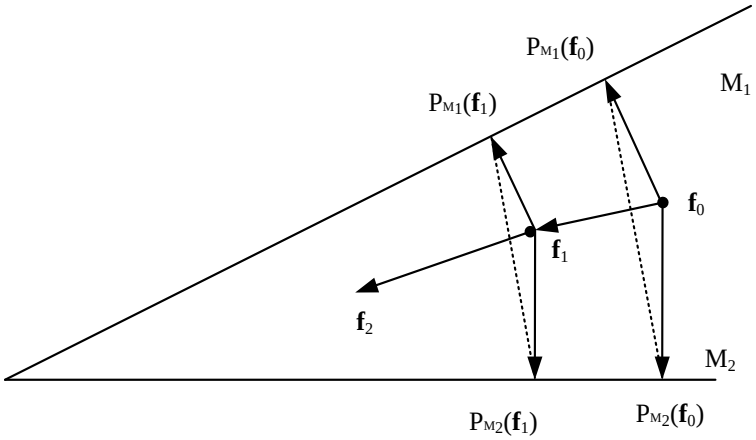


Figure 17. Parallel Projections onto Closed Sets

Different versions and modifications of this theorem can be found in [24], [25] and [26].

In Adaptive Projections Subgradient Method, every new information that becomes available at consecutive time samples are treated as new convex sets generated by a special mapping. Then a parallel projection mechanic over all or most recent data of desired degree, leads us to a weak convergence region for each data. Assume we have sequentially received data for instants of n=1, 2, …. . A modification on Eq. (47) which has been revealed in [27] for the projection mechanism to be run over most recent "q" number of data is:

$$\mathbf{f}_{n+1}=\sum_{i=p-q+1}^{n} w_i P_{Mi}(\mathbf{f}_n) \text{ such that } \sum_{i=p-q+1}^{n} w_i=1 \qquad (48)$$

Note that $w_i=1/q$ for all i, is always a valid and a trivial choice.

### 3.1.2.2.Linear Classification

As stated above the reason for using convex sets in APSM algorithm is to use projections onto these sets as a way of respecting the constraints those sets imply.

In this section, brief information and definitions about linear classification and kernel functions will be presented which are heavily used in the APSM algorithm Using these information, the idea behind the algorithm, solving a non-linear problem in a linear way, will be revealed.

Linear Classification

Choosing a simple case where SOI symbol is $s_n=\pm 1$ and we try to classify the symbol according to array signals ($\mathbf{x}_n$) where symbol estimate is denoted by $\hat{s}_n$. The linear classifier is as follows:

$$\hat{s}_n=\begin{cases}+1 \text{ if } \mathbf{x}_n \in W_1\\ -1 \text{ if } \mathbf{x}_n \in W_{-1}\end{cases} \tag{49}$$

where $W_1$ and $W_{-1}$ are linearly separable. Here the aim is to find a linear function with a margin-of-error $\delta$ such that:

$$\mathbf{f}(\mathbf{x}_n)=\begin{cases}\mathbf{w}^T\mathbf{x}_n+c\geq \delta, \text{ if } s_n=1\\ \mathbf{w}^T\mathbf{x}_n+c\leq \delta, \text{ if } s_n=-1\end{cases} \tag{50}$$

Augmenting $\mathbf{w}\rightarrow [\mathbf{w}^T\ c]^T$ and $\mathbf{x}_n\rightarrow[\mathbf{x}_n^T\ 1]^T$ and adding $s_n$ directly into the equation, Eq. (50) and the task of looking for a classifier becomes finding a $\mathbf{w}$ such that:

$$s_n\mathbf{w}^T\mathbf{x}_n\geq\delta \tag{51}$$

where loss function for each time instant becomes:

$$\Psi_n=\max\{0, \delta-s_n\mathbf{w}^T\mathbf{x}_n\} \tag{52}$$

$\Psi_n$ is called soft margin loss function, the total loss function to be minimized over all data samples with regularization becomes:

$$\sum_{n=1}^{N} \Psi_n + \lambda\Phi(\|\mathbf{f}\|) \qquad (53)$$

A comparison between Eq. (51) and Eq. (38) determines that the classifier is a closed half space. Here we can deduce that a linear classifier function lies in a closed half space and if a point is outside this set, a projection can always be found using Eq. (39).

In [1], APSM algorithm has been implemented in a BPSK input scheme scenario where the beamformer has been used as a BPSK SOI classifier. In this thesis work, beamformers have been implemented as SOI estimators such that SOI ($s_n$) can also take any real value rather than ±1 alone. In this case, estimator for each $s_n$ belongs to a hyperslab, which corresponds to a portion of Hilbert Space for every SOI value. Therefore, estimator projections are made via Eq.(41).

### 3.1.2.3.Reproducing Kernel Hilbert Spaces and Non-Linear Classification

Hilbert Space

A space, which is a closed inner product space and has a norm operation, is called a Hilbert Space [27], which is denoted as Ω, such as:

- Vector Space $R^n$ with inner product defined as: $\langle \mathbf{a},\mathbf{b}\rangle = \mathbf{a}^T\mathbf{b}$ and norm is defined as $\|\mathbf{a}\|^2 = \sqrt{\langle \mathbf{a},\mathbf{a}\rangle}$.
- $L_2$ space square integrable functions defined in the interval $[x_l, x_h]$ with inner product defined as $\langle \mathbf{f},\mathbf{g}\rangle = \int_{x_l}^{x_h} \mathbf{f}(x)\mathbf{g}^H(x)dx$

Kernel

A positive definite function k: $R^n \times R^n \rightarrow R$ is a kernel [28] if there exists a Hilbert Space and a mapping $\phi$: $R^n \rightarrow \Omega$ such that $\forall \mathbf{x},\mathbf{x}' \in R^n$.

$$k(\mathbf{x},\mathbf{x}') \triangleq \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle \tag{54}$$

Example: Gaussian Kernel on $R^n$ with infinite dimensions is defined as:

$$k(\mathbf{x},\mathbf{x}') = e^{-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{\sigma^2}} \tag{55}$$

An illustration of kernel mapping from $R^n$ to high-dimensional Hilbert Space is seen in Figure 18.

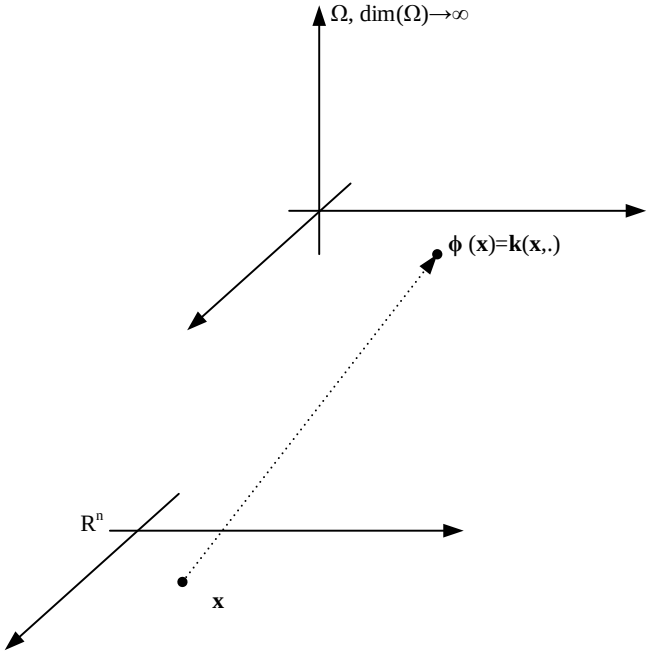


Figure 18. Kernel Mapping from $R^n$ to Hilbert Space using Gaussian Kernel

Reproducing Property of Hilbert Space Kernels

For any x, y $\in R^n$, and a kernel mapping k, inner product of two kernels for two points in Hilbert Space is given as:

$$k(\mathbf{x},\mathbf{y}) = \langle k(\mathbf{x},.), k(\mathbf{y},.) \rangle \tag{56}$$

This property is called as kernel trick and has been used in the algorithm for a way of computing norm and inner product.

Non-linear Classification

Cover's Theorem in [29] states that increasing dimensions of a space which includes two groups to be separated, also increases the probability that these two groups are linearly separable. When faced with a non-linear classification problem, using an appropriate kernel mapping leads us to a high dimensional space. Using the idea presented in Cover's Theorem, we turn the non-linear classification problem in the original space into a linear classification problem in high dimension Hilbert Space as seen in Figure 19.

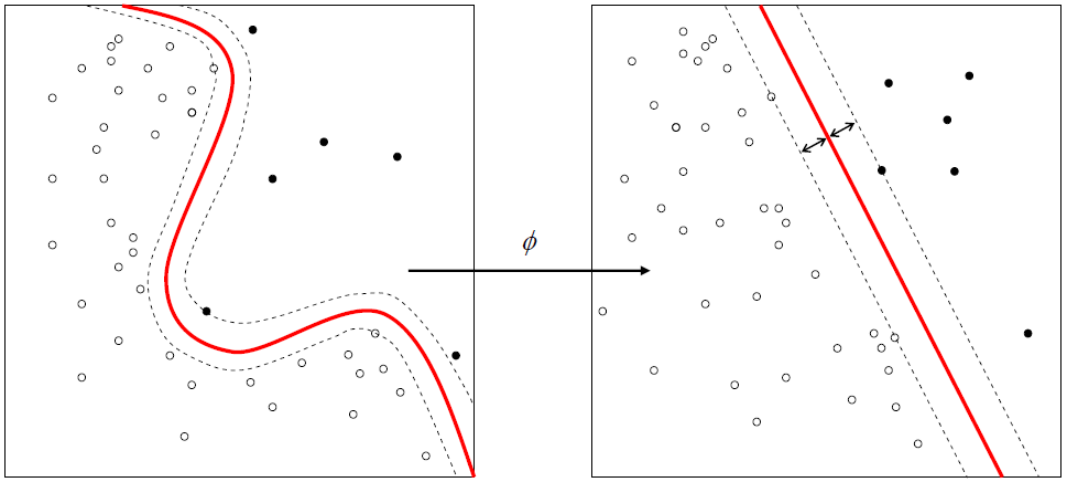


Figure 19. Turning a Non-Linear Classification into a Linear Classification using Kernels

Selecting a suitable kernel is of utmost importance for increasing success of classification of high number of data and having the end-space dimension as high as possible plays a significant role in the success. Gaussian kernel is the most frequent selection due to its dimension is infinite as proven in [30].

Remembering Eq. (34), it is now time to tell how the idea of kernels being used in order to find an estimator. It is stated in Representer Theorem in [31] and [32] that the minimizer of a regularized total loss function, Eq. (53), is of the form:

$$\mathbf{f}_{*}=\sum_{i=1}^{n}\gamma_{i}\mathbf{k}(\mathbf{x}_{i},.) \tag{57}$$

This rationale turns the search of an estimator weight vector of Eq. (51) into a search for $\gamma_i$ for related kernels, which are generated from $\mathbf{x}_n$, sensor data in training period for relative time indices.

#### 3.1.2.4.Robustness and A priori Information

Among the beamforming algorithms, the issue of robustness is incorporated into algorithm through an interval that includes a priori information. For e.g.: with a priori DOA of SOI being 30° and a robustness range of 10° means we design a beamformer steered at 30° while the algorithm may tolerate and acknowledge receptions in the interval [25°, 35°] as legitimate SOI receptions.

In the case of APSM algorithm, robustness and a priori information is added in the form of a ball with its center determined by a priori DOA of SOI and the radius adjusted according to the robustness range. Considering the norm and inner product as a measure of distance, a point in Hilbert space **f** is an estimator that makes the following inequality hold:

$$\langle \mathbf{f},\mathbf{k}(\tilde{\mathbf{x}},.)\rangle \leq \varepsilon \tag{58}$$

where $\mathbf{k}(\tilde{\mathbf{x}},.)$ denotes kernel belonging to a training data coming from a priori DOA of SOI and ε is related to maximum distance between $\mathbf{k}(\tilde{\mathbf{x}},.)$ and $\mathbf{k}(\overline{\overline{\mathbf{x}}},.)$. Term $\mathbf{k}(\overline{\overline{\mathbf{x}}},.)$ denotes kernel belonging to a training data coming from boundary of angle of robustness range. The idea of robustness implies that Eq. (58) holds for all the points

inside the ball, $B[\mathbf{k}(\tilde{\mathbf{x}},.),\delta(\varepsilon)]$, for the estimator $\mathbf{f}$ we are looking for, which corresponds to infinite number of inequalities to hold simultaneously. In short, we include robustness into our estimator as looking for it in the set denoted by:

$$\Gamma \triangleq \{\mathbf{f} \in \Omega: \langle \mathbf{f},\mathbf{h} \rangle \leq \varepsilon, \forall \mathbf{h} \in B[\mathbf{k}(\tilde{\mathbf{x}},.),\delta(\varepsilon)]\} \tag{59}$$

It has been revealed in [21] and [33] that finding an $\mathbf{f}$ and subjecting it into an infinite number of inequalities is analogous to finding a real number $\tau$ such that the point $(\mathbf{f}, \tau)$ belongs to the intersection of an ice-cream cone (K) and a hyperplane (H) as defined below:

$$K \triangleq \{(\mathbf{f},\tau) \in \Omega \text{ x } R: \|\mathbf{f}\| \leq \tau\} \tag{60}$$

$$H \triangleq \{(\mathbf{f},\tau) \in \Omega \text{ x } R : \langle (\mathbf{f},\tau), B[\mathbf{k}(\tilde{\mathbf{x}},.),\delta(\varepsilon)] \rangle = \varepsilon\} \tag{61}$$

### 3.1.2.5.Sparsification

During the training phase, array signal receptions are turned into kernels using Gaussian kernel formula in Eq. (55) and each one is tested for the value of its information. If it is computationally inefficient to add the training data sample group of convex sets seen in Eq. (57), the sample is discarded and simply treated as inexistent. This idea has been summarized in [34].

In APSM algorithm as each training data arrives per each iteration, distance of the kernel belonging to newest training data is calculated to already existing kernels of the previous data. If the sum of distance to all kernels is smaller than some pre-defined parameter "μ" then the newly defined kernel is not taken into account as far as the algorithm is involved and the mechanism of training period continues for the next sample.

### 3.1.3. APSM Algorithm

Having given all the data related to the ideas and derivation of the APSM algorithm in the previous sections, the summary and the steps of the algorithm are given in the following sections.

#### 3.1.3.1.Summary

APSM algorithm has been developed in order to deal with non-linear problems in a time-adaptive manner. The idea is to treat every bit of information such as a priori knowledge of the scenario faced, robustness range and training data as constraints. Then map these constraints into convex sets using kernel functions where an initial point of solution is projected onto in a high dimensional space of kernel functions representing the constraints to be respected. During the training session where we control the scenario in order to prepare an operational solution, a set of convex sets and their coefficients, which determine the level of "importance" that every training data sample, has in the solution. An intermediate phase of sparsification is included in the training session to check the level of "uniqueness" of the information that each training sample holds according to a level of threshold. If sample is not unique enough to be included, the training sample is left out of solution set.

#### 3.1.3.2.Algorithm Steps

**1. <u>Training Phase</u>**

1.1. A non-negative real sparsification parameter, "μ" is selected at will depending on the level of importance attributed to training session. If a priori information is deemed precise, μ might be chosen to be large (approaching 1), if more emphasis is placed upon training session then μ is chosen to be as small as possible.

1.2. A pre-processing operation is included in the algorithm in order to deal with real kernel vectors belonging to complex sensor data such as:

$$\mathbf{v_{n,1}}(\theta)=\begin{bmatrix}\text{Real}(\mathbf{x}_n(\theta))\\ \text{Imag}(\mathbf{x}_n(\theta))\end{bmatrix},\mathbf{v_{n,2}}(\theta)=\begin{bmatrix}\text{Imag}(\mathbf{x}_n(\theta))\\ \text{-Real}(\mathbf{x}_n(\theta))\end{bmatrix} \quad (62)$$

1.3. A preamble basis is prepared using a priori SOI DOA, maximum deviated SOI DOA and SOI modulation type. Using those, sensor data ($\mathbf{x}_n$) are generated according to Eq. (1) using randomly generated but known SOI values alone. No interferers or noise components are used. It is important to know the pool of SOI signal values to generate an able preamble basis. The number of elements in the preamble basis is left to vary according to the designer's will. Its function is to make an initial estimator set according to a priori information and build projection balls as mentioned in section 3.1.2.4 Robustness and A priori Information. Let $\mathbf{x}_n(\theta_{ap})$ denote complex valued sensor data for signal coming from a priori DOA of SOI without any interference or noise and $\mathbf{x}_n(\theta_{rob})$ denote complex valued sensor data for signal coming from boundary of DOA robustness range of SOI without any interference or noise; the preamble basis (**Π**) looks like:

$$\mathbf{\Pi}=\left[\mathbf{v_1}(\theta_{ap})\ \mathbf{v_2}(\theta_{ap})\ \mathbf{v_1}(\theta_{ap})\ \ldots\ \mathbf{v_1}(\theta_{rob})\ \mathbf{v_2}(\theta_{rob})\ \ldots\right] \quad (63)$$

The relative number of elements in **Π** determine focus of final projections into robustness ( $\mathbf{v}_{1,2}(\theta_{rob})$) or a priori SOI DOA ( $\mathbf{v}_{1,2}(\theta_{ap})$).

1.4. An initial solution set is initialized with basis (**β**) with its initial elements being **Π** and equal-element coefficient vector ($\gamma_{APSM}$) for its elements.

1.5. As each training sample arrives at time instant "n", related algorithm inputs ($\mathbf{v}_n$) are generated according to Eq. (62) and the distance between $\mathbf{v}_n$ and **β** is calculated**.**

1.6. If the distance is less $\mu$ selected in step 1, then $\mathbf{v}_n$ is added to $\boldsymbol{\beta}$; if distance is small, that is the information that the new training sample bears is already in the Basis ($\boldsymbol{\beta}$) with some difference margin, the data is discarded.

1.7. The new element of $\boldsymbol{\beta}$, is projected onto hyperslabs belonging to previous basis elements in Hilbert Space. The coefficient vector $\boldsymbol{\gamma}_{APSM}$ is updated as an inner product according to Eq. (41) with its number of elements increased for the new basis element.

1.8. The solution basis ($\boldsymbol{\beta}$) and its coefficient vector ($\boldsymbol{\gamma}_{APSM}$) is pseudo-projected onto the robustness balls as defined in section 3.1.2.4 Robustness and A priori Information using hyperplane and ice-cream cone projectors.

1.9.For each new training data arrival, return to step 1.5 and go over the steps until last training data arrives.

1.10. At the end of training session with the processing of last training sample, freeze $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}_{APSM}$ as the solution set of estimator looked for.

1.11. The beamformer solution is in the form:

$$\mathbf{f}_* = \sum_{i=1}^{N_{\boldsymbol{\beta}}} \gamma_i \mathbf{k}(\mathbf{x}_i, .) \tag{64}$$

where $N_{\boldsymbol{\beta}}$ denotes total number of convex projection sets in the basis matrix.

**2. <u>Operational Phase</u>**

2.1. At reception of each new test data sensor signals, related $\mathbf{v}_n$ is found according to Eq. (62)

2.2. $\mathbf{v}_n$ is projected onto elements of $\boldsymbol{\beta}$ using Eq. (55) and a weighted summation over the projection results is made using $\boldsymbol{\gamma}_{APSM}$ corresponding to each projection set. The beamformer output is computed as:

$$y_{APSM}(t)=\sum_{i=1}^{N_{\boldsymbol{\beta}}} \gamma_i \, k(\mathbf{x}_i,\mathbf{x}(t)) \tag{65}$$

2.3.The numeric result of step 2.2 is mapped to its SOI estimate based on the mechanism explained in section 1.5 Beamformers for SOI Estimation.

Having a small threshold value "μ" populates the basis matrix (**β**) which is used to provide projection sets for the sensor data. There is a rough relation between the μ value and the number of elements of **β** as follows:

$$N_{\boldsymbol{\beta}}= \begin{Bmatrix} \approx 2N_{tra}, \text{ for } \mu<0.6 \\ \text{quadratic wrt } N_{tra} \text{ , for } 0.6<\mu<0.9 \\ \approx \ 12(1\text{-}\mu)\ N_{tra}+N_{pre}, \text{ for } \mu>0.9 \end{Bmatrix} \tag{66}$$

where $N_{\boldsymbol{\beta}}$ denotes total number of convex projection sets in the basis matrix, $N_{\boldsymbol{\Pi}}$ denotes total number of convex projection sets in the preamble basis matrix and $N_{tra}$ denotes total number of training data samples.

### 3.2. Adaptive Weighted Covariance Matrix

Implementing and understanding APSM algorithm lead to the design of a novice algorithm which was a synthesis algorithm of:

- Matrix Inversion idea seen in Eq. (19),
- Covariance matrix reconstruction idea of section 2.3 Covariance Matrix Reconstruction,
- Error induced mechanism of Eq. (30) of RLS algorithm,
- Look up & sparsification methods employed in APSM algorithm.

The motivation is as follows: during the initial phases of training session, an initial beamformer weight vector is computed using the training data received until then. Then an iteration during the rest of the training period is started such that using the designed beamformer, an estimate of the next training sample is found. Using the magnitude of difference between the estimate and the actual training sample for the SOI, sample covariance matrix is adaptively. During covariance matrix update, different weights are applied on each training data per iteration depending on estimation error. Estimation error for a training sample is viewed as a measure of new information that sample bears for the estimation mechanism. Iteration ends with the final training sample and the final beamformer is used as is, during the operational scenario.

The steps can be explicitly stated as:

**1. Initialization**

1.1. Chose a sparsification parameter "μ".

1.2. Compute sample covariance matrix (**R**) according to Eq. (20) using the first $\alpha N_{tra}$ snapshots where α is a positive real number close to 0.

1.3. Compute beamformer weight vector according to Eq. (19) using the sample covariance matrix computed in the previous step.

1.4. Set number of elements of sample covariance matrix, $N_R = \alpha N_{tra}$

**2. Intermediate Training Session**

2.1. Using the beamformer weight vector already available, estimate the current SOI value according to:

$$\widehat{s_n}=\mathbf{w}_{n-1}{}^{H}\mathbf{x}_n \tag{67}$$

2.2. Find the estimation error magnitude: $\Delta_n=|s_n-\widehat{s_n}|$

2.3. If estimation error is large compared to sparsification threshold, that is $\Delta_n > \mu$, update the sample covariance matrix according to Eq. (68) and increment $N_R$ and update beamformer weight vector according to Eq. (19) using the sample covariance matrix updated in Eq. (68):

$$\mathbf{R}_{n+1}=\frac{N_R\mathbf{R_n}+\Delta_n{}^2(\mathbf{x}_n-\mathbf{a}(\theta_{ap})s_n)\,(\mathbf{x}_n-\mathbf{a}(\theta_{ap})s_n)^{H}}{N_R+1}, \tag{68}$$

2.4. Return to step 2.1 for the next training sample and run until the last training data.

It can be intuitively understood that having smaller values of $\mu$ increases the total number of training period data samples used in the generation of sample covariance matrix.

# CHAPTER 4

# COMPARATIVE PERFORMANCE EVALUATION OF BEAMFORMER ALGORITHMS

Beamformer algorithms, which were presented in previous chapters, have been implemented in MATLAB for this thesis work. Their signal-of-interest estimation performances have been compared via MATLAB simulations according to several performance criteria. The organization of this chapter is as follows: first, simulations have been explained with basic steps and major parameters. Then performance criteria are given with their definitions and the particular reasons for employment. Before presenting comparisons of algorithms for different scenario parameters, an analytical approximation for symbol error rate has been mentioned and its similarity with MATLAB simulations for simple beamforming scenario is checked. Then for the rest of the chapter, beamformer comparisons have been made with varying specific scenario parameter. Finally a summary of performance comparisons is given.

## 4.1. Comparison Descriptions

Each simulation has been made in two sections:

1. **Training session:** A number of snapshots have been supplied to each algorithm with true signal value knowledge for SOI. The signal values used for training signals for each algorithm are the same, complex valued sensor measurement noise have been added to snapshot data which are randomly generated for each training session for each algorithm. At the end of training sessions, the input of testing session has been generated which will be used to estimate SOI values using test data. Input to test session for all the algorithms other than APSM

algorithm is beamformer weight vector while for APSM algorithm a look-up table and relevant coefficients for each element of that look-up table are the inputs to test session.

2. **Testing session:** A number of snapshots with a higher number than given in training session have been supplied to each algorithm for testing session of each simulation. The signal values used to generate testing signals impinging on sensors are the same with sensor measurement noise added are complex valued and randomly generated for each run. Using the output of the training session (a weight vector for array elements or a look-up table with coefficients) the algorithms output an estimate of SOI value for each test snapshot.

### 4.2. Simulation Parameters

Unless otherwise stated, 200 snapshots have been used for training session in the comparisons made and the testing phase have been made using 100.000 snapshots for each run. "$\mu_{AWCM}$" parameter has been taken as 0.2 for AWCM algorithm. "$\mu_{APSM}$" parameter has been taken as 0.95 and 1 for APSM algorithm. In some cases, "$\mu_{APSM}$" has been taken as 0.25 or 0.3 for additional presentation of the effect of it in APSM algorithm. Robustness angle for RCB and APSM are both taken as 5°. In the nominal case, antenna arrays are ULA of four elements with inter-elemental spacing d=0.5λ. SOI and Jammer SNR values are 10 dB each. Only one parameter is swept for each simulation in order to isolate the full effect of the analyzed parameter on the performance of the algorithms. Parameters sweep information for each algorithm will be presented in relevant sections of this chapter.

### 4.3. Performance Criteria

Signal-to-Interference-Noise Ratio is a frequently used performance criteria in beamforming. However, because the algorithms have been implemented as SOI estimators, simulations that compared symbol error rate of algorithms were made. Performance evaluation of the algorithms has been made using the following

parameters: SER, SINR and Computational Complexity. Simultaneous comparison of linear beamformers with non-linear beamformers without forced the use of performance criteria that do not use beamformer weight vectors.

For BPSK and QPSK input/output schemes, digitization step of beamformer output seen in Figure 6 makes SER a reasonable criterion of performance with the following definition:

$$\text{SER}= \frac{\text{Number of test symbols estimated wrong}}{\text{Total number of test symbols}} \tag{69}$$

In linear beamforming, SINR is defined as follows:

$$\text{SINR}= \frac{\mathbf{w}^{H}\mathbf{R}_{s}\mathbf{w}}{\mathbf{w}^{H}\mathbf{R}_{e+n}\mathbf{w}} \tag{70}$$

In order to estimate SINR using finite beamformer output samples, Eq. (71) has been given in [35] and [36]:

$$\text{SINR}= \frac{\frac{1}{N}\sum_{i=1}^{N}\left|\mathbf{w}^{H}\mathbf{a}(\theta_1)s_1(t_i)\right|^2}{\frac{1}{N}\sum_{i=1}^{N}\left| y(t_i)-\mathbf{w}^{H}\mathbf{a}(\theta_1)s_1(t_i)\right|^2} \tag{71}$$

However, the lack of a linear beamformer weight vector in APSM made a performance criterion free of weight vector necessary for AM signal input/output scheme. Assuming that $\mathbf{w}^{H}\mathbf{a}(\theta_1)=1$, a modified version of Eq. (71) has been used as SINR estimate with the following definition:

$$\text{SINR}_{\text{used}} = \frac{\frac{1}{N}\sum_{i=1}^{N}|s_1(t_i)|^2}{\frac{1}{N}\sum_{i=1}^{N}\left| y(t_i)-s_1(t_i)\right|^2} \tag{72}$$

where $s_1(t)$ is the SOI and $y(t)$ is the beamformer output. Usage of this SINR definition for BPSK and QPSK signals scheme gives parallel results to usage of

SER. That is, in a case where SER seems to decrease for a direction of parameter sweep (algorithm estimation performance appreciating), SINR increases.

For computational complexity comparisons, MATLAB tic/toc command results have been used to get algorithm computation times.

## 4.4. Comparison Results

### 4.4.1. Analytical Checkout of Simulation Results

In this section, an analytical estimation of symbol error rate has been studied. Starting from the well-established telecommunications theory, an approximation for SER has been made for beamforming. Then simulation results have been compared with analytical numerically.

For simplicity, BPSK input/output scheme has been taken as a starting point. In [37] SER for estimating a BPSK signal in Gaussian noise has been given as:

$$SER_{BPSK} = \frac{1\text{- erf}\left(\sqrt{SNR_{Linear}}\right)}{2} \tag{73}$$

To extend this formula into beamforming, the concept of array gain given in [13] can be used. Array gain is the ratio of output SNR of a beamformer to its input SNR. That is, SNR improvement of the beamformer. In a ULA of M antennas, which is perfectly steered to SOI, the SNR of the signal is amplified by a factor of "M" at the beamformer output. Therefore, SER can be approximated as:

$$SER_{BPSK} = \frac{1\text{- erf}\left(\sqrt{M\ SNR_{Linear}}\right)}{2} \tag{74}$$

For s source of interest that transmits a signal $s_{SOI}$ onto a ULA of M antennas in a Gaussian noise environment with zero mean and $\sigma^2$ variance, the beamformer output is normally distributed as $N(Ms_{SOI},M\sigma^2)$. It is well-established fact in [31] that a "hyper plane" is the optimal Bayesian classifier between variable classes with variations of same magnitude. Figure 20 is an illustration of decision mechanism for beamformer output for s specific time instant, where in dimension of our variables, the optimal classifier becomes a straight line. The transmitted symbol is "1", and the beamformer output mean is "M" which is antenna number. The beamformer noise is normal noise with variance $M\sigma^2$ around M. It is easily deduced that in the ideal case, increasing antenna number decreases probability of error and decreasing noise power improves the beamformer SOI estimation performance by lowering error probability.

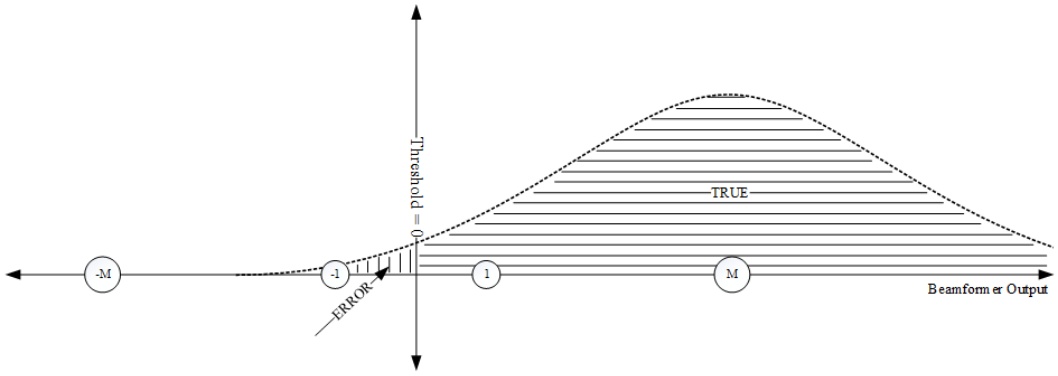


Figure 20. BPSK Error Illustration

Inserting M=4 and SNR=0 dB to Eq. (74) above yields an error probability of 0.0023 which is approximately correct for the simulation result seen in Figure 21 of a scenario of 0° angular error and 0 dB SNR.

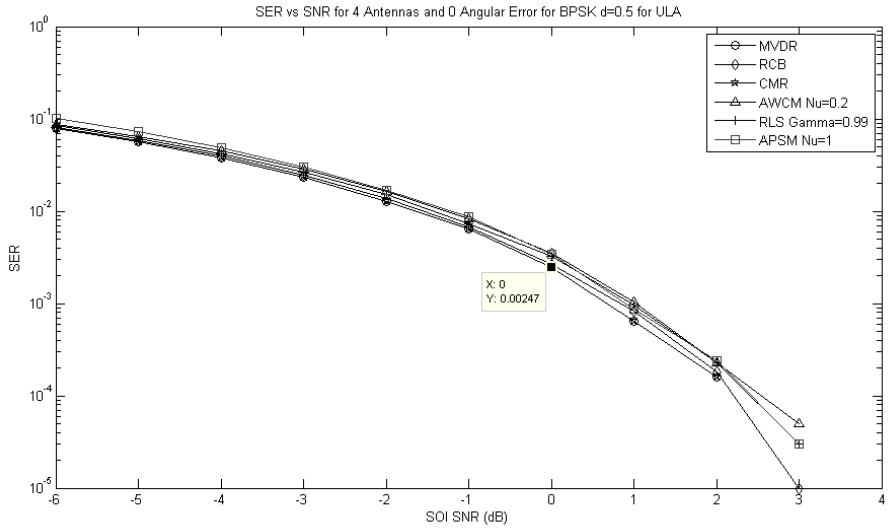


Figure 21. SER vs SOI SNR for ULA of 4 Antennas and 0° Angular Mismatch for BPSK Input Scheme

The QPSK case probability of error for a signal transmitted in the simple case: "one-signal in Gaussian noise environment" turns out to be:

$$SER_{QPSK} = \frac{1 - \text{erf}\left(\sqrt{\frac{1}{2} M SNR_{Linear}}\right)}{2} \cdot \frac{3 + \text{erf}\left(\sqrt{\frac{1}{2} M SNR_{Linear}}\right)}{2} \qquad (75)$$

Inserting M=4 and SNR=0 dB to Eq. (75) yields $SER_{QPSK}$ as 0.045 and the simulation result for this case is in Figure 22. It can be seen that the analytical error approximations hold for the simulations. At this point, it is claimed that the simulations, results of which are symbol error rates for each case, are correct and reliable. In the following sections of this chapter, simulation results where a single parameter in the scenario is being swept and SER or SINR is being given depending on the input/output signal type are going to be presented to make a comparison of algorithms for each parameter.

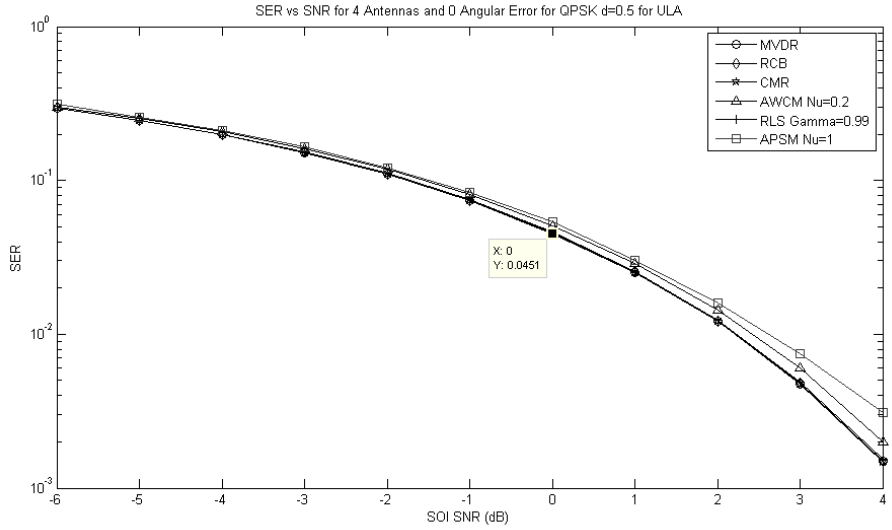


Figure 22. SER vs SOI SNR for ULA of 4 Antennas and 0° Angular Mismatch for QPSK Input Scheme

### 4.4.2. Angular Error

In angular mismatch comparisons, two signal sources with 10 dB SNR have been placed on 30° and 135° with the one at 30° is the SOI. Sources have been selected angularly so far in space to make sure that no performance loss dues to jammer-to-SOI interference occurs. A priori SOI DOA value has been swept in the interval [15°-45°]. Beamforming algorithms have been implemented for BPSK, QPSK and AM signal case I/O schemes with ULA (d=0.5λ) array and AM signal scheme for UCA (d=0.5λ). Figure 23 shows that all the algorithms are very robust against angular mismatches for BPSK inputs.

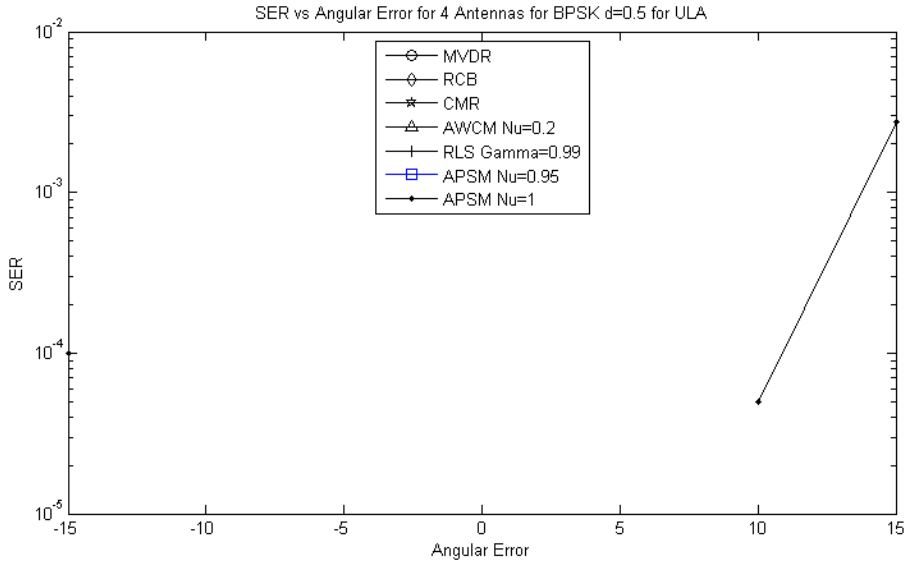


Figure 23. SER vs Angular Mismatch for ULA of 4 Antennas and BPSK Input Scheme with SOI Angle 30° and Interference Angle 135° both have SNR of 10 dB

Figure 24 shows that APSM ($\mu$=0.95) performs as good as other algorithms in QPSK case. However, a quick comparison between Figure 23 and Figure 24 shows that QPSK performance of algorithms are worse than that of BPSK, an observation which is in agreement with Eq. (74) and Eq. (75).

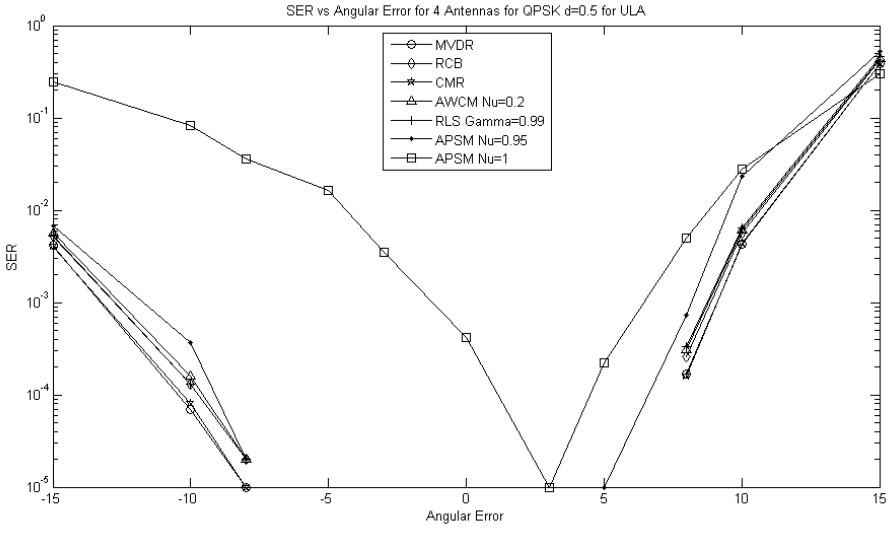


Figure 24. SER vs Angular Mismatch for ULA of 4 Antennas and QPSK Input Scheme with SOI Angle 30° and Interference Angle 135° both have SNR of 10 dB

A look on Figure 25 shows that use of APSM algorithm is meaningless for AM signal I/O scheme. This is because of the fact that APSM algorithm works through projection sets belonging to training sensor data, SOI pairs ($\mathbf{x}_n$,$s_n$). However, there are not enough training data pairs to build a basis matrix ($\boldsymbol{\beta}$) to map the sensor test data to relevant SOI value.

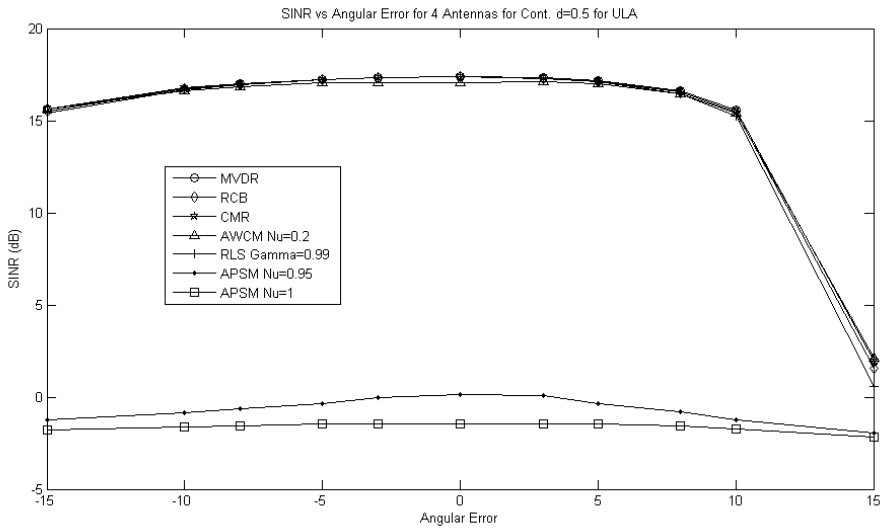


Figure 25. SINR vs Angular Mismatch for ULA of 4 Antennas and AM Signal Input Scheme with SOI Angle 30° and Interference Angle 135° both have SNR of 10 dB

Simulation results on the effect of angular error in UCA for AM I/O scheme can be seen in Figure 26 and comparing it with Figure 25 shows that circular array's SINR performance is more flat and symmetrical over the angular error range and no performance loss is observed as of its ULA counterpart. This phenomena has been encountered and explained in [7], beampattern main lobe is of same width in UCAs over the whole azimuth whereas in ULAs it gets narrower as array is steered to its boresight.

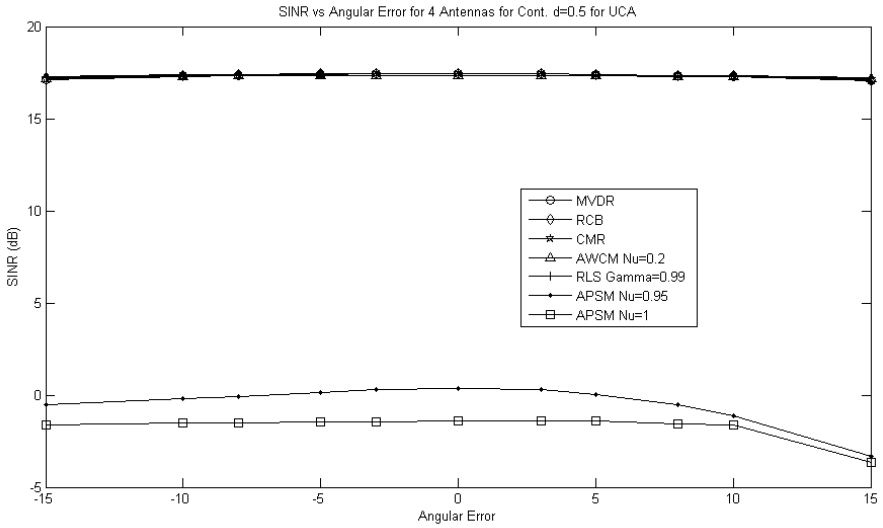


Figure 26. SINR vs Angular Mismatch for UCA of 4 Antennas and AM Signal Input Scheme with SOI Angle 30° and Interference Angle 135° both have SNR of 10 dB

The effect of using a smaller μ parameter for APSM algorithm is seen in the performance improvements observed in each figure. As stated in Training Phase step 1.1 of section 3.1.3.2 Algorithm Steps of APSM, using a smaller μ puts more importance on training data rather than precision of a priori data and the algorithm becomes more robust to angular mismatches in the end.

### 4.4.3. Snapshot Count

In the comparisons made for number of training snapshots, two signal sources with 10 dB SNR have been placed on 30° and 60° with the one at 30° is the SOI. Number of snapshots used have been swept in the interval [20, 100]. In order to get enough symbol errors to make conclusions, some level of angular mismatch have been induced in the simulations. Beamforming algorithms have been implemented for BPSK, QPSK and AM signal I/O schemes with ULA (d=0.5λ) and UCA (d=0.5λ). Figure 27 depicts the performance of algorithms, which re-emphasizes the effect of μ parameter in APSM algorithm. As seen for APSM (μ=1), increasing number of snapshots does not improve the performance because the training samples are simply not included in the solution basis (**β**). However, even a small decrease in μ

parameter shows its effect for the performance of APSM (μ=0.95) with increasing number of training data. Proportional effect of μ on the size of basis and number of training data samples has been given in Eq. (66)

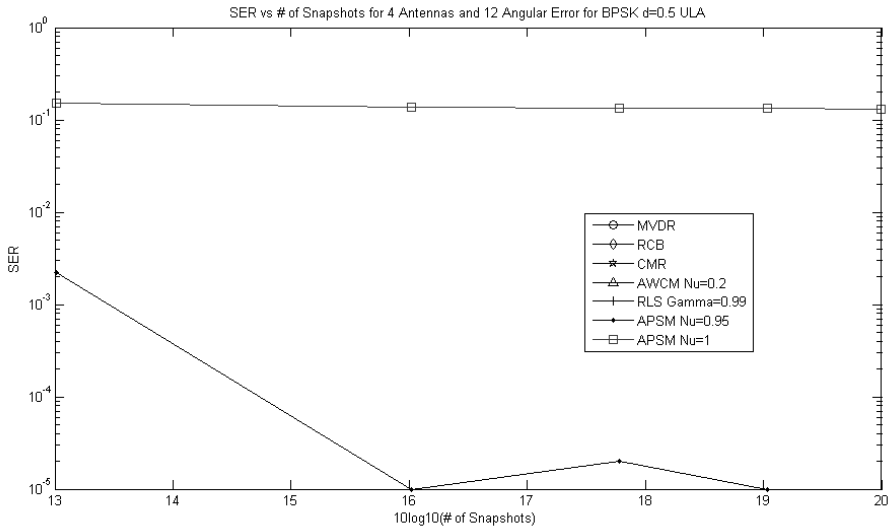


Figure 27. SER vs Number of Snapshots for ULA of 4 Antennas and BPSK Input Scheme with SOI Angle 30° and Interference Angle 60° both have SNR of 10 dB in 12° Angular Mismatch

Increasing number of snapshots for the generation of beamformer improves the performance.

### 4.4.4. Input SNR

In the comparisons made for SOI SNR, two signal sources have been placed on 30° and 60° with the one at 30° being the SOI and changing SNR, the other signal is jammer with fixed SNR of 10 dB. SOI SNR has been swept in the interval [0, 12 dB] with increments of 2 dB while jammer SNR is kept fixed at 10 dB. No angular mismatch is induced. Simulations have been made for BPSK and QPSK schemes for both ULA and UCA geometries with d=0.5λ.

It can be observed in all figures of this section that using APSM (μ=1) at low SNR cases almost makes no sense due to the fact that the algorithm runs on the preamble basis which has SOI information alone. Using perfect SOI information with no

jammers or noise in basis production results in bad performance when met with low SOI SNR reality of the test scenario. Decreasing μ makes APSM algorithm a better "learner" in the training session and gives it a performance boost to meet with its MI counterparts.

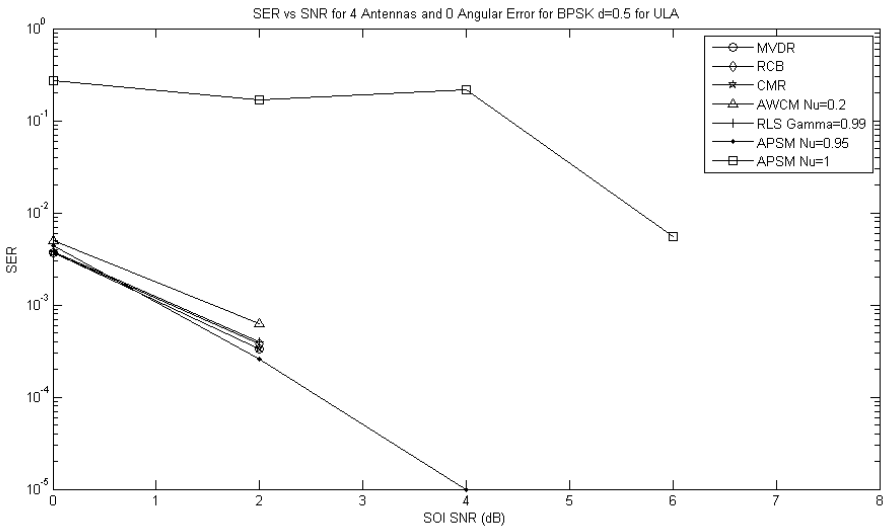


Figure 28. SER vs SOI SNR for ULA of 4 Antennas and BPSK Input Scheme with SOI Angle 30° and Interference Angle 60° with SNR of 10 dB in 0° Angular Mismatch

A quick comparison between Figure 28 and Figure 29 reaffirms universal QPSK case performance loss of SOI estimators compared to BPSK case.

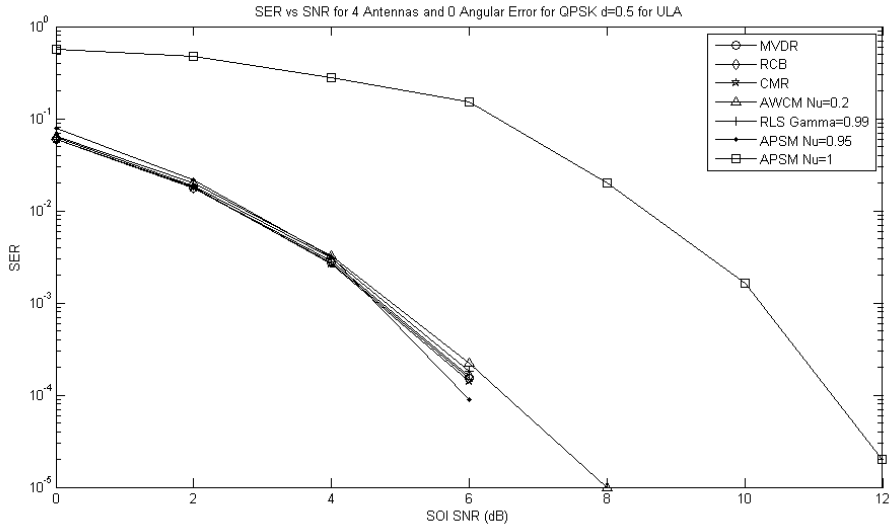


Figure 29. SER vs SOI SNR for ULA of 4 Antennas and QPSK Input Scheme with SOI Angle 30° and Interference Angle 60° with SNR of 10 dB in 0° Angular Mismatch

Simulation results seen in Figure 28 and Figure 29 coincide with expectations that are derived from Eq. (74) and Eq. (75). That is, estimation performance improves with input SNR.

A comparison between Figure 28 with Figure 30 and Figure 29 with Figure 31 shows that estimation performance of MI algorithms degrades when the array geometry is changed from ULA to UCA. This is because the array aperture facing the sources are different when the array geometry is changed, as seen in Figure 32. This phenomenon is dependent on azimuth angle of the source and lengthily discussed in [7].

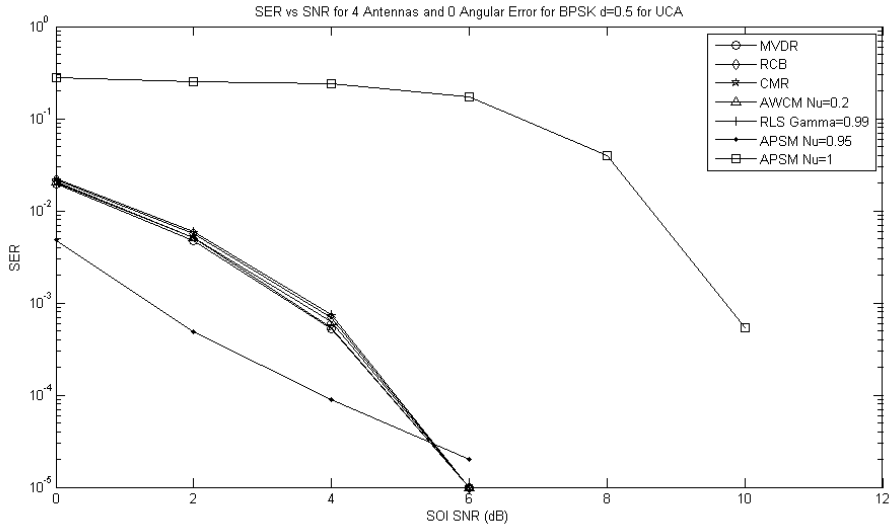


Figure 30. SER vs SOI SNR for UCA of 4 Antennas and BPSK Input Scheme with SOI Angle 30° and Interference Angle 60° with SNR of 10 dB in 0° Angular Mismatch

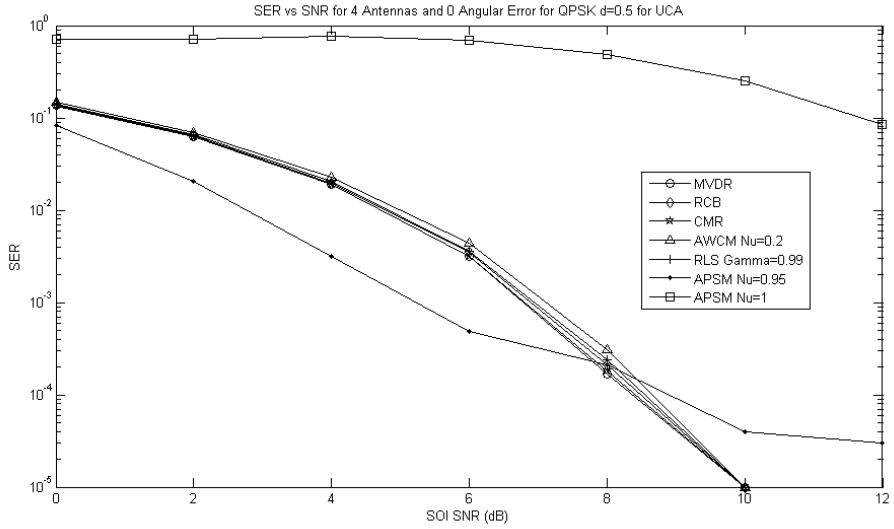


Figure 31. SER vs SOI SNR for UCA of 4 Antennas and QPSK Input Scheme with SOI Angle 30° and Interference Angle 60° with SNR of 10 dB in 0° Angular Mismatch

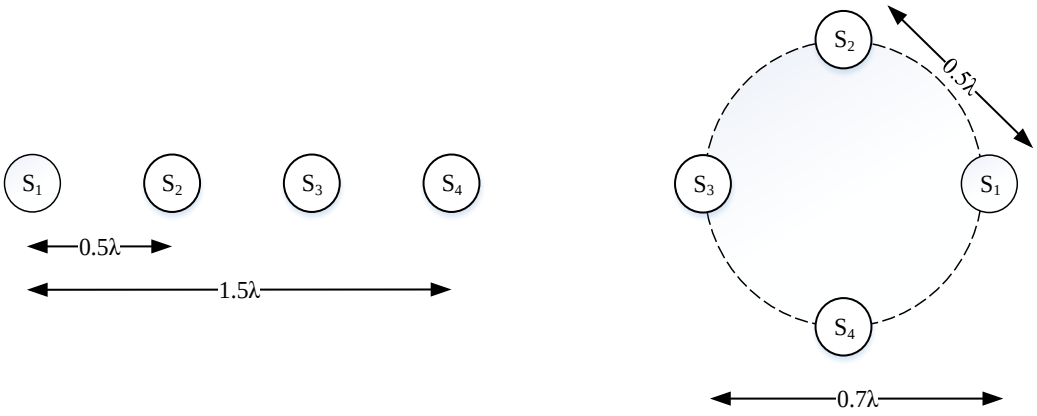


Figure 32. Array Aperture Difference of ULA and UCA for θ=90°

### 4.4.5. Antenna Number

It has been stated in many resources [38] [39] in different disciplines that having a longer array aperture results in higher array gains and narrower main lobes which result in better performance in the sense of maximizing SINR. It is seen in In Eq. (74) and Eq. (75) that beamformer output SINR plays a major role in SOI estimation performance of a beamformer.

In the comparisons made for array antenna number, two signal sources have been placed on 30° and 60° with the one at 30° being the SOI and the other signal is jammer, both with fixed SNR of 10 dB. Array antenna number has been swept in the interval [2, 10]. Some comparisons have been made with no angular mismatch just to observe the sole effect of antenna number. Another set of comparisons have been made where antenna number is swept for different levels of angular mismatch to understand the effect of antenna number in adverse situation. Simulations have been made for BPSK, QPSK and AM signal schemes for both ULA and UCA geometries with d=0.2λ and d=0.5λ.

The performance increasing effect of array antenna number can be observed in Figure 33 and Figure 34 with no angular mismatch case. However, it shall also be noted that ULA performance of algorithms is better than that of UCA based on the array aperture shrinkage of circular geometry as seen in Figure 32.

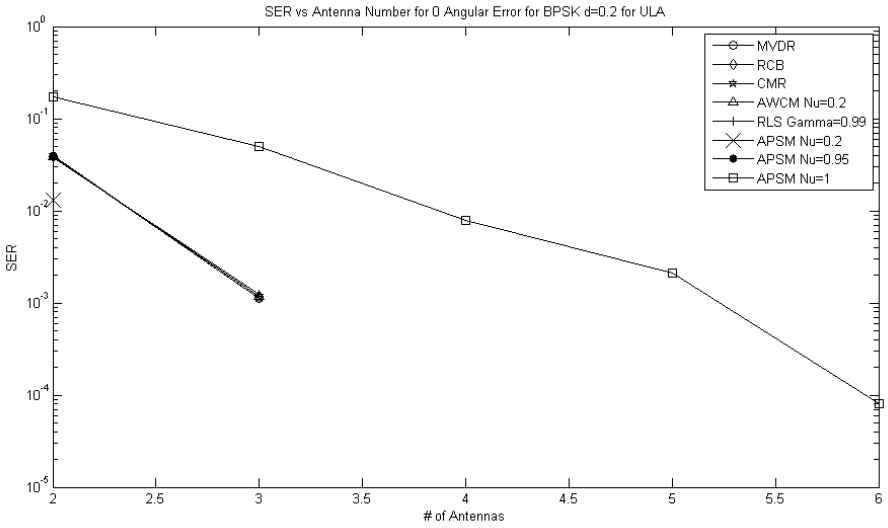


Figure 33. SER vs Number of Antennas for ULA of d=0.2λ and BPSK Input Scheme with SOI Angle 30° and Interference Angle 60° with SNR of 10 dB in 0° Angular Mismatch

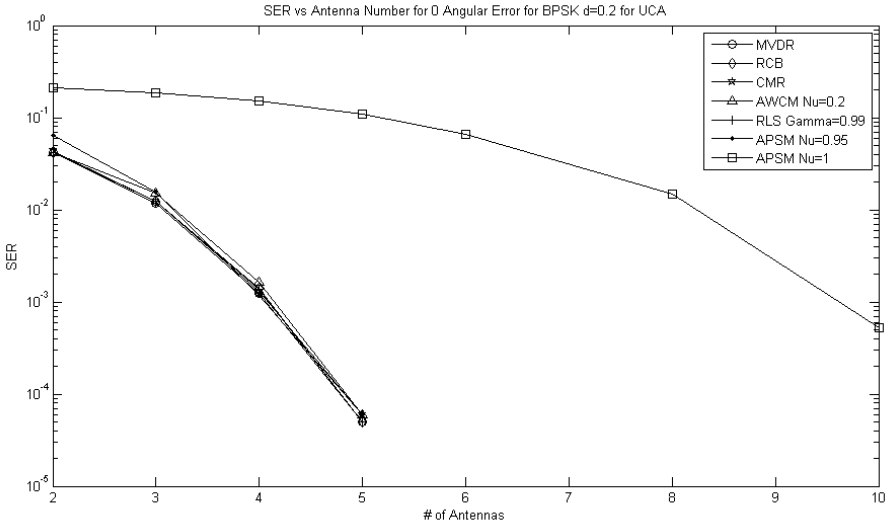


Figure 34. SER vs Number of Antennas for UCA of d=0.2λ and BPSK Input Scheme with SOI Angle 30° and Interference Angle 60° with SNR of 10 dB in 0° Angular Mismatch

Comparing figures between Figure 35 and Figure 41, an interesting case that is not often observed is seen: an increase in the array antenna number may not always lead to a better beamforming performance. It is seen that when some level of angular mismatch is induced in the simulations, the estimation performance tends to improve until an antenna number and then degrades.

As seen between Figure 35 and Figure 38 for BPSK cases, SER levels decrease to a minimum with increasing antenna number in each scenario and then it rapidly increases. It can be observed that the corresponding antenna number for the SER minima is decreasing with the increase in the angular mismatch.

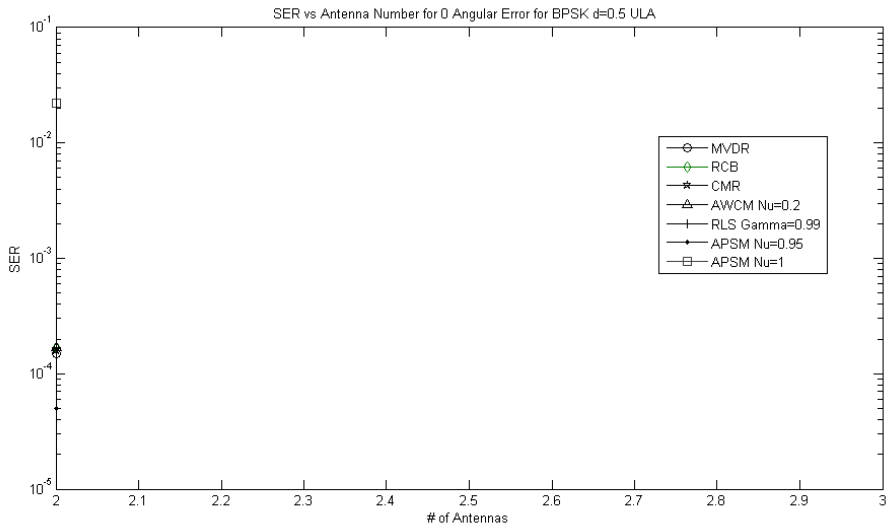


Figure 35. SER vs Number of Antennas for ULA of d=0.5λ and BPSK Input Scheme with SOI Angle 30° and Interference Angle 60° with SNR of 10 dB in 0° Angular Mismatch

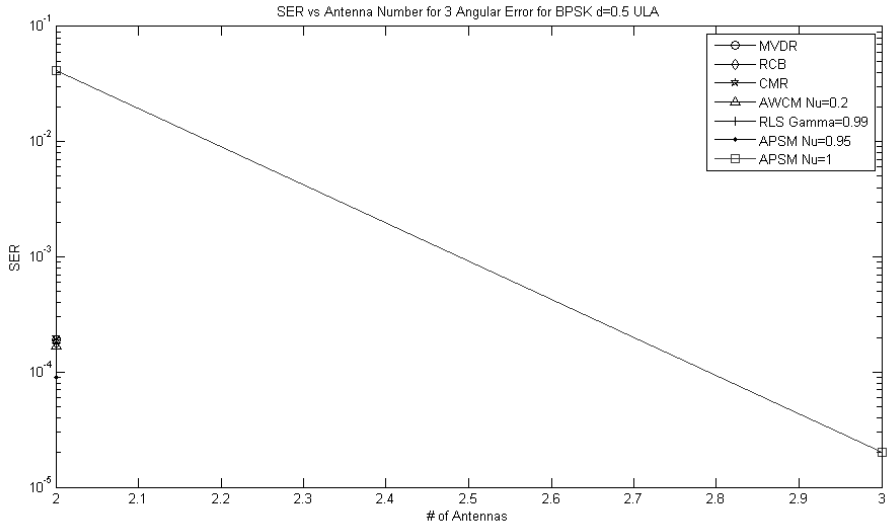


Figure 36. SER vs Number of Antennas for ULA of d=0.5λ and BPSK Input Scheme with SOI Angle 30° and Interference Angle 60° with SNR of 10 dB in 3° Angular Mismatch

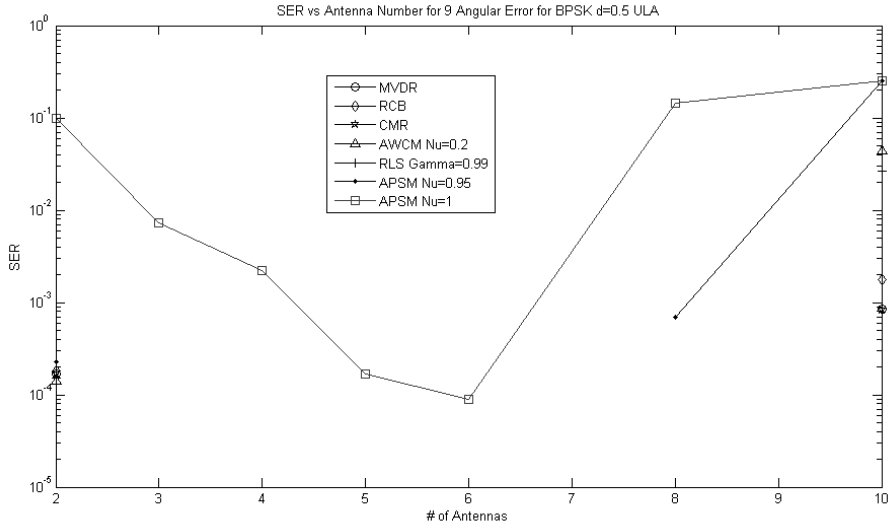


Figure 37. SER vs Number of Antennas for ULA of d=0.5λ and BPSK Input Scheme with SOI Angle 30° and Interference Angle 60° with SNR of 10 dB in 9° Angular Mismatch

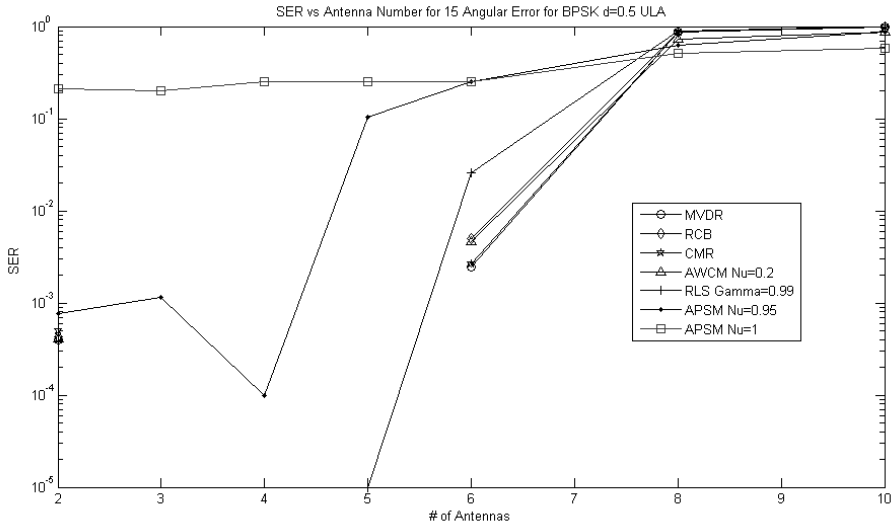


Figure 38. SER vs Number of Antennas for ULA of d=0.5λ and BPSK Input Scheme with SOI Angle 30° and Interference Angle 60° with SNR of 10 dB in 15° Angular Mismatch

Moving from Figure 39 to Figure 41 for AM signal scheme, it can be observed that SINR peak moves to less and less number of antenna as angular mismatch increases.

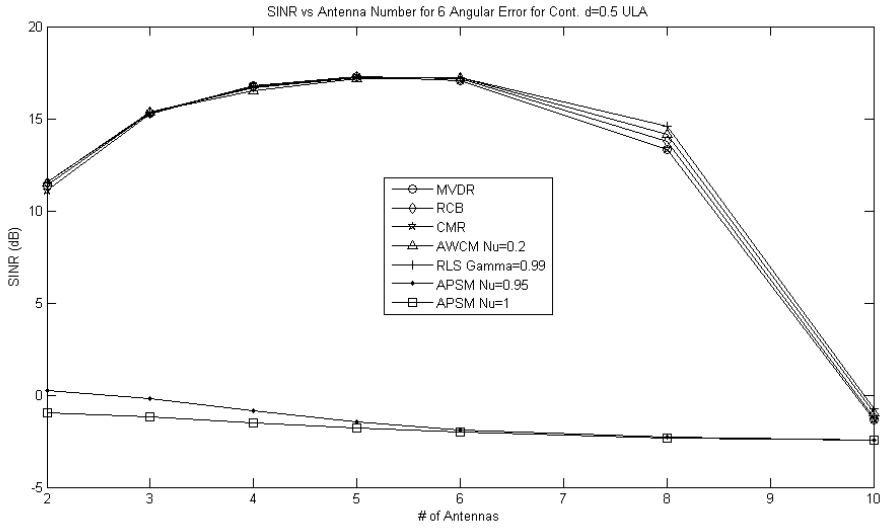


Figure 39. SINR vs Number of Antennas for ULA of d=0.5λ and AM Signal Input Scheme with SOI Angle 30° and Interference Angle 60° with SNR of 10 dB in 6° Angular Mismatch

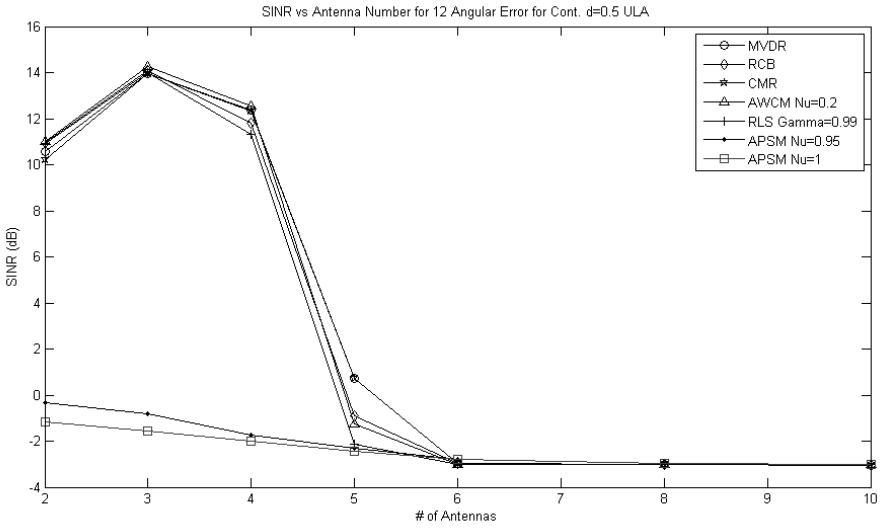


Figure 40. SINR vs Number of Antennas for ULA of d=0.5λ and AM Signal Input Scheme with SOI Angle 30° and Interference Angle 60° with SNR of 10 dB in 12° Angular Mismatch

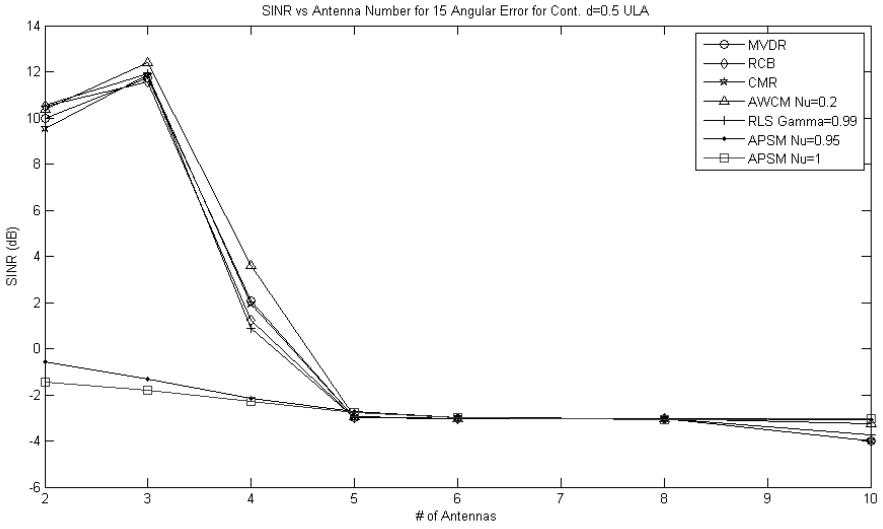


Figure 41. SINR vs Number of Antennas for ULA of d=0.5λ and AM Signal Input Scheme with SOI Angle 30° and Interference Angle 60° with SNR of 10 dB in 15° Angular Mismatch

The reason for the performance loss seen with increasing number of antennas in high angular mismatch is as follows: Having more antenna in the array results in having a narrower main lobe of array beampattern. With a high angular mismatch the SOI

may easily end out of the narrower main lobe of the beamformer, thus it might be received and processed with a performance poorer than ideally possible.

### 4.4.6. Jammer/Interference Number

The primary advantage of APSM algorithm was the assertion in [1] that APSM algorithm would perform well in an array with antenna number less than total number of signals (SOI and interferers). In order to check the validity of this assertion a scenario was generated with one SOI placed at 30° and additional signal sources operating as jammers up to a number of 7, with angular placements of 60°, 15°, 110°, 130°, 125°, 80° and 145° all with 10 dB SNR. SER and SINR performances of algorithms in case of BPSK, QPSK and AM signal schemes on ULA and UCA geometries were observed in a fashion that a new jammer was introduced into the test scenario in each sweep instant.

Figure 42 confirms the assertion for the performance of APSM with a good selection of $\mu$ (0.95) algorithm for BPSK inputs in ULA. Estimation errors start to be observed at a relatively high number of jammers (three jammers present) with very little SER ($\sim 10^{-4}$) with respect to SER ($\sim 10^{-2}$) of MI algorithms and tend to stay satisfactorily below for the rest of the scenario.

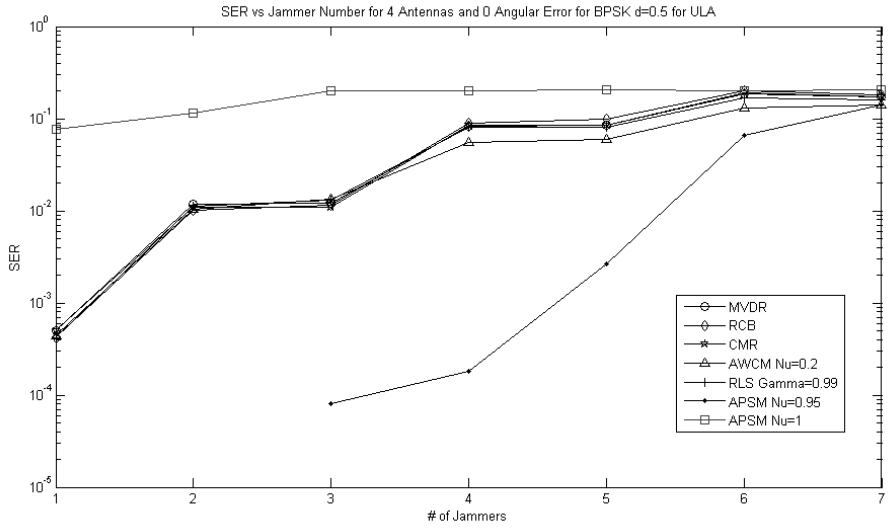


Figure 42. SER vs Number of Jammers for ULA of 4 Antennas and BPSK Input Scheme with SOI Angle 30° and Jammer Angles 60°, 15°, 110°, 130°, 125°, 80°, 145° with SNR of 10 dB in 0° Angular Mismatch

A further look into Figure 42 shows that the design goal of AWCM algorithm has also been reached to a degree for BPSK inputs in ULA of four antennas. The figure reveals that AWCM algorithm performs with known MI algorithms until the scenario that three jammers are present, where MI algorithms' covariance matrices are at full rank with the existence of noise. When the number of jammers start to increase further, that is number of signals (SOI, interferers, noise) impinging on the array exceeds number of antennas in the array, AWCM algorithm out-performs other MI algorithms.

Figure 43 discloses the underlying reason behind the relatively good performance of AWCM algorithm with respect to MVDR algorithm for the case that seven jammers are present in addition to SOI. Heuristic AWCM algorithm places lower levels of real part of beampattern gain at the angular positions of jammers in the scenario. This is a result of the *Intermediate Training Session* mechanism used in AWCM algorithm, which adaptively updates sample covariance matrix with respect to estimation errors.

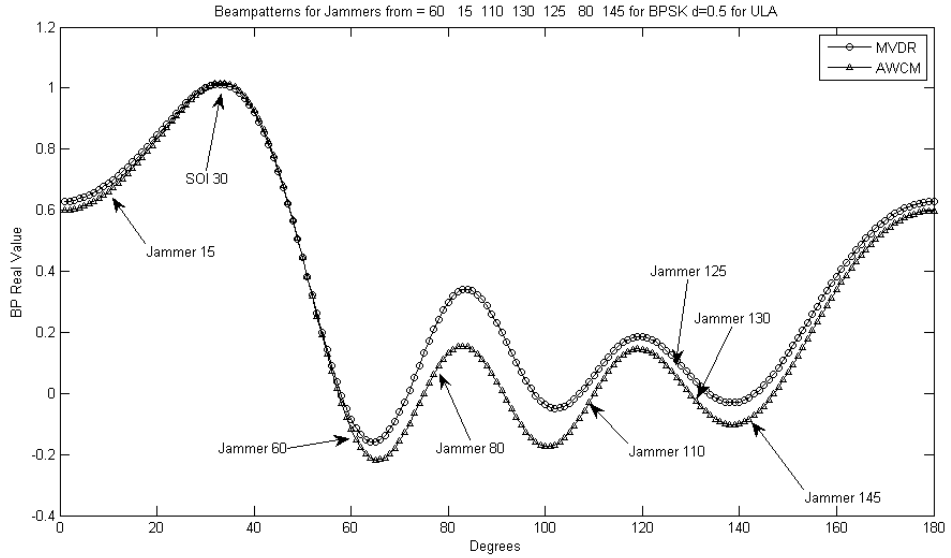


Figure 43. Beampatterns for ULA of 4 Antennas and BPSK Input Scheme with SOI Angle 30° and Jammer Angles 60°, 15°, 110°, 130°, 125°, 80°, 145° with SNR of 10 dB in 0° Angular Mismatch

Figure 44 shows that APSM performs considerably well compared to other algorithms in UCA case. However, it is observed that AWCM algorithm fails to maintain its performance advantage over other MI algorithms in this array geometry.

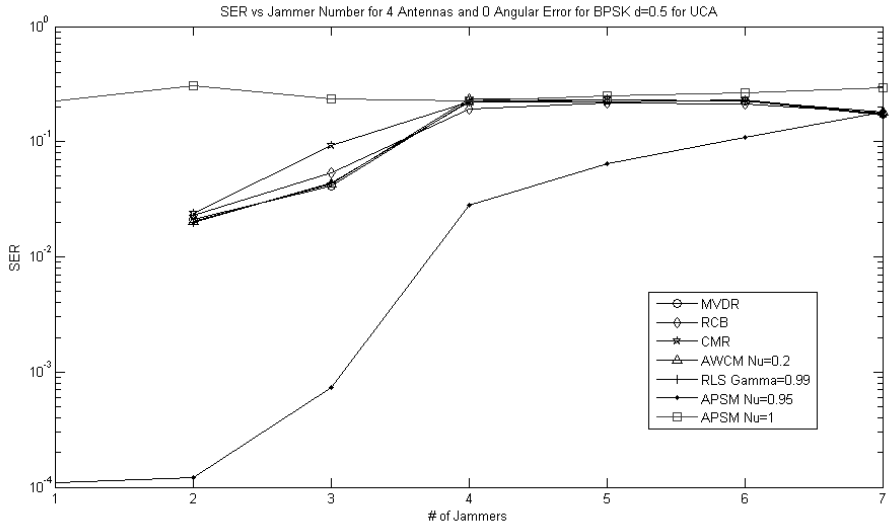


Figure 44. SER vs Number of Jammers for UCA of 4 Antennas and BPSK Input Scheme with SOI Angle 30° and Jammer Angles 60°, 15°, 110°, 130°, 125°, 80°, 145° with SNR of 10 dB in 0° Angular Mismatch

A comparative look on Figure 45 and Figure 43 explains for the performance loss of AWCM with respect to MVDR algorithm in UCA case. Circular array geometry limits null placement ability of the algorithm for suppression of jammers properly. It is observed that UCA geometry places one null on real part of beampattern whereas ULA places three nulls for jammer rejection. Uniform circular arrays' null placement performance has been found to be below that of ULAs' in various sources: [7] [40].

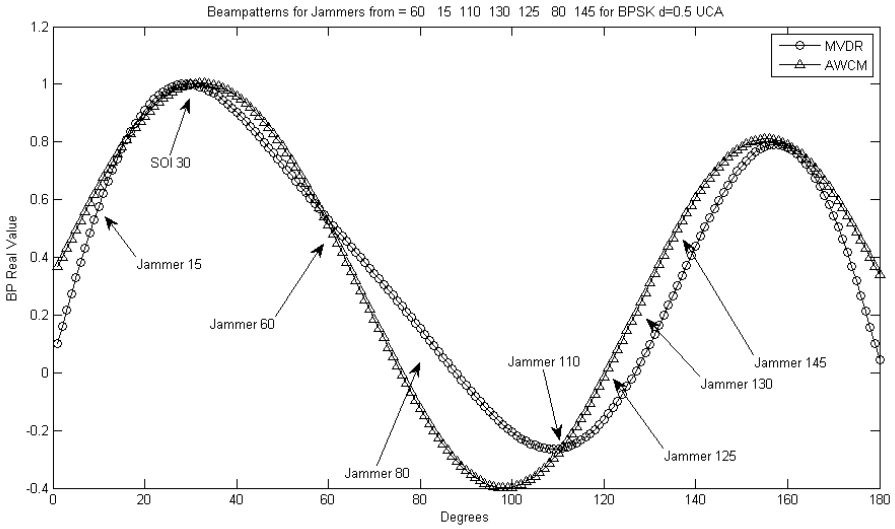


Figure 45. Beampatterns for UCA of 4 Antennas and BPSK Input Scheme with SOI Angle 30° and Jammer Angles 60°, 15°, 110°, 130°, 125°, 80°, 145° with SNR of 10 dB in 0° Angular Mismatch

### 4.4.7. Jammer/Interference – to – SOI Disturbance

As stated in sections Minimum Variance Distortionless Response, Robust Capon Beamformer and Covariance Matrix Reconstruction, self-nulling is a phenomena that occurs when SOI data is received in the training period to generate sample covariance matrix. This situation occurs frequently when interference is in close neighborhood to SOI in angular space. Such a spatial positioning has adverse effects during the testing phase too, in the case that the interferer signal can fall into main lobe of the beamformer.

Jammer-to-SOI disturbance comparisons have been made using two different sets of signal sources: two signal with one SOI placed at 30° and another one (jammer) revolving around SOI in the interval [15°,45°] both with 10 dB SNR. The other case includes four signals with one SOI placed at 30° and another one (jammer) revolving around SOI in the interval [15°,45°] and the other jammers placed at 60° and 90°, all sources with 10 dB SNR.

It can be seen in Figure 46 and Figure 47 that APSM (μ=0.95) algorithm performs very well with SOI in close vicinity of jammer at any number of jammers due to the existence of training data which bears SOI data in addition with jammer distortion. APSM (μ=1) performs badly because it has a solution basis which includes only SOI data that results in a failure during the operational phase.

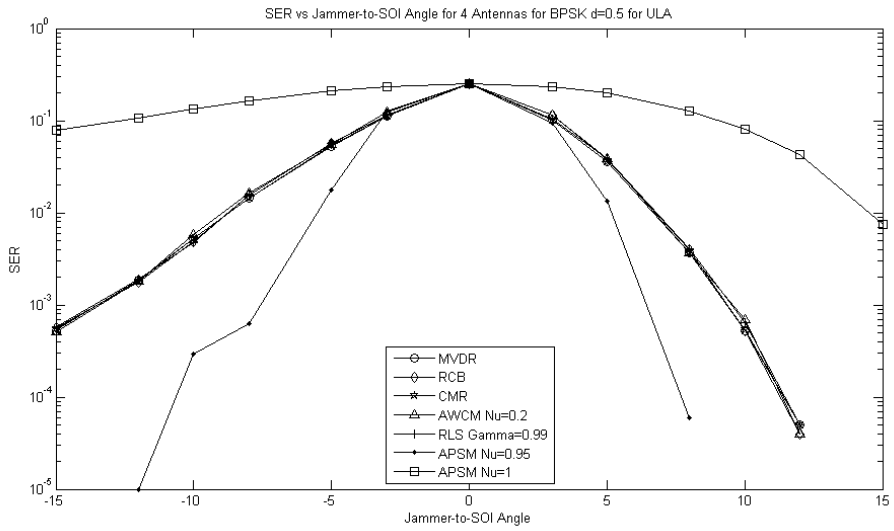


Figure 46. SER vs Jammer-to-SOI Angle for ULA of 4 Antennas and BPSK Input Scheme with SOI Angle 30° and One Jammer with SNR of 10 dB in 0° Angular Mismatch

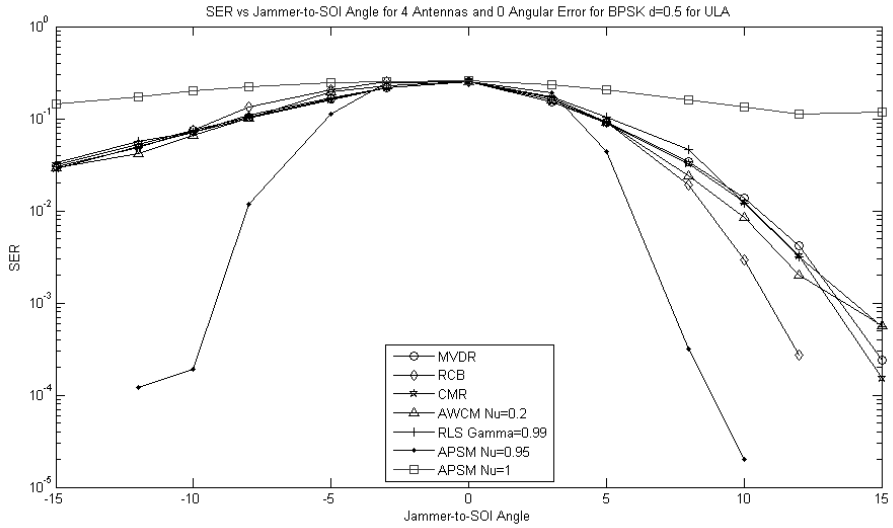


Figure 47. SER vs Jammer-to-SOI Angle for ULA of 4 Antennas and BPSK Input Scheme with SOI Angle 30° and Three Jammers of angles 30°, 60°, 90° with SNR of 10 dB in 0° Angular Mismatch

A performance loss in the form of an elevated SER charts for all algorithms can be observed when Figure 48 and Figure 49 are compared with their ULA counterparts: Figure 46 and Figure 47.

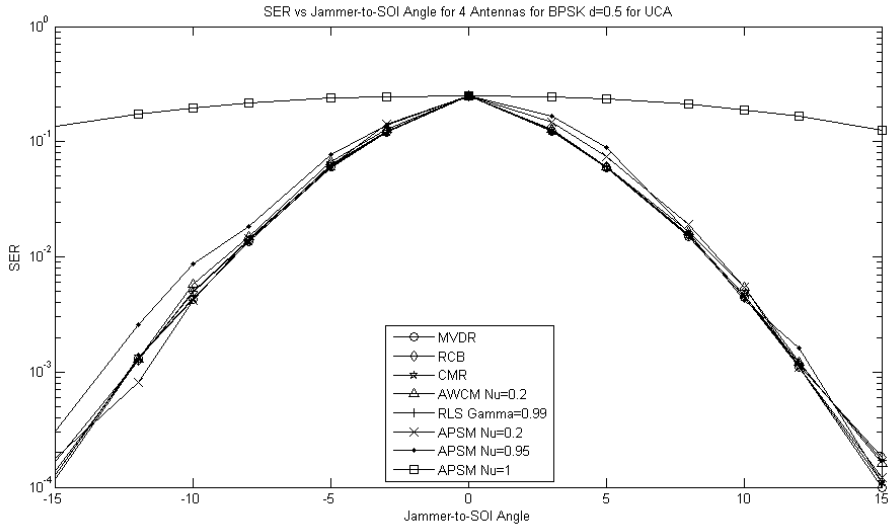


Figure 48. SER vs Jammer-to-SOI Angle for UCA of 4 Antennas and BPSK Input Scheme with SOI Angle 30° and One Jammer with SNR of 10 dB in 0° Angular Mismatch

A quick observation on Figure 48 and Figure 49 shows that angular sweep performance is symmetrical around SOI, a property that is not held by ULAs due to their difference in performance between array boresight and endfire.

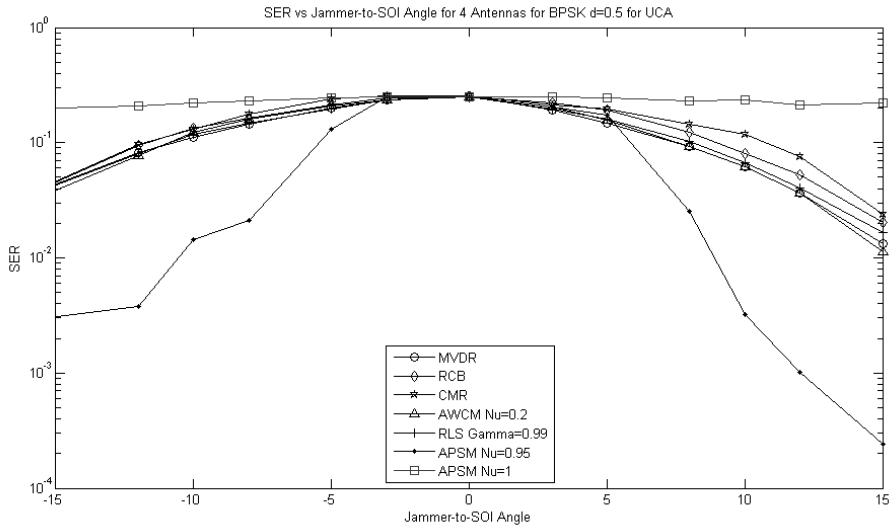


Figure 49. SER vs Jammer-to-SOI Angle for UCA of 4 Antennas and BPSK Input Scheme with SOI Angle 30° and Three Jammers of angles 30°, 60°, 90° with SNR of 10 dB in 0° Angular Mismatch

A comparison between Figure 48 and Figure 50 confirms that the relation between Eq. (74) and Eq. (75) that the QPSK estimation performance is worse than that of BPSK performance still holds for this case.

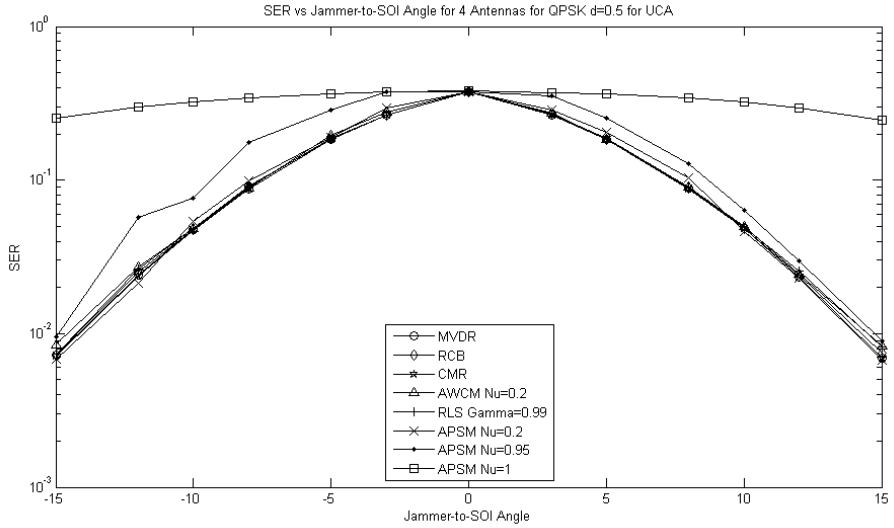


Figure 50. SER vs Jammer-to-SOI Angle for UCA of 4 Antennas and QPSK Input Scheme with SOI Angle 30° and One Jammer with SNR of 10 dB in 0° Angular Mismatch

Looking at Figure 51 and comparing it with Figure 25 and Figure 39 shows how adverse it is to have a jammer close to SOI in terms of worsening the estimation performance of MI algorithms in analog modulated signal case. In Figure 25, where a single jammer is 105° away from SOI and in Figure 39, where the jammer is 30° away from the SOI, SINR values are almost 16 dB. In a scenario, that jammer is 15° away from the SOI; SINR value becomes almost 7 dB.

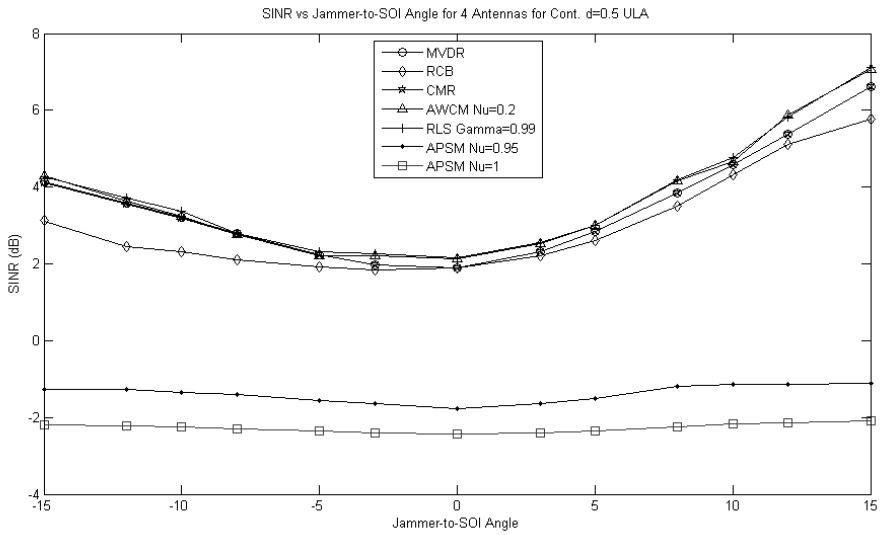


Figure 51. SINR vs Jammer-to-SOI Angle for ULA of 4 Antennas and Analog Modulated Input Scheme with SOI Angle 30° and One Jammer with SNR of 10 dB in 0° Angular Mismatch

### 4.4.8. SOI-to-Jammer/Interference Power Ratio

SOI-to-Jammer Power Ratio comparisons have been made using two signal with one SOI placed at 30° with SNR in the interval [-30dB, 10 dB] and another one (jammer) at 60° with 10 dB fixed SNR.

Figure 52 and Figure 53 show that MI algorithms can perfectly estimate SOI symbols even at low SNR levels due to their beampattern narrow enough, unrelated to SINR, to isolate SOI from jammer and noise. APSM algorithms relatively fail depending the magnitude of μ parameter because preamble basis matrix does not reflect the operational scenario situation where the solution basis is ideal and having a relatively large μ prevents the algorithm to train itself in the training session. Decreasing the value of μ helps the algorithm to recover and perform with satisfactory results quickly

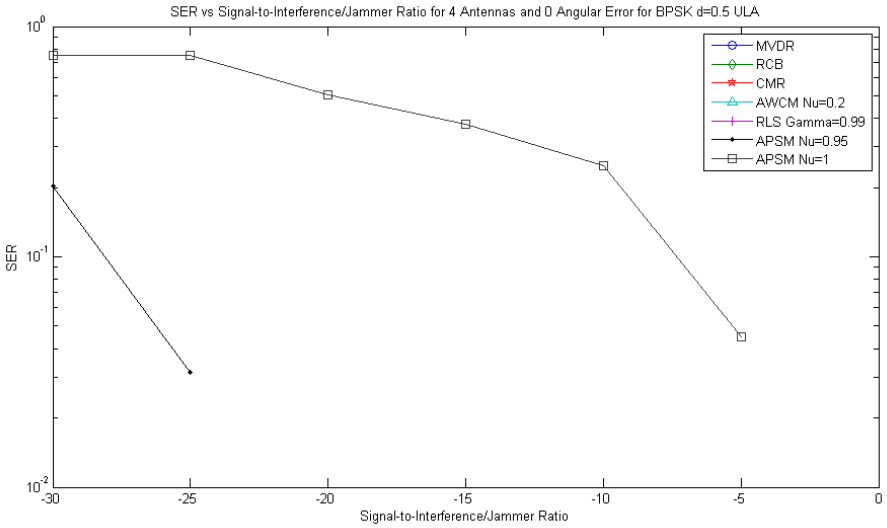


Figure 52. SER vs SOI-to-Jammer Power Ratio for ULA of 4 Antennas and BPSK Input Scheme with SOI Angle 30° and One Jammer of angle 60° with SNR of 10 dB in 0° Angular Mismatch

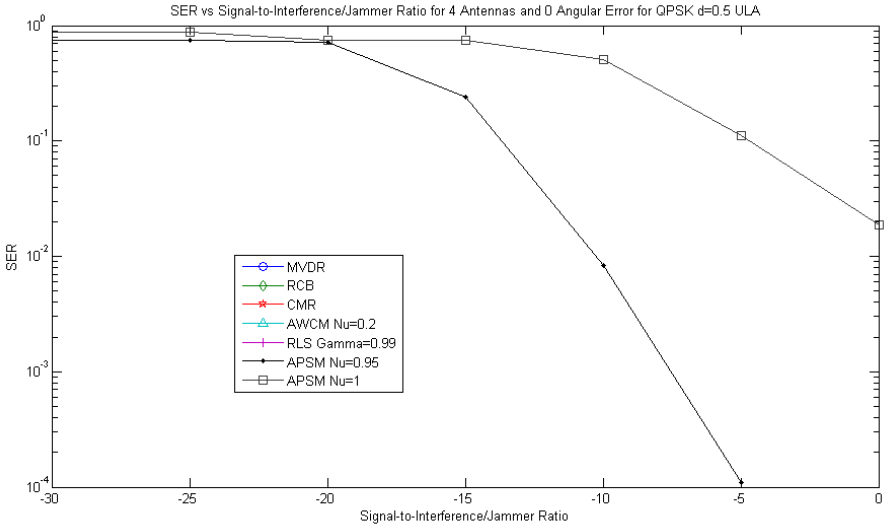


Figure 53. SER vs SOI-to-Jammer Power Ratio for ULA of 4 Antennas and QPSK Input Scheme with SOI Angle 30° and One Jammer of angle 60° with SNR of 10 dB in 0° Angular Mismatch

### 4.4.9. Carrier Frequency

Having an array operating at a fixed or nearly fixed carrier frequency makes the array aperture constant relative to the wavelength of the carrier. Operating an array at lower frequencies than what it is designed for, relatively shrinks the array aperture, which results in performance loss compared to higher carrier frequency case.

In the carrier frequency comparisons, two signal sources have been used in the regular scenario placements: 30° for SOI and 60° for jammer both with 10 dB SNR. The signals impinged on ULA and UCA of four antennas with fixed inter-elemental spacing $d=0.5\lambda_{min}$ where $\lambda_{min}$ corresponds to wavelength at maximum frequency used in the simulations. The carrier frequency has been increased five-fold with respect to minimum carrier frequency. Fixing the inter-elemental spacing, we get the following relative aperture length for different cases:

Table 1. Carrier Frequency vs. Aperture Length for Fixed Inter-elemental Spacing

| Carrier Frequency | Aperture Length wrt. carrier wavelength for each case |
|---|---|
| $f_{min}$ | $0.3\lambda$ |
| $2f_{min}$ | $0.6\lambda$ |
| $4f_{min}$ | $1.2\lambda$ |
| $5f_{min}$ | $1.5\lambda$ |

It must be noted that the carrier frequency is swept in the simulations not the frequency of the SOI. That is, algorithms' narrowband beamforming performances on a wide range of carrier frequencies have been compared. As mentioned in previous sections and comparison results, having a short array aperture length with respect to carrier wavelength results in worse performance than best that can be attainable. Figure 54 and Figure 55 depict this theory for ULA and UCA geometries.

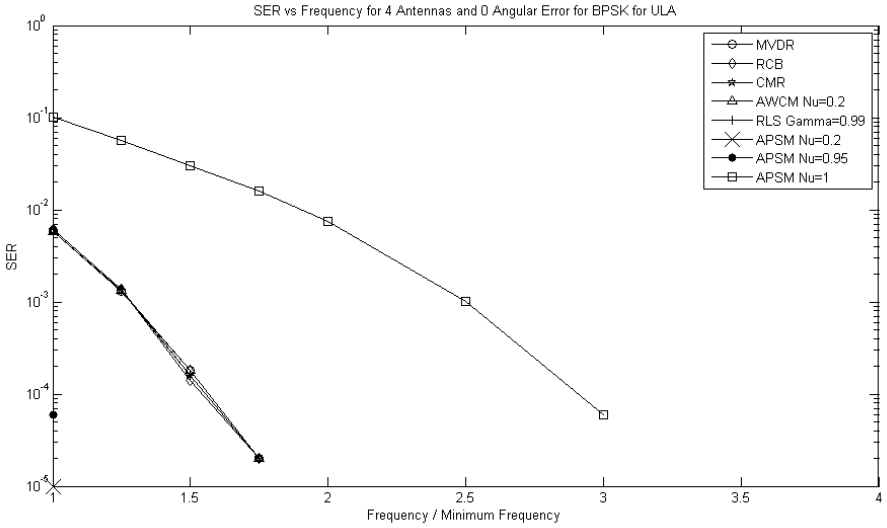


Figure 54. SER vs Carrier Frequency for ULA of 4 Antennas with fixed inter-elemental spacing $d=0.5\lambda_{min}$ and BPSK Input Scheme with SOI Angle 30° and One Jammer of angle 60° with SNR of 10 dB in 0° Angular Mismatch

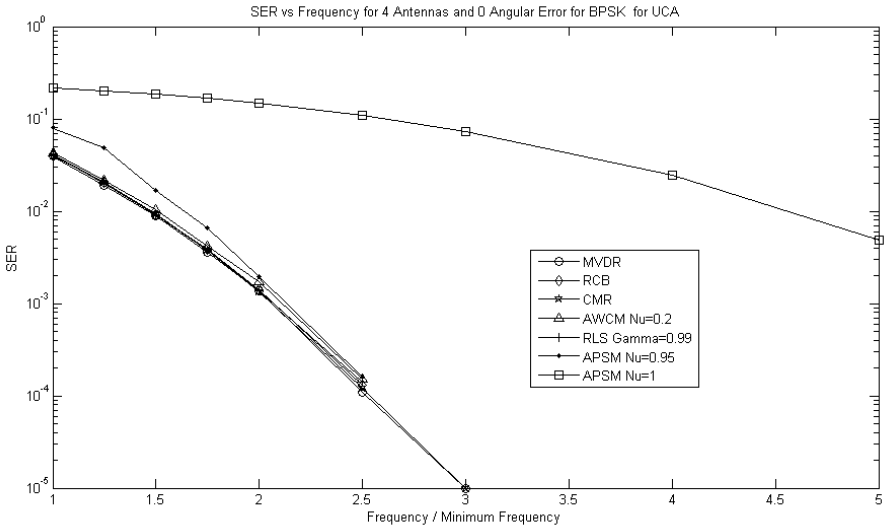


Figure 55. SER vs Carrier Frequency for UCA of 4 Antennas with fixed inter-elemental spacing $d=0.5\lambda_{min}$ and BPSK Input Scheme with SOI Angle 30° and One Jammer of angle 60° with SNR of 10 dB in 0° Angular Mismatch

### 4.4.10. Computational Complexity

In addition to performance comparisons of algorithms with the results presented in the previous sections of this chapter, computational complexity simulations for all the algorithms have been made using a personal computer with Intel i-7 processor running at 3.4 GHz processing speed and 8 GB RAM. A single core has been used in the simulations to make an objective conclusion on the algorithms instead of parallel loops being run.

Computation time for beamformer weight vector computation or look-up table generation time called “training session” using a 10 element preamble basis, 200 snapshots with and testing session using 10000 symbols for all algorithms can be seen in Table 2. APSM algorithm computation time has been generated for three different values of μ parameter to comprehend the effect of including more look-up table elements in both training and test sessions.

Table 2. Computation Time for Algorithms

| **Method** | **Training Session Time** | **Testing Session Time** |
|---|---|---|
| MVDR | 4.40698e-05 | 4.78978e-05 |
| RCB | 1.4715 e-03 | 4.75604e-05 |
| CMR | 1.04952e-03 | 4.88993e-05 |
| AWCM | 2.59091e-02 | 4.73901e-05 |
| RLS | 5.07622e-03 | 4.79938e-05 |
| APSM μ=1 | 0.569717 | 0.842854 |
| APSM μ=0.95 | 0.520974 | 2.15321 |
| APSM μ=0.3 | 1.18376 | 20.5652 |

Comparing computation time of algorithms, MVDR algorithm has the best time for training session as it only has one-step computation of beamformer weight vector. However, RCB and CMR methods have slightly higher computational complexity

for training session. This is because RCB algorithm includes index search and norm operations for computing a decent diagonal loading factor for robustness and CMR algorithm has a Beampattern minima search. Iterative-based adaptive algorithms such as AWCM and RLS methods have the most computational complexity of training period for Matrix Inversion (MI) algorithms. However, due to lack of covariance matrix inversion in RLS method, its computational complexity it much smaller than that of AWCM. Testing times for all MI based algorithms however, are almost same because after having a beamformer weight vector ready the only thing to compute an output is to multiply it with beamformer input.

APSM methods have the longest computation time of algorithms, increasing with decreasing μ parameter. Having smaller μ means more training samples are used to form a look-up table in training session. Using a more crowded look-up table means more computation in the testing phase even with same number of test samples.

Analyzing APSM algorithm steps, and computation complexity for most frequently and computationally demanding used functions such as taking norms and matrix inversions, the following approximations have been made for computation times for both training and testing phases of APSM algorithms depending on the value of μ used:

$$\mathrm{TIME}_{\text{APSM-Training}} \propto \begin{Bmatrix} 10N_{tra}^2, \text{ for } \mu<0.6 \\ 2N_{tra}N_{pre}+4N_{tra}^2, \text{ for } \mu\rightarrow 1 \end{Bmatrix} \tag{76}$$

$$\mathrm{TIME}_{\text{APSM-Testing}} \propto \begin{Bmatrix} 2N_{tra}N_{test}, \text{ for } \mu<0.6 \\ N_{pre}N_{test}, \text{ for } \mu\rightarrow 1 \end{Bmatrix} \tag{77}$$

where $N_{pre}$ denotes number of preamble basis elements, $N_{tra}$ denotes number of training snapshots and $N_{test}$ denotes number of test snapshots.

### 4.4.11. Summary of Algorithm Evaluations

In the previous sections of this chapter, several performance evaluations of beamformer estimators have been made for different scenario parameters. Classical beamformers, covered in chapter 2, are compared with APSM and AWCM algorithms with appropriate μ parameters for each algorithm and overall evaluation summary can be seen in Table 3.

Table 3. Beamformers' Estimator Performance Parameter Dependence

| Parameter | Classical Beamformers | APSM | AWCM |
|---|---|---|---|
| AM I/O scheme | Low | High | Low |
| Angular Mismatch | Low | Low | Low |
| Number of Training Snapshots | Low | Medium | Low |
| Input SNR | Medium | Medium | Medium |
| Antenna Number | Medium | Medium | Medium |
| Jammer Number | Very high | Low | High |
| Jammer-to-SOI Disturbance | High | Medium | High |
| SOI-to-Jammer Power Ratio | Low | Medium | Low |
| Carrier Frequency | High | Low | High |
| Computational Complexity | Very low | Very high | Very low |

# CHAPTER 5

# CONCLUSION AND FUTURE WORK

In this master thesis, a non-linear beamformer has been implemented along with other classical beamformers in order to make an extensive performance evaluation between them. This method and the mechanisms it uses have been summarized in [1] [27]. However, performance of this method has been briefly presented for a static scenario simulation. This approach has not fully disclosed the mechanisms behind the algorithm and its performance dependence on the scenario parameters. Furthermore, the lack of understanding the algorithm's performance in detail may prevent its application in an efficient way. To overcome this and understand the algorithm in detail, simulations have been made for varying scenarios with different scenario parameters. APSM algorithm's performance, on changing scenario parameters, has been compared to well-known performances of classical beamformers to discover its advantages and disadvantages clearly.

## 5.1. Summary of Work

Different known and widely used adaptive beamforming algorithms [2] [3] [4] [5], as well as APSM algorithm using projections in Reproducing Kernel Hilbert Space [1] have been implemented in uniform linear and circular arrays for inputs of BPSK, QPSK and AM signals; then used as signal-of-interest estimators. After understanding the mechanics of all algorithms implemented, a novel method combining advantages of these algorithms have been proposed and implemented. Estimation performance of these algorithms have been compared in MATLAB with varying scenario parameters such as angular mismatch, input SNR, Signal-to-Interference-plus-Jammer Ratio, training data number, array antenna number,

interferer signal source number, jammer-to-SOI angular neighborhood and carrier frequency. Additional comparison of computational complexity has been made for the algorithms.

## 5.2. Conclusions on APSM Algorithm and AWCM Algorithm

APSM algorithm works in Hilbert Space. That is, array signals are turned into kernel functions in Hilbert Space. In contrast to classical beamformers that use a weight vector, APSM algorithm uses a basis ($\boldsymbol{\beta}$) and a coefficient vector ($\boldsymbol{\gamma}_{APSM}$) for the elements of that basis. $\boldsymbol{\beta}$ is initiated by a preamble basis that depends on a priori DOA of SOI. During a training period, when signal sources transmit on the array with known signal values, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}_{APSM}$ are iteratively populated using the kernel that belong to each new training data. In the operational phase, array signals for each instant are similarly turned into a kernel without the knowledge of SOI value for that instant. This kernel is projected onto each kernel of $\boldsymbol{\beta}$ and the projections are summed with coefficient vector $\boldsymbol{\gamma}_{APSM.}$ The final point after projections is the beamformer output for that particular instant. Depending on the input signal scheme (PSK or AM), beamformer output is either mapped to a symbol value or used as is as the final output of the estimator.

Using a basis that stores past data in a weighted manner characterizes APSM algorithm. It enables the beamformer to work with less number of antennas with respect to signal sources, a case that leads the classical beamformers to fail dramatically. Using a basis and employing it in a look-up table fashion makes the algorithm very efficient in handling signals in a wider frequency band or separating closely spaced signal sources. However, having a basis set, elements of which are used in computationally complex mathematical operations, make APSM algorithm very slow with respect to classical beamforming algorithms. Another shortcoming of APSM algorithm is that it can only work with signal types that has a finite number of elements such as BPSK or QPSK. In the case of infinite element signal types (AM signal), the algorithm fails because the basis ($\boldsymbol{\beta}$) must have enough elements

belonging to each possible signal value. That is, an infinite length basis is necessary to deal with AM signals. In order to implement APSM algorithm, modulation type must be known before operational phase. This information is necessary to build the preamble basis before training period.

Training period is very important in APSM algorithm because the basis is initiated only with a priori SOI DOA information and algorithm would fail in the existence of jammers. During the training period, the basis is populated with new elements and the algorithm "learns" to deal with jammers iteratively. However, a long training period results in a larger basis, which would lead to higher computational burden in the operational phase.

To sum up, the advantages of APSM algorithm over classical beamformers are improved estimation performance in the following cases:

- Number of signals are more than number of array antennas,
- Jammers are closely spaced to SOI,
- Working in a wide carrier frequency range.

However, APSM has three important disadvantages:

- Ability to work in finite element signal types,
- Necessity to know SOI modulation type to generate a preamble basis,
- Large computational complexity.

In the search for a possible synthesis algorithm, which would enhance classical beamformers performance via "look-up" method used in APSM algorithm, Adaptive Weighted Covariance Matrix algorithm has been proposed. This algorithm is implemented as a linear beamformer, which uses a weight vector that is a

modification of Eq. (19). It starts with an initial weight vector generated by a sample interference-plus-noise covariance matrix belonging to a portion of training samples. During the rest of the training period, beamformer works as an estimator and it updates covariance matrix iteratively using the estimation error with the advantage of having full knowledge of training data. This "test-in-training phase" enables the algorithm to perform slightly better than classical beamformers in the case that there are more signal sources than array antennas for ULAs. This performance enhancement is not shadowed by any extra computation complexity over classical matrix inverse (MI) algorithms because AWCM is an MI algorithm.

### 5.3. Comparison of Methods

Although being statistically optimum, MVDR algorithm [2] suffers from the lack of true covariance matrix in practical situations; to overcome this deficiency adaptive algorithms with covariance matrix diagonal loading [3], covariance matrix reconstruction [4] or adaptive update [5] of covariance matrix exist. The observations of this thesis lead to the conclusion that these methods do not substantially improve estimation performance of MI algorithm (with driving equation seen in Eq. (19)) because these algorithms perform considerably well in the task of estimation in simple cases.

It has been observed in section 4.4.2 that all the algorithms are robust against angular mismatch; it has been observed in section 4.4.3 that all algorithms perform well with scarce number of training data. Results of section 4.4.4 show that increasing input SNR positively affect estimation for all algorithms as expected. It shall also be noted that a relatively low SNR (~3 dB) is enough for algorithms to reach a satisfactory ($\sim 10^{-4}$) symbol error rate in a ULA designed to resolve BPSK symbols. The intuitive result that *increasing number of antennas in the array improves array performance* has been considered in section 4.4.5 for small angular error. However, it has interestingly been observed that increasing number of antennas yields in worsening

estimation performance because SOI falls out of the narrow beam of the beamformer because of angular mismatch.

In section 4.4.6, it is observed that APSM algorithm out-performs all other algorithms in the case that fewer antennas exist in the array compared to total number of signal sources. Projection method that lead to a memory-full algorithm brings this advantage. AWCM algorithm, which was designed to deal with high number of jammers compared to array antennas, perform better than other classical beamformers.

In section 4.4.7, it has been seen that APSM algorithm, due to its adaptively learning mechanism during the training session, out-performs other algorithms in the case that SOI being placed close to jammers. Section 4.4.8 shows a disadvantage of APSM algorithm to MI algorithms that it need SOI not to be very weak in power with respect to Interference-plus-Noise. This shortcoming is a result of a preamble solution basis used as a part of final solution set which includes only SOI in its generation. This means comparing a high SNR data with low SNR data, this approach results in failure. However, this shortcoming can be overcome by the inclusion of training data to the solution set.

A disadvantage of MI algorithms over APSM has been witnessed in section 4.4.9, that estimation performance of a fixed elemental-spacing array is dependent on carrier frequency, this happens because of a shrinkage of array aperture with respect to carrier wavelength in low frequencies. APSM with a decent training session performs well in different carrier frequencies, as its performance is free of its array aperture size. It can be concluded that APSM algorithm would perform better than classical beamformers when array receives signals over a wider range of carrier frequency.

APSM algorithm's biggest disadvantage has been revealed in section 4.4.10, is computational complexity. It is observed that use of kernels, which depends on norm operations in loops, brings extreme computational load to APSM over MI algorithms. Other algorithms bear acceptable computational complexity with respect to APSM.

In conclusion, it can be stated that APSM has one significant advantage over MI algorithms: performing well in arrays with antennas less than total number of signal sources. However, APSM algorithm is computationally expensive. It is suggested that for SOI symbol classification purposes of beamformers, APSM algorithm with an appropriate selection of μ shall be implemented as a complementing algorithm, to be used when necessary with another MI algorithm running in a default mode, in order to exploit its advantage with its computational burden. Furthermore, it can be suggested to use AWCM algorithm, generated in the study leading to this thesis, as a default estimator in the above-mentioned setting.

## 5.4. Future Work

As stated repetitively before, AWCM algorithm is an intuitive algorithm, which lacks its mathematical reasoning of dynamics for its good performance. A future work dedicated to build an adaptive beamforming algorithm, which combines optimality of covariance matrix estimation as well as robustness to ambiguous scenario assumptions in a minimal antenna array, could result in improved performance of existing algorithms in the least computational complexity.

# REFERENCES


[1] S. THEODORIDIS, K. SLAVAKIS and I. YAMADA, "Adaptive Learning in a World of Projections," *IEEE Signal Processing Magazine,* pp. 97-120, 2011.

[2] J. CAPON, "High resolution frequency-wavenumber spectrum analysis," *Proceedings of the IEEE,* vol. 57, pp. 1408-1418, 1969.

[3] "Robust Capon Beamforming," in *Robust Adaptive Beamforming*.

[4] R. MALLIPEDDI, J. P. LIE, S. G. RAZUL, P. N. SUGANTHAN and C. M. S. SEE, "Robust Adaptive Beamforming Based on Covariance Matrix Reconstruction for Look Direction Mismatch," *Progress in Electromagnetic Research Letters, Vol. 25,* pp. 37-46, 2011.

[5] "LabVIEW Adaptive Filter Toolkit 1.0," National Instruments, 2008.

[6] T. E. TUNCER, "Sensor Array Signal Processing Lecture Notes," METU, EEE, Ankara, 2012.

[7] Q. ZHENG, "M. Sc. Thesis: Simulation of Adaptive Circular Array for CDMA Systems," Concordia University, 2002.

[8] J. LI and P. STOICA, Robust Adaptive Beamforming, Wiley & Sons.

[9] T. LO, H. LEUNG and J. LITVA, "Nonlinear beamforming," *Electronics Letters*, vol. 27, no. ISSN 0013-5194, pp. 350-352, 1991.

[10] A. R. MOGHIMI and R. M. STERN, "An analysis of binatural spectro-temporal masking as nonlinear beamforming," Carnegie Mellon University, Pittsburgh.

[11] S. CHEN, L. HANZO and A. WOLFGANG, "Kernel-based nonlinear beamforming construction using orthogonal forward selection with fisher ratio class seperability measure," *IEEE Signal Processing Letters,* 2003.

[12] A. KOÇ, "Spectral Estimation Lecture Notes," METU, EEE, Ankara, 2013.

[13] H. Van TREES, Optimum Array Processing, John Wiley & Sons, Inc., 2002.

[14] R. PLACKETT, "Some Theorems in Least Squares," *Biometrika, 37,* pp. 149-157, 1950.

[15] R. KALMAN, "A New Approach to Linear Filtering and Prediction Problems," *Trans. ASME, J. Basic Eng., 82,* p. 35–45, 1960.

[16] M. H. HAYES, "Recursive Least Squares," in *Statistical Digital Signal Processing and Modeling*.

[17] J. B. HIRIART-URRUTY and C. LEMARECHAL, Convex Analysis and Minimization Algorithms, Berlin: Springer-Verlag, 1993.

[18] A. H. SAYED, Fundamentals of Adaptive Filtering, Wiley, 2003.

[19] H. BAUSCHKE and J. BORWEIN, "On projection algorithms for solving convex feasibility problems," *SIAM Rev.,* vol. 38, no. 3, pp. 367-426, 1996.

[20] H. STARK and Y. YANG, Vector Space Projections, New York: Wiley, 1998.

[21] S. BOYD and L. VANDENBERGHE, Convex optimization, Cambridge: Cambridge University Press, 2004.

[22] H. BAUSCHKE, "Ph.D. Dissertation:Projection algorithms and monotone operators," Simon Fraiser University, Mathematics Department, Canada, 1996.

[23] G. PIERRA, "Decomposition through formalization in a product space," *Math. Program,* vol. 28, pp. 96-115, 1984.

[24] Y. CENSOR and S. ZENIOS, Paralel Optimization:Theory, Algorithms and Applications, London: Oxford University Press, 1997.

[25] P. COMBETTES, "Convex set theoritic image recovery by extrapolated iterations of paralel subgradient projections," *IEEE Trans. Image Processing,* vol. 6, no. 4, pp. 493-506, 1997.

[26] D. BUTNARIU, Y. CENSOR and S. REICH, "Inherently Paralel Algorithms for Feasibility and Optimization and Their Applications," *Amsterdam Elsevier,* 2001.

[27] I. YAMADA, K. SLAVAKIS and K. YAMADA, "An Efficient robust adaptive filtering algorithm based on parallel subgradient projection techniques," *IEEE Trans. Signal Processing*, vol. 50, no. 5, pp. 1091-1101, 2002.

[28] GATSBY UNIT, "Introduction to RKHS, Advanced Topics in Machine Learning," University College London, 2014.

[29] T. M. COVER, "Geometrical and statistical properties os systems of linear inequalities with applications in pattern recognition," *IEEE Transactions Electron. Comput.,* vol. 14, pp. 326-334, 1965.

[30] A. SMOLA and B. SCHOLKOPF, Learning with Kernels, Cambridge: MIT Press, 2001.

[31] S. THEODORIDIS and K. KOUTROUMBAS, Pattern Recognition, San Diego, CA:Academic, 2008.

[32] G. WAHBA and G. KIMELDORF, "Some results on Chebychevian Spline Functions," *Mathematical Applications,* vol. 33, no. 1, pp. 82-95, 1971.

[33] K. SLAVAKIS and I. YAMADA, "Robust wideband beamforming by hybrid steepest descent method," *IEEE Trans. Signal Processing*, vol. 55, no. 9, pp. 4511-4522, 2007.

[34] B. HAMERS, J. SUYKENS and B. DE MOOR, "Compactly supported RBF kernels for sparsifying the Gram matrix in LS-SVM regression models," *ICANN Proceedings*, pp. 720-726, 2002.

[35] Y. CHEN and J. LEE, "Finite Data Performance Analysis of MVDR Antenna Array Beamformers with Diagonal Loading," *Progress In Electromagnetics Research*, vol. 134, pp. 475-507, 2013.

[36] A. GERSHMAN, Y. RONG and Y. ELDAR, "Performance Tradeoffs among Beamforming Approaches," *IEEE Sensor Array and Multichannel Processing Workshop*, pp. 26-30, 2006.

[37] S. ALOUNI, Digital Communications on Fading Channels.

[38] M. A. RICHARDS, Fundamentals of Radar Signal Processing, McGraw-Hill, 2005.

[39] E. ASGEDOM , "INF5410 Array Signal Processing Lecure Notes:CH3," University of Oslo, Oslo, 2012.

[40] T. W. MILLER and R. A. MONZINGO, Introduction to Adaptive Arrays, New York: John Wiley & Sons, 1981.

[41] V. N. VAPNIK, The Nature of Statistical Learning Theory, Springer-Verlag, 2000.

[42] P. BARTLETT, "Reproducing Kernel Hilbert Spaces, Statistical Learning Theory," 2008.

**TEZ FOTOKOPİ İZİN FORMU**

**ENSTİTÜ**

Fen Bilimleri Enstitüsü ☐

Sosyal Bilimler Enstitüsü ☐

Uygulamalı Matematik Enstitüsü ☐

Enformatik Enstitüsü ☐

Deniz Bilimleri Enstitüsü ☐

**YAZARIN**

Soyadı : .....
Adı : .....
Bölümü : .....

**TEZİN ADI** (İngilizce) : .....
.....
.....
.....
.....

**TEZİN TÜRÜ** : Yüksek Lisans ☐ Doktora ☐

1. Tezimin tamamı dünya çapında erişime açılsın ve kaynak gösterilmek şartıyla tezimin bir kısmı veya tamamının fotokopisi alınsın. ☐
2. Tezimin tamamı yalnızca Orta Doğu Teknik Üniversitesi kullancılarının erişimine açılsın. (Bu seçenekle tezinizin fotokopisi ya da elektronik kopyası Kütüphane aracılığı ile ODTÜ dışına dağıtılmayacaktır.) ☐
3. Tezim bir (1) yıl süreyle erişime kapalı olsun. (Bu seçenekle tezinizin fotokopisi ya da elektronik kopyası Kütüphane aracılığı ile ODTÜ dışına dağıtılmayacaktır.) ☐

Yazarın imzası ..... Tarih .....