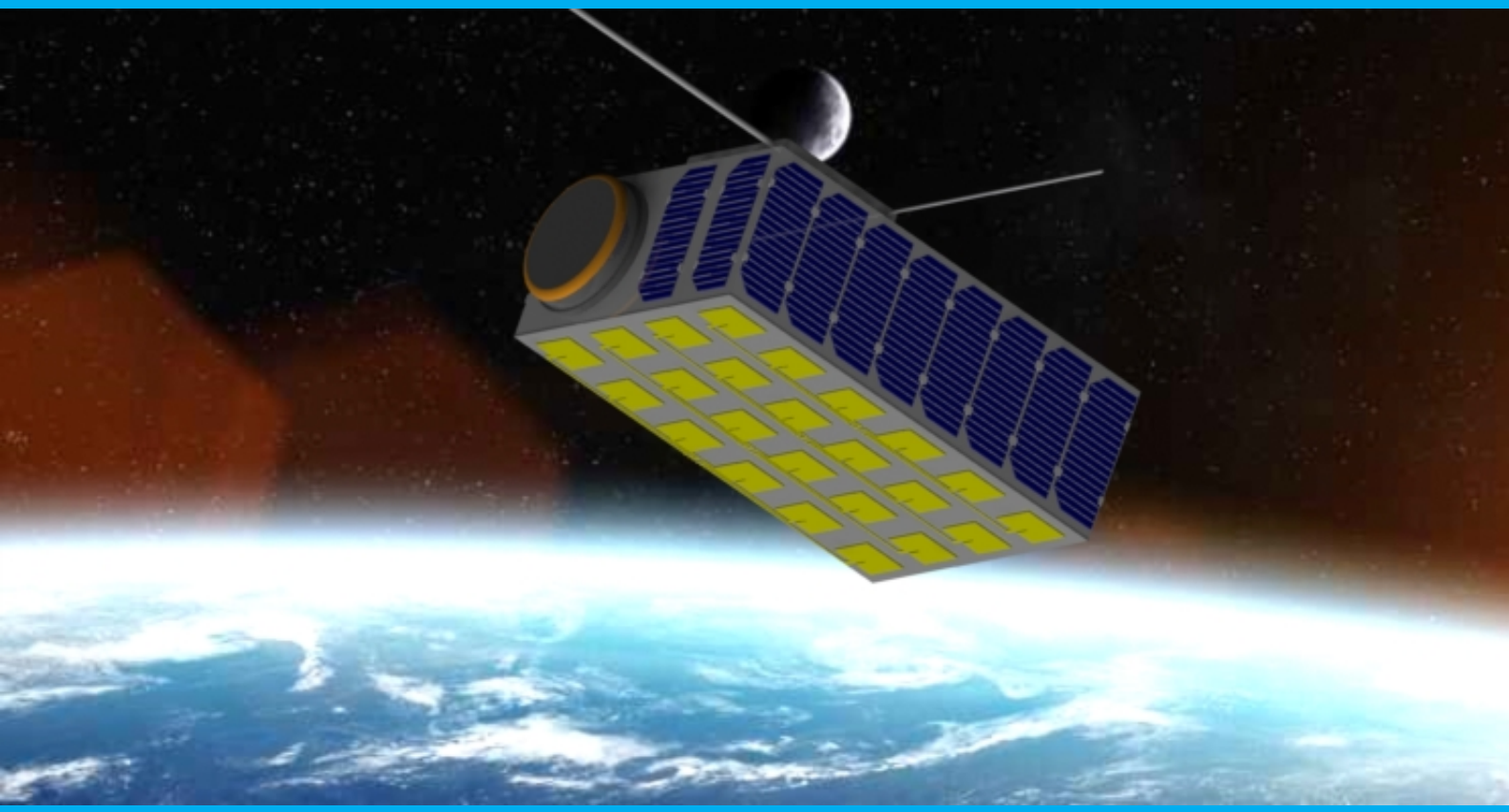
A Comparison of Angle-of-Arrival algorithms

for space based geolocation

C.H. Broekhuizen



TU Delft

Koninklijke Luchtmacht

A Comparison of Angle-of-Arrival algorithms

for space based geolocation

by

C.H. Broekhuizen

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Friday 8 November 2019.

Student number:	4720393	
Project duration:	April, 2019 – October, 2019	
Thesis committee:	Prof. dr. ir. E. K. A. Gill,	TU Delft, Chair
	Dr. A. Menicucci,	TU Delft, Supervisor
	Ir. K. J. Cowan,	TU Delft, Examiner
	Ir. B. A. Oving,	RNLAF, Additional

This thesis is confidential and cannot be made public until 31 October 2024

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Koninklijke Luchtmacht

Preface

This document describes and presents research activities and results produced over a time period of nine months as part of a project at the Royal Netherlands Air Force (RNLAF). This document is offered as a thesis report at the Delft University of Technology as part of the Graduate program Aerospace Engineering (MSc) in the SpaceFlight profile, specifically Space Systems Engineering. I would like to acknowledge my project supervisor, Dr. Alessandra Menicucci, for providing guidance, feedback and support throughout the duration of the project. I would also like to extend my gratitude to the RNLAF for providing me the opportunities and freedom to further develop myself as an engineer.

C.H. Broekhuizen
Delft, September 2019

Abstract

Geolocation of Radio Frequency (RF) emitters, is finding the geographical location of said emitter. This thesis explores the concepts of geolocation, specifically, Space-Based geolocation of earth-surface emitters by means of measuring the Angle-of-Arrival using a Uniform-Rectangular Antenna Array. Initially the literature is explored, the significance of Geolocation is made clear and the research outline and scope is defined. Afterwards, Angles-of-Arrival finding algorithms are investigated and implemented, specifically the subspace algorithms: "MUltiple SIgnal Classification" (MUSIC) and "Estimation of Signal Parameters via Rotational Invariance"(ESPRIT). These algorithms are thoroughly explained, a model is obtained and implemented into a Simulation framework. The Simulation Framework is used to create a Graphical Tool used to calculate performance figures of Space-Based Geolocation. By using the obtained Simulation, a sensitivity analysis is produced. This sensitivity analysis observes the effect of design parameters of a satellite platform and geolocation payload on the geolocation performance. This sensitivity analysis is used to find the optimal approach when designing a geolocation payload. This approach can be used to synthesize a geolocation payload design that maximizes performance.

List of Figures

1.1 Schematic overview of space-based geolocation 2

2.1 Methods of deriving the AoA from a signal 4
2.2 Schematic overview of geolocation using AoA 4
2.3 Frequency plot of narrow-band signal 5
2.4 ULA receiving RF signal in far-field, the array elements measure the signal at a smale time (phase) offset 5
2.5 3x3 URA in XY-Plane, each blue dot represents an antenna array element 6
2.6 2D MUSIC Spatial spectrum of incoming signal at $(\theta, \phi) = (40°, 200°)$ 9
2.7 Uniform Linear Array consisting of two identical overlapping sub-arrays. The blue and red rectangles represent the sub-arrays, the blue dots the antenna elements. 10
2.8 URA with two identical sub-arrays in x-direction 11
2.9 URA with two identical sub-arrays in y-direction 11

3.1 Schematic overview of software objects 16
3.2 Iterative smaller search-space and step-size of Peak search algorithm step size of 10°, 1° and .1° from left to right 18
3.3 Earth-Centered Earth-Fixed frame 19
3.4 Nort-East-Down frame 19
3.5 'North' Vector in ECEF Frame 19
3.6 NED Frame, satellite and emitter in ECEF frame 20
3.7 Estimated direction vector of emitter in satellite NED frame 20
3.8 Example of AOA-Chain simulation for both MUSIC and ESPRIT for relatively high noise (0 dB SNR) and low noise (10 dB SNR) 21
3.9 Polar plot of noiseless ESPRIT simulation 23
3.10 Geographic plot of noiseless ESPRIT simulation 23
3.11 Polar plot of noiseless ESPRIT simulation 24
3.12 Geographic plot of noisy ESPRIT simulation (5 dB) 24
3.13 Obtained Simulation versus *Zoltowski et al.*[28] 24
3.14 Polar plot of noiseless MUSIC simulation 25
3.15 Geographic plot of noiseless MUSIC simulation 25
3.16 Polar plot of noiseless MUSIC simulation 25
3.17 Geographic plot of noisy MUSIC simulation (5 dB) 25
3.18 Schematic overview of GUI structure 28
3.19 Schematic overview of *input parameters* window 29
3.20 Schematic overview of GUI 30
3.21 Overview of Graphical User Interface 30

4.1 Baseline Satellite-Target Location of Analysis 34
4.2 box plot of Geolocation error vs Signal-to-Noise 36
4.3 Geolocation RMSE vs Signal-to-Noise 37
4.4 Algorithm timing vs Signal-to-Noise 38
4.5 Geolocation RMSE vs SNR in semilog Scale 39
4.6 box plot of Geolocation error vs number of samples 40
4.7 Geolocation RMSE vs Number of Samples 41
4.8 Geolocation RMSE vs Number of Samples in Log-Log Scale 42
4.9 Algorithm run-time vs number of samples 42
4.10 Geolocation RMSE vs Number of Samples in Log-Log Scale 43
4.11 Propagation of circular orbit 44

4.12 Movement of satellite during 20 seconds measurement . . . 44
4.13 Geolocation RMSE vs Number of Samples including drift . . . 45
4.14 Geolocation RMSE vs Number of Samples including drift on Log-Log scale with proportionality approximation . . . 45
4.15 box plot of Geolocation error vs number of elements . . . 46
4.16 Geolocation RMSE vs Number of Elements in antenna array . . . 47
4.17 Geolocation RMSE vs Number of Elements in antenna array on log-log scale with approximation line . . . 47
4.18 Algorithm timing vs Number of elements . . . 48
4.19 Geolocation RMSE vs Number of Elements in antenna array . . . 48
4.20 Geolocation RMSE vs Number of Elements in antenna array with different configuration . . . 49
4.21 Geolocation RMSE vs Number of Elements in antenna array with different configuration on log scale . . . 49
4.22 two incoming signals on a 5x5 array, peaks are clearly separate . . . 50
4.23 two incoming signals on a 2x2 array, peaks are slightly merging . . . 50
4.24 three incoming signals on a 5x5 array, peaks are clearly separate . . . 50
4.25 three incoming signals on a 2x2 array, peaks are clearly merging . . . 50
4.26 box plot of Geolocation error vs element spacing . . . 51
4.27 Geolocation RMSE vs Element Spacing . . . 52
4.28 Algorithm timing vs Element Spacing . . . 53
4.29 Geolocation RMSE vs Element Spacing . . . 53
4.30 Schematic of increase of M causing decrease in d . . . 54
4.31 Geolocation RMSE vs increasing M, decreasing d . . . 54
4.32 Error in pointing propagating to geographic error . . . 55
4.33 Additive Geolocation RMSE vs Pointing uncertainty 3σ . . . 56
4.34 Error in position propagating to geographic error . . . 56
4.35 Additive Geolocation RMSE vs positioning uncertainty σ . . . 57
4.36 Effect on design parameter x on geolocation performance . . . 58
4.37 Overview of design steps to maximize Geolocation performance . . . 59

5.1 schematic of inset feed patch antenna . . . 62
5.2 Proposed Design of Patch Antenna Array . . . 62
5.3 Example Design of geolocation Satellite using the GLAP . . . 63
5.4 Analysis of Geolocation Accuracy using the AOA Tool . . . 65

List of Tables

1.1 Satellite and Payload Parameters of interest 2

3.1 Verification cases for geographical coordinates to angle-of-arrival conversion 22
3.2 Verification cases for geographical coordinates to angle-of-arrival conversion 22
3.3 Verification cases for angle-of-arrival conversion to geographical coordinates 23
3.4 Functional and Content Requirements of Graphical User Interface 27
3.5 Functional and Content Sub-Requirements of Graphical User Interface 27
3.6 Available user input parameters 29
3.7 Functional and Content Sub-Requirements of Graphical User Interface 31
3.8 Functional and Content Requirements of Graphical User Interface 31

4.1 Parameters investigated during sensitivity analysis 33
4.2 Baseline Parameters of Analysis 33
4.3 Example Calculation SNR 35
4.4 Performance of geolocation (RMSE km) vs SNR 37
4.5 Algorithm timing vs Signal-to-Noise 38
4.6 Geolocation RMSE vs Number of Samples 41
4.7 Algorithm run-time vs number of samples 42
4.8 Geolocation RMSE vs Number of Elements in antenna array 47
4.9 Algorithm timing vs Number of elements 48
4.10 Geographic coordinates of two additional earth-surface emitters 48
4.11 Geolocation RMSE vs element spacing 52
4.12 Algorithm timing vs Element Spacing 53
4.13 Geolocation RMSE vs increasing M, decreasing d 54
4.14 Distribution in pointing accuracy and how-much additive RMSE in geolocation 56
4.15 Distribution in pointing accuracy and how-much additive RMSE in geolocation 57
4.16 Investigated parameters with their respective effect and optimization approach 57

5.1 Mock Mission Requirements 61
5.2 Calculation SNR for Mission Scenario 64
5.3 Selected GLAP design parameters 64
5.4 Selected scenario location Parameters of Analysis 64
5.5 Results of RMSE after simulation 65

Contents

List of Figures vii

List of Tables ix

1 Introduction 1
- 1.1 Motivation of Research . . . 1
- 1.2 Research Project Outline . . . 2

2 Theory and Model of Geolocation techniques 3
- 2.1 Overview of Geolocation Techniques . . . 3
- 2.2 Geolocation using Angle-Of-Arrival . . . 4
  - 2.2.1 Emitter Model . . . 4
  - 2.2.2 Receiver Model. . . . 5
  - 2.2.3 AoA Algorithms . . . 7

3 Simulation of Geolocation Antenna Array 15
- 3.1 Overview of Simulation Chain . . . 15
- 3.2 Implementation . . . 16
- 3.3 Simulation Verification . . . 22
- 3.4 Graphical User Interface . . . 25
  - 3.4.1 Strategy . . . 26
  - 3.4.2 Scope . . . 26
  - 3.4.3 Structure . . . 28
  - 3.4.4 Skeleton . . . 28
  - 3.4.5 Surface . . . 30
- 3.5 Graphical User Interface Verification and Validation . . . 30

4 Sensitivity Analysis 33
- 4.1 Overview of Parameters . . . 33
- 4.2 Performance vs Signal-to-Noise . . . 34
- 4.3 Performance vs Number of Samples . . . 40
  - 4.3.1 Performance vs Number of Elements . . . 45
  - 4.3.2 Performance vs Element Spacing . . . 51
- 4.4 Performance vs Satellite Platform Parameters . . . 55
  - 4.4.1 Performance vs Satellite Pointing Accuracy . . . 55
  - 4.4.2 Performance vs Satellite Positioning Accuracy . . . 56
- 4.5 Summary of Sensitivity Analysis . . . 57
- 4.6 Recommendation of Design Synthesis . . . 58

5 Test Case: The design of a geolocation antenna array 61
- 5.1 Proposed mission scenario . . . 61
- 5.2 Satellite and Antenna Array Parameters . . . 61
- 5.3 Performance of Proposed Satellite . . . 64

6 Conclusion 67
- 6.1 Summary and Key Outcomes . . . 67
- 6.2 Recommendations for Future Work . . . 67

Bibliography 69

1 Introduction

Geolocation is the process of determining the geographical-location of a target. In practice it is possible to achieve this localization by exploiting radio transmissions. The geolocation concept discussed in this document is about locating a surface based emitter by means of space based sensing. Obtaining information about the geographical-location of radio emitters could potentially provide an advantage. This study has been carried out in collaboration with the Royal Netherlands Air Force (RNLAF), The Royal Netherlands Air Force is a modern and technology-rich part of the Dutch ministery of Defense that is internationally active[1]. The RNLAF started space activities in 2013 with the founding of 'bureau Space', which currently transitioned into 'Section Space'.

1.1. Motivation of Research

The RNLAF's Section Space has several tasks and projects, some of these are: to increase space situational awareness, to gain knowledge in the area of space developments and operations, increase space awareness within the Ministery of Defense, and development of small satellite projects. One of the interests of Section Space is to gain knowledge and explore the concept of space based geolocation satellites. Space based geolocation can have a variety of applications, for example to find and locate interfering emitters, to locate emergency beacons, locating ship-radar, locating signal jammers on GNSS frequencies, etc. The primary focus of this project is to investigate space based geolocation of radio emitters using the Angle-of-Arrival (AoA). Due to these techniques requiring the least amount satellites and the least amount of prior knowledge of the signal (as further explained in section 2.1). Evaluations of geolocation of AoA already exist and most commonly presents a Root-Mean-Squared-Error of the estimated AoA of the incoming signal plotted against the Signal-to-Noise ratio or presents observations regarding the estimated signal spatial spectrum. These existing evaluations are primarily not specified for space based geolocation and the results given are also commonly only in one dimension and do not provide the accuracy of the geolocation fully end-to-end (e.g. Longitude and Latitude instead of incoming angle). The studies that do take a space-based system limit themselves to for example one specific algorithm or antenna array configuration[15]. The novelty of this thesis is to:

- Provide a complete overview , simulation and analysis of existing algorithms.
- Extend Angle-of-Arrival algorithms to Geolocation algorithms.
- Provide end-system performance figures (e.g. RMSE of longitude and latitude).
- Investigate design synthesis procedures.
- Include a complete implementation of a simulation of AoA based geolocation.

1.2. Research Project Outline

In this thesis, AoA estimation algorithms are investigated and implemented, and the entire system is modeled and simulated in order to determine performance figures. A sensitivity analysis is performed on this performance to find how parameters of the model affect the geolocation performance. This is done in order to obtain an insight in the design synthesis of such a system. Finally, using the obtained insight on design synthesis, a test-case geolocation mission is analyzed and an geolocation spacecraft is designed.

A sketch of a space based geolocation scenario can be seen in figure 1.1. Using this sketch, one can see that the geolocation problem consists of three elements, namely: the emitter model, the link model and the spacecraft model. The spacecraft model consists of a platform and a payload capable of doing AoA measure-

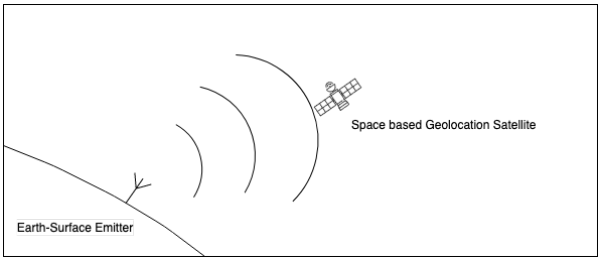


Figure 1.1: Schematic overview of space-based geolocation

ments of an RF signal. A payload capable of doing this is an RF receiver payload and in order to obtain AoA data, the payload should be an array of RF receivers. Thus the spacecraft model can be split up into the platform model and the payload.

Model parameters of interest for sensitivity analysis defined in this project can be found in table 1.1. A more detailed explanation per parameter is given in section 4.

Symbol	Parameter Description	Unit
	Payload Parameters	
Z	**Additive Noise of received Signal:** This model parameter describes the amount of noise received by the payload, this value is commonly defined using the Signal-to-Noise Ratio (SNR)	dB
N	**Number of Samples:** Amount of measurements obtained by the payload before estimating the AoA. The number of samples might be limited by storage capabilities, and obtaining a high number of frequency might not be beneficial to the geolocation performance due to the satellite drift during sampling.	-
M	**Number of elements:** Number of antenna elements per dimension in a $(M \times M)$ antenna array.	-
d	**Element Spacing:** Spacing between the elements of the Array.	-
	Satellite Parameters	
θ_{error}	**Pointing Error:** Pointing Determination error of satellite platform, defined by performance of the Attitude control and Determination (ACDS) of the satellite. The performance figure of ACDS Module of interest is the Attitude determination, as an error in the measured attitude and the real attitude propagates into an error in the geolocation. Determination performance figures are commonly given in variance of a standard distribution (σ)	Degrees
x_{error}	**Position Error:** Positioning error of satellite platform, defined by performance of the Position control and Determination of the satellite. Position Determination is commonly performed by GNSS modules. Performance figures are also commonly given in variance of a standard distribution (σ)	m

Table 1.1: Satellite and Payload Parameters of interest

2

Theory and Model of Geolocation techniques

2.1. Overview of Geolocation Techniques

By 'geolocation', the process of determining the geographical location of a target is described. Geolocation of RF emitters is done by obtaining information of the emitter by receiving the signal. Common methods of RF geolocation are:

- **AoA - Angle of Arrival**: Finding the direction of an incoming RF to estimate the emitter location.
- **TOA - Time of Arrival**: Using the timing information of the emitter, one can estimate the emitter location by measuring the time of arrival at the receiver.
- **FOA - Frequency of Arrival**: Using information of the emitter's frequency, one can observe the frequency difference at the receiver (e.g. differential Doppler) to estimate the emitter location.

Noticeable from the methods described above is that TOA and FOA require prior knowledge of the emitter to work, namely the timing and frequency respectively. In order to geolocate an emitter without this prior knowledge, multiple receivers can work together and the differential of the measurement:

- **TDOA - Time Difference of Arrival**: One can use the differential of the timing of arrival to estimate the emitter geographic location.
- **FDOA - Frequency Difference of Arrival**: Using multiple measurements of the arrival frequency, one can look at the frequency difference (differential Doppler) between the receivers to estimate the emitter location.

When comparing these methods, the AoA method allows one to obtain the geolocation of the emitter without requiring differential measurements and requires the least amount of prior knowledge of the emitter.

2.2. Geolocation using Angle-Of-Arrival

Geolocation using the AoA requires that the AoA is extracted from the received signal. In order to extract this AoA, an antenna array can be implemented and the phase difference per element can be used to derive the AoA. Methods for obtaining the AoA can also be divided into sub-categories, namely quadratic estimators and subspace algorithms[27]. These two categories and their respective most common algorithm can be seen in figure 2.1.

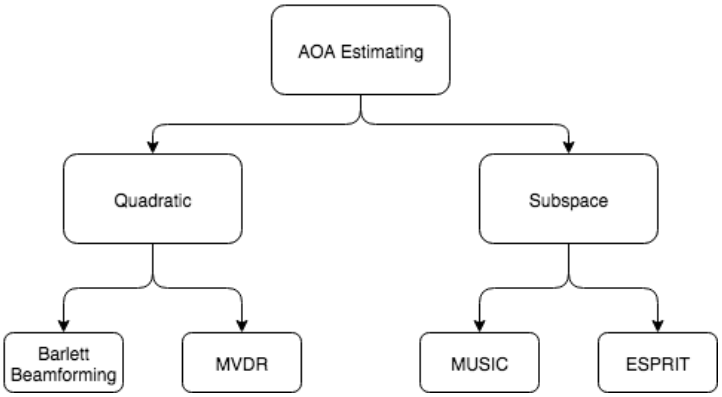


Figure 2.1: Methods of deriving the AoA from a signal

Quadratic estimators estimate the angle-of-arrival by means of maximizing spectral power, or most-likelyhood calculations [27]. Subspace algorithms however, estimate the angle-of-arrival by means of the eigenspace of the received signal. Subspace based algorithms have a superior accuracy over quadratic estimators since more information is added to the equation array[5]. However, subspace algorithms are much more computationally intensive as it requires eigenvalue- or singular value decomposition. The focus in this thesis are on subspace algorithms, due to the superior accuracy.

The process of geolocation using AoA can be observed in figure 2.2. The earth surface emitter broadcasts signal S and is received by the Antenna Array with the incoming angle-of-arrival (θ, ϕ) based on the location of the emitter (lat, lon). From the received signal $\tilde{X}$ the AoA is estimated to be $(\tilde{\theta}, \tilde{\phi})$. From the estimated AoA the geographic location of the emitter is estimated to $(\tilde{lat}, \tilde{lon})$

Figure 2.2: Schematic overview of geolocation using AoA

2.2.1. Emitter Model

In order to keep the project constraint to a defined scope, the following assumptions are made about the emitter:

1. The emitter is static (non-moving).
2. The emitter is Earth-surface based.
3. The emitter broadcasts a narrow-band RF signal (e.g. Phase-Shift-Keying modulation) at fixed frequency.

A signal is defined as being narrow-band if it's center frequency is much higher than it's occupied bandwidth ($f_c >> B$, figure 2.3).

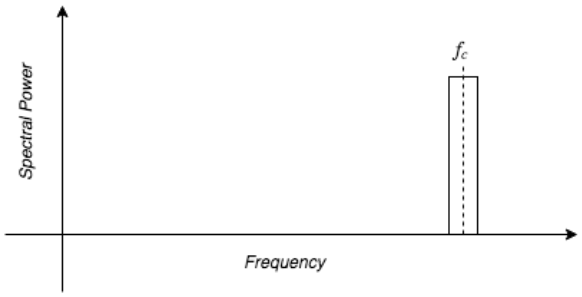


Figure 2.3: Frequency plot of narrow-band signal

A narrow-band RF signal can be represented as in equation (2.1). Where $\rho_s(t)$ is the information (base-band) signal, ω_c the carrier frequency and $s(t)$ the complex envelope of the signal. An important property of such a narrow-band signal for the consecutive models is that a small delay in the signal can be approximated as equation (2.2), given that such a delay τ is much smaller than half the base-band bandwidth ($\tau << \frac{1}{2B_\rho}$), meaning that when a narrow-band signal is delayed by τ this can be approximated as purely a phase shift.

$$s(t) \approx \rho_s(t) \cdot e^{j\omega_c t} \tag{2.1}$$

$$s(t+\tau) \approx \rho_s(t+\tau) \cdot e^{j\omega_c(t+\tau)} \approx \rho_s(t) \cdot e^{j\omega_c(t+\tau)} \approx s(t) \cdot e^{j\omega_c \tau} \tag{2.2}$$

2.2.2. Receiver Model

As mentioned earlier, the AoA of an incoming signal can be estimated from the phase difference when having multiple receiver antenna's. A system consisting of multiple antennas in a certain formation is commonly referred to as an antenna array. The antenna array can have different geometries, however, in order to further constraint the scope of the thesis, the antenna array geometries of interest are the so-called Uniform-Rectangular-Array (URA).

2D Receiver Array

Initially the receive model is derived in two dimensions, using a Uniform-Linear-Array (ULA) instead of an URA. When an narrow-band RF signal ($s_j(t)$) arrives at the antenna array it is assumed that the array is in the emitter's far-field, meaning the signal impinges as a plane-wave. When such a plane-wave arrives at the array at an angle (θ_j), the antenna array elements sense the signal at a slight offset in time, due to this angle. This effect is represented in figure 2.4.

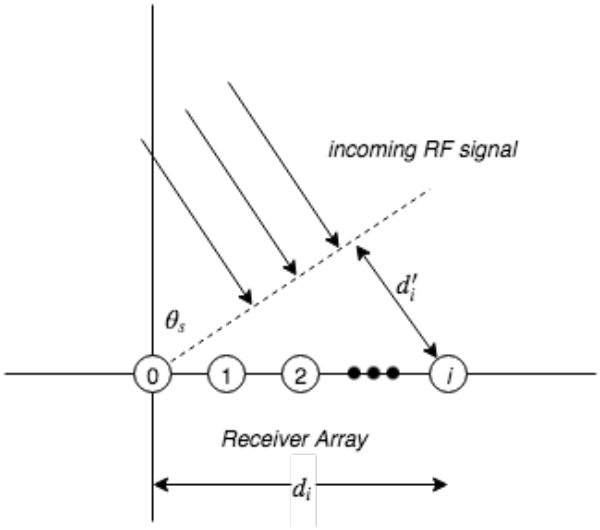


Figure 2.4: ULA receiving RF signal in far-field, the array elements measure the signal at a smale time (phase) offset

The exact timing offset induced can be calculated using the antenna geometry as seen in equation (2.3). Where $\tau_{i,j}$ is the timing offset of element i of the measurement of incoming wave-front $s_j(t)$, $d'_{i,j}$ is the distance of element i to incoming plane-wave $s_j(t)$ and c is the wave propagation speed ($c \approx$ *lightspeed* for RF signals).

$$\tau_{i,j} = \frac{d'_{i,j}}{c} \tag{2.3}$$

The distance $d'_{i,j}$ between the element and the incoming plane-wave can be calculated using the dot-product between the incoming signal $s_j(t)$ and a vector pointing to element i. This can be observed in equation (2.4).

$$d'_{i,j} = \vec{s_j} \cdot \vec{e_i} \tag{2.4}$$

where $\vec{s_j}$ is the unit direction vector of incoming wave $s_j(t)$ and $\vec{e_i}$ is the vector pointing to element i. In case of an antenna array as shown by figure 2.4, the direction vectors can be given by equation (2.5).

$$\vec{s_j} = \begin{bmatrix} \sin\theta_j \\ \cos\theta_j \end{bmatrix}, \quad \vec{e_i} = \begin{bmatrix} d_i \\ 0 \end{bmatrix} \tag{2.5}$$

Substituting equation (2.5) in equation (2.4) and (2.3), one can derive equation (2.6)

$$\tau_{i,j} = \frac{d_i \sin\theta_j}{c} \tag{2.6}$$

3D Receiver Array

In order for the receiver model to handle incoming signals in a three dimensional space, the equations need to be rewritten. It is assumed that the antenna array is an URA in the XY-Plane (figure 2.5) and the incoming angle-of-arrival is described by the inclination θ_j and azimuth ϕ_j.

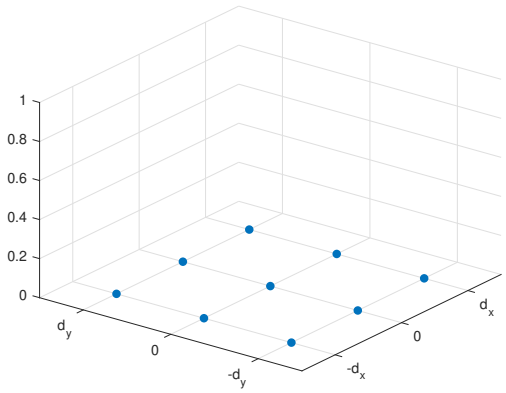


Figure 2.5: 3x3 URA in XY-Plane, each blue dot represents an antenna array element

When the equations are expanded to this 3D environment, the signal direction unit vector s_j and element position vector e_i can be given by equations (2.7).

$$\vec{s_j} = \begin{bmatrix} \sin\theta_j \cos\phi_j \\ \sin\theta_j \sin\phi_j \\ \cos\theta_j \end{bmatrix}, \quad \vec{e_i} = \begin{bmatrix} x_i \\ y_i \\ 0 \end{bmatrix} \tag{2.7}$$

Note that $z_i = 0$, due to all elements being in the XY-Plane. The timing difference $\tau_{i,j}$ can be calculated in three dimensions by substituting equation (2.7) in equation (2.4) and (2.3) leading to equation (2.8).

$$\tau_{i,j} = \frac{x_i \sin\theta_j \cos\phi_j + y_i \sin\theta_j \sin\phi_j}{c} \tag{2.8}$$

Phase shift and sample model

By substituting the timing offset (equation (2.8)) in the narrow-band approximation (equation (2.2)) given that $\omega_c = 2\pi c/\lambda$, , where ω_c is the carrier frequency and λ the carrier wavelength which leads to equation (2.9)

$$s(t+\tau_{i,j}) \approx s(t)\cdot e^{j\omega_c\tau_{i,j}} \approx s(t)\cdot e^{j2\pi d'_{i,j}/\lambda_c} \approx s(t)\cdot e^{j\Phi_{i,j}} \tag{2.9}$$

This equation suggest that each element i in the array receives the identical signal $s_j(t)$ rotated by a phase shift $\Phi_{i,j}$ on received signal. Where this phase shift $\Phi_{i,j}$ is given by equation (2.10).

$$\Phi_{i,j} = 2\pi d'_{i,j}/\lambda = \begin{cases} \text{1D:} & 2\pi(x_i \sin\theta_j)/\lambda \\ \text{2D:} & 2\pi(x_i \sin\theta_j \cos\phi_j + y_i \sin\theta_j \sin\phi_j)/\lambda \end{cases} \tag{2.10}$$

Now given an antenna array with $i = 0,1,...,M-1$ elements in the antenna array and $j = 0,1,...,D-1$ incoming RF plane-waves $s_j(t)$, each element i in the antenna array measures the sum of all D signals, phase shifted at $\Phi_{i,j}$ respectively. Thus, the received measurement of antenna element i can be given by equation (2.11)

$$x_i(t) = \sum_{j=0}^{D} s_j(t)\cdot e^{j\Phi_{i,j}} \tag{2.11}$$

Equation (2.11) can be rewritten into an matrix calculation which can be seen in equation (2.12).

$$\begin{bmatrix} x_0 \\ x_i \\ \vdots \\ x_M \end{bmatrix} = \begin{bmatrix} e^{j\Phi_{0,0}} & e^{j\Phi_{0,j}} & \dots & e^{j\Phi_{0,D}} \\ e^{j\Phi_{i,0}} & e^{j\Phi_{i,j}} & \dots & e^{j\Phi_{i,D}} \\ \vdots & \vdots & \ddots & \vdots \\ e^{j\Phi_{M,0}} & e^{j\Phi_{M,j}} & \dots & e^{j\Phi_{M,D}} \end{bmatrix} \begin{bmatrix} s_0 \\ s_j \\ \vdots \\ s_D \end{bmatrix} \tag{2.12}$$

$$\mathbf{X} = \mathbf{AS} \tag{2.13}$$

This equation can be used to obtain the measured samples of all elements in the antenna array simultaneously in one calculation. Rewriting this matrix calculation in short gives equation (2.13), where X are the received samples A is the steering matrix and S are the incoming wave-fronts.

The antenna array should induce noise on the measured samples, thus a noise vector Z should be added to the measurements, which can be represented as mean zero additive white Gaussian noise ($Z \sim N(0,\sigma_N^2)$).

$$\tilde{\mathbf{X}} = \mathbf{AS} + \mathbf{Z} \tag{2.14}$$

Note that when multiple measurements are taken over time ($t = 0,1,...,T$), this can be modeled into the same matrix calculation by simply extending the incoming wave-front matrix S.

$$X = \begin{bmatrix} x_{0,0} & \dots & x_{0,T} \\ x_{i,0} & \dots & x_{i,T} \\ \vdots & \vdots & \vdots \\ x_{M,0} & \dots & x_{M,T} \end{bmatrix} = \begin{bmatrix} e^{j\Phi_{0,0}} & e^{j\Phi_{0,j}} & \dots & e^{j\Phi_{0,D}} \\ e^{j\Phi_{i,0}} & e^{j\Phi_{i,j}} & \dots & e^{j\Phi_{i,D}} \\ \vdots & \vdots & \ddots & \vdots \\ e^{j\Phi_{M,0}} & e^{j\Phi_{M,j}} & \dots & e^{j\Phi_{M,D}} \end{bmatrix} \begin{bmatrix} s_{0,0} & \dots & s_{0,T} \\ s_{j,0} & \dots & s_{j,T} \\ \vdots & \vdots & \vdots \\ s_{D,0} & \dots & s_{D,T} \end{bmatrix} \tag{2.15}$$

2.2.3. AoA Algorithms

As mentioned prior, the algorithms of interest are subspace algorithms and subspace algorithms exploit the eigenspace of the samples. More specifically, the subspace algorithms exploit the eigenspace of the autocovariance of the obtained samples. The autocovariance of the sample matrix can be found using equation (2.16). By substituting the obtained receive model (2.13) into the autocovariance equation, equation (2.17) is obtained[24].

$$\mathbf{R_X} \triangleq E[\mathbf{X},\mathbf{X}^H] \tag{2.16}$$

$$\mathbf{R_X} \triangleq \mathbf{A}E[\mathbf{S},\mathbf{S}^H]\mathbf{A}^H + E[\mathbf{Z},\mathbf{Z}^H] = \mathbf{A}E[\mathbf{S},\mathbf{S}^H]\mathbf{A}^H + \sigma_N^2\mathbf{I}_M \tag{2.17}$$

In this equation, the operation $E[\cdot]$ and $(\cdot)^H$ represent the expected value and the Hermitian transpose respectively. Decomposing the autocovariance into its eigenvalues and eigenvectors yields a decomposition into two seperate eigenspaces, namely the signal-space and the noise-space[24]. This is shown in equation. (2.18)

$$\mathbf{R_X} = \mathbf{E_X}\Lambda_\mathbf{X}\mathbf{E}_\mathbf{X}^H = \mathbf{E}_S\Lambda_S\mathbf{E}_S^H + \mathbf{E}_N\Lambda_N\mathbf{E}_N^H \tag{2.18}$$

Where E_X are the eigenvectors and Λ_X a diagonal matrix of the eigenvalues. E_S and E_N are the signal- and noise-eigenvectors and Λ_S and Λ_N which are the signal- and noise eigenvalues[22]. Assuming that the steering matrix and the signal autocovariance are full rank, the amount of eigenvalues and eigenvectors should be M (number of elements in array)

Given a sample matrix $\tilde{\mathbf{X}}$, it's autocovariance matrix can not be obtained using equation (2.18), since the exact signal information is unknown. Instead, a more heuristic approach can be used and the autocovariance $\tilde{\mathbf{R}_\mathbf{X}}$ can be obtained using the samples(Equation (2.19)).

$$\tilde{\mathbf{R}_\mathbf{x}} = \tilde{\mathbf{X}}\tilde{\mathbf{X}}^H \tag{2.19}$$

This sample autocovariance $\tilde{\mathbf{R}_\mathbf{x}}$ can similarly be decomposed into it's noise subspace and signal subspace according to equation (2.18). There are multiple subspace algorithms that exploit these sub-spaces in order to obtain the AoA. The most widely used algorithms are the Multiple Signal Classification (MUSIC) Algorithm[23] and Estimation of Signal Parameters using Rotational Invariance Techniques (ESPRIT)[21].

MUSIC

The first algorithm discussed is the MUSIC algorithm[23]. The MUSIC algorithm can be used to find an estimate of the angle-of-arrival of the incoming signal by exploiting the eigenvectors of the noise-space that occupy the null-space of the entire eigenspace.

The steering matrix $\mathbf{A}$ (equation (2.12)) consists of linearly independent vectors (columns), thus $\mathbf{A}$ is full rank. If the incident wave-fronts on the antenna array are not correlated, then the autocovariance matrix $\mathbf{R}_S = E[\mathbf{S},\mathbf{S}^H]$ is non-singular, since it's columns are also independent. This means that $\mathbf{A}\mathbf{R}_S\mathbf{A}^H$ in equation (2.17) is of rank D (number of impinging signals), which implies that $M-D$ eigenvalues of $\mathbf{A}\mathbf{R}_S\mathbf{A}^H$ are zero.

Adding this information to equation (2.17) means that $\mathbf{R}_x$ has $M-D$ eigenvalues that are equal to σ_N^2. Note that this only works under the assumption that the incoming signals are non-coherent and the signals and noise are not correlated (as is the case with Gaussian noise).

Equation (2.20) must hold for any pair of eigenvalue (λ_i) and eigenvector ($\mathbf{v}_i$) of $\mathbf{R}_x$

$$(\mathbf{R}_x - \lambda_i\mathbf{I})\mathbf{v}_i = 0 \tag{2.20}$$

By substituting equation (2.17) in equation (2.20) for the $M-D$ eigenvalues that are equal to σ_N^2 and knowing that $E[\mathbf{S},\mathbf{S}^H]$ is non-singular leads to equation (2.21)

$$(\mathbf{R}_x - \sigma_N^2\mathbf{I})\mathbf{v}_i = \mathbf{A}E[\mathbf{S},\mathbf{S}^H]\mathbf{A}^H\mathbf{v}_i = \mathbf{A}^H\mathbf{v}_i = 0 \quad \text{for } i = D+1, D+2, ..., M \tag{2.21}$$

This equation suggests that the eigenvectors from the $M-D$ eigenvalues (that are equal to σ_N^2) belong to a noise space. Which is orthogonal to the steering matrix $\mathbf{A}$[23].

$$\mathbf{A} \perp \mathbf{E}_N \tag{2.22}$$

Since these matrices are orthogonal, the euclidean distance between the steering vector and the noise-subspace should theoretically be zero:

$$||\mathbf{E}_N\mathbf{A}|| = \mathbf{A}\mathbf{E}_N\mathbf{E}_N^H\mathbf{A}^H = 0 \text{ for } \theta = \theta_i \tag{2.23}$$

Taking the inverse of this equation leads to the MUSIC Spatial Spectrum equation (2.24)[23], which should be maximum when the selected steering matrix $\tilde{\mathbf{A}}(\theta,\phi)$ is close to the real (unknown) steering matrix $\mathbf{A}$.

$$P_{MUSIC}(\theta,\phi) = \frac{1}{||\tilde{\mathbf{E}}_N\tilde{\mathbf{A}}(\theta,\phi)||} = \frac{1}{\tilde{\mathbf{A}}(\theta,\phi)\tilde{\mathbf{E}}_N\tilde{\mathbf{E}}_N^H\tilde{\mathbf{A}}^H(\theta,\phi)} \tag{2.24}$$

An example of the MUSIC spatial spectrum has been calculated for a steering matrix with one incoming wave-front at $(\theta,\phi) = (40°, 200°)$. This spatial spectrum can be seen in figure 2.6.

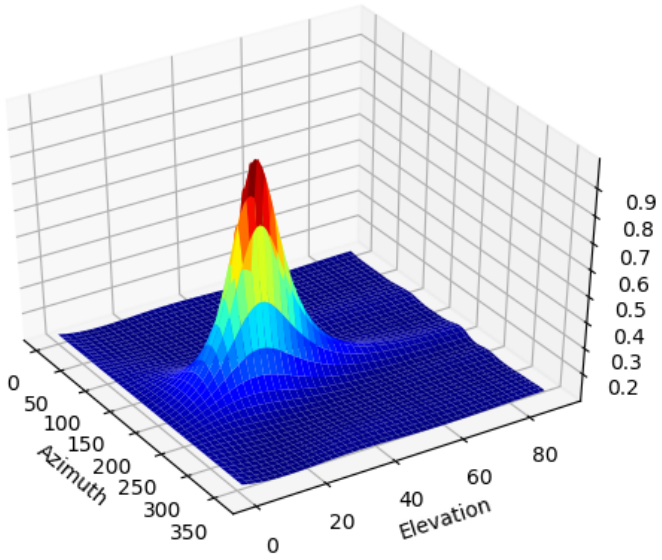


Figure 2.6: 2D MUSIC Spatial spectrum of incoming signal at $(\theta,\phi) = (40°, 200°)$

Thus, in order to find the angle-of-arrival of the incoming signals using the MUSIC algorithm, one has to take the following steps[23]:

1. Collect sample matrix $\tilde{\mathbf{X}}$ and construct autocovariance matrix $\tilde{\mathbf{R}}_{\mathbf{X}}$.

2. Obtain noise space $\tilde{\mathbf{E}}_N$ by eigendecomposition of $\tilde{\mathbf{R}}_{\mathbf{X}}$.

3. Evaluate P_{MUSIC} (Eqn. (2.24)) for range of (θ,ϕ).

4. Pick D peaks from P_{MUSIC}, which are the estimated AoA

The final step in the MUSIC algorithm implements a peak search, being one of the major drawbacks of the algorithm, since a peak-search is computationally intensive and costly in time.

ESPRIT

ESPRIT (Estimation of Signal Parameters using Rotational Invariance Techniques) is another technique used to obtain the AoA. ESPRIT exploits a rotational invariance in the signal eigenspace caused by a translation invariance in the antenna array[21]. The ESPRIT algorithm produces a result without using a search space, hence decreasing the computational time in respect with the MUSIC algorithm [2].

The ESPRIT algorithm does put a physical constraint on the antenna array because it assumes that the array is made-up of identical sub-arrays with a known displacement between them. These sub-arrays may be overlapping. A 1D example of two identical sub-arrays being 'selected' from a Uniform Linear Array (ULA) can be seen in figure 2.7.

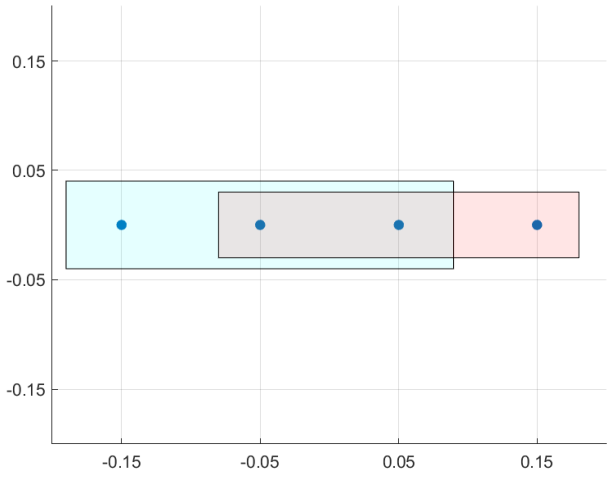


Figure 2.7: Uniform Linear Array consisting of two identical overlapping sub-arrays. The blue and red rectangles represent the sub-arrays, the blue dots the antenna elements.

Following the example given in figure 2.7, one can observer that the two sub-arrays are translated by a fixed value (Δ_x). Similarly to equation (2.10), this translation leads to a phase difference between the samples of the two sub-arrays (ϕ). This effect can be modeled in the steering matrix A as seen in equation (2.25)

$$\mathbf{A}_2 = \mathbf{A}_1\mathbf{\Phi} \tag{2.25}$$

where Φ is the rotational invariance (phase difference) induced by the fixed translation between sub-array 1 and sub-array 2. Due to this translation being purely in the x-direction, Phi is defined as a diagonal matrix with phase-shift Φ_j for every incoming signal $j = \{0, 1, ..., D-1\}$ (Eqn. (2.26)).

$$\mathbf{\Phi} = \text{diag}\{e^{j\Phi_0}, ..., e^{j\Phi_{D-1}}\} \quad \text{, where} \quad \Phi_j = 2\pi\Delta_x \sin\theta_j/\lambda \tag{2.26}$$

Since both the signal covariance matrix $\mathbf{R}_{ss}$ and the steering matrix $\mathbf{A}$ have full rank (as in eqn. (2.17)) and decompose in the signal subspace $\mathbf{E}_s$ (as in eqn. (2.18)), there exists a unique matrix $\mathbf{T}$ such that equation (2.27) holds[2].

$$\mathbf{E}_s = \mathbf{AT} \tag{2.27}$$

Now substituting equation (2.25) in equation (2.27), one gets the results shown in equations (2.28)

$$\mathbf{E}_1 = \mathbf{A}_1\mathbf{T} \tag{2.28a}$$

$$\mathbf{E}_2 = \mathbf{A}_1\mathbf{\Phi T} \tag{2.28b}$$

In similar fashion to equation (2.25), one can construct the same invariance for the signal spaces $\mathbf{E}_1$ and $\mathbf{E}_2$ [13].

$$\mathbf{E}_2 = \mathbf{\Psi E}_1 \tag{2.29}$$

By substituting equation (2.29) into equation (2.28) one can find that the relationship between $\mathbf{\Psi}$ and $\mathbf{\Phi}$ is one of eigendecomposition (equation (2.30))[2].

$$\mathbf{AT\Psi} = \mathbf{A\Phi\Psi T} \rightarrow \mathbf{\Psi} = \mathbf{T\Phi T}^{-1} \tag{2.30}$$

Acknowledging that the data has a rotational invariance induced by a translation, one can find the AoA by doing the inverse. In a more practical approach, the sub-array does not have to be physically implemented but can be 'virtual' by using a selection matrix on the actual antenna array.

$$\mathbf{A}_1 = \mathbf{J}_1\mathbf{A}, \quad \mathbf{A}_2 = \mathbf{J}_2\mathbf{A} \quad \rightarrow \quad \mathbf{J}_2\mathbf{A} = \mathbf{J}_1\mathbf{A\Phi} \tag{2.31a}$$

$$\mathbf{E}_1 = \mathbf{J}_1\mathbf{E}_s, \quad \mathbf{E}_2 = \mathbf{J}_2\mathbf{E}_s \quad \rightarrow \quad \mathbf{J}_2\mathbf{E}_s = \mathbf{J}_1\mathbf{E}_s\boldsymbol{\Psi} \tag{2.31b}$$

Note that ESPRIT allows for overlapping sub-arrays, selecting a maximum overlapping sub-array ensures the maximum amount of elements per sub-array.

In practice, the data available is noisy, thus solving the invariance equation (2.31) cannot be solved exactly, hence a solving technique is required. Approaches to solve the invariance can be e.g. a Least-Squares approach, Total-Least-Squares approach or a Moore-Penrose inverse.

After $\boldsymbol{\Psi}$ is obtained by solving the invariance equation using the signal sub-space. One can find the invariance $\boldsymbol{\Phi}$ by means of eigendecomposition. Afterwards, one can find the phase-shifts ($\mu_j = e^{j\Phi_j}$) induced by the sub-array translation by taking the diagonals of $\boldsymbol{\Phi}$ (Eqn. (2.26)). The AoA impinging on the antenna array in figure 2.7 can be extracted from the eigenvalues of $\boldsymbol{\Phi}$ using the inverse of equation (2.26).

$$\phi_j = \arg\{\mu_j\} \quad \rightarrow \quad \theta_j = \arcsin\frac{\lambda\phi_j}{2\pi\Delta_x} \tag{2.32}$$

In the case of estimating the AoA in 2D (elevation θ_j and azimuth ϕ_j) using an URA, one can extend the ESPRIT algorithm in another dimension, meaning sub-arrays need to be selected in both dimensions. An example of this can be seen in figure 2.8.

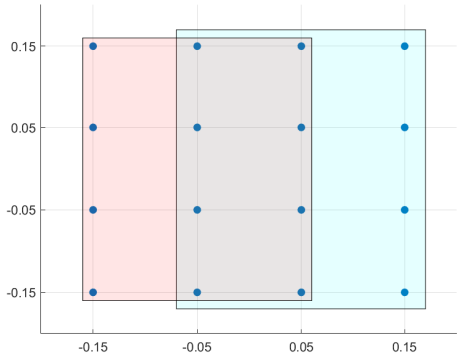


Figure 2.8: URA with two identical sub-arrays in x-direction

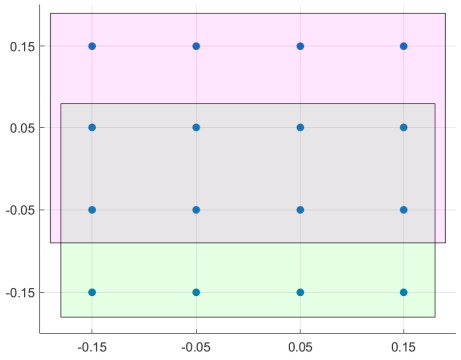


Figure 2.9: URA with two identical sub-arrays in y-direction

The invariance equations of the steering and signal-space matrices are obtained in similar fashion to the 1D scenario ((2.31)) and are represented in equation (2.33).

$$\mathbf{J}_{\mu_2}\mathbf{A} = \mathbf{J}_{\mu_1}\mathbf{A}\boldsymbol{\Phi}_\mu \tag{2.33a}$$

$$\mathbf{J}_{\nu_2}\mathbf{A} = \mathbf{J}_{\nu_1}\mathbf{A}\boldsymbol{\Phi}_\nu \tag{2.33b}$$

$$\mathbf{J}_{\mu_2}\mathbf{E}_s = \mathbf{J}_{\mu_1}\mathbf{E}_s\boldsymbol{\Psi}_\mu \tag{2.33c}$$

$$\mathbf{J}_{\nu_2}\mathbf{E}_s = \mathbf{J}_{\nu_1}\mathbf{E}_s\boldsymbol{\Psi}_\nu \tag{2.33d}$$

Similarly to the 1D scenario (eqn. (2.26)), one can find the estimated angles of arrival by extracting the diagonals of $\boldsymbol{\Psi}_\mu$ and $\boldsymbol{\Psi}_\nu$ respectively representing the phase shift induced by the translation in the x-direction (μ_i) and y-direction (ν_i).

$$\boldsymbol{\Phi}_\mu = \text{diag}\{e^{j\mu_0}, ..., e^{j\mu_{D-1}}\} \tag{2.34a}$$

$$\boldsymbol{\Phi}_\nu = \text{diag}\{e^{j\nu_0}, ..., e^{j\nu_{D-1}}\} \tag{2.34b}$$

These phase shifts per incoming signal, induced by the translation, can be shown to relate to the elevation and azimuth of the incoming signal by breaking equation (2.8) into it's x and y component[9].

$$\mu_i = \frac{2\pi}{\lambda}\Delta_x \sin\theta_i \cos\phi_i = \frac{2\pi}{\lambda}\Delta_x\mu_i' \tag{2.35a}$$

$$\nu_i = \frac{2\pi}{\lambda}\Delta_y \sin\theta_i \sin\phi_i = \frac{2\pi}{\lambda}\Delta_y\nu_i' \tag{2.35b}$$

where θ_i and ϕ_i are the elevation and azimuth of the incoming signal respectively. These values can easily be extracted from the phase shifts μ_i and ν_i using intermediate ξ_i as shown in equation (2.36)[2].

$$\xi_i = \mu_i' + j\nu_i' = \sin\theta_i e^{j\phi_i} \rightarrow \begin{cases} \theta_i = \arcsin(||\xi_i||) \\ \phi_i = \arg(\xi_i) \end{cases} \tag{2.36}$$

Thus, in order to find the AoA of the incoming signals using the ESPRIT algorithm, one has to take the following steps:

1. Collect sample matrix $\tilde{\mathbf{X}}$ and construct autocovariance matrix $\tilde{\mathbf{R}}_{\mathbf{X}}$.
2. Obtain Signal space $\tilde{\mathbf{E}}_S$ by eigendecomposition of $\tilde{\mathbf{R}}_{\mathbf{X}}$.
3. Create selection matrices in x and y-direction ($\mathbf{J}_{\mu_1}$, $\mathbf{J}_{\mu_2}$, $\mathbf{J}_{\nu_1}$ and $\mathbf{J}_{\nu_2}$)
4. Solve invariance equation $\mathbf{J}_{\mu_2}\mathbf{E}_s = \mathbf{J}_{\mu_1}\mathbf{E}_s\mathbf{\Psi}_\mu$ and $\mathbf{J}_{\nu_2}\mathbf{E}_s = \mathbf{J}_{\nu_1}\mathbf{E}_s\mathbf{\Psi}_\nu$ for $\mathbf{\Psi}_\mu$ and $\mathbf{\Psi}_\nu$. (Eqn. (2.33))
5. Find $\mathbf{\Phi}_\mu$ and $\mathbf{\Phi}_\nu$ by means of eigendecomposition of $\mathbf{\Psi}_\mu$ and $\mathbf{\Psi}_\nu$ respectively. (Eqn. (2.30))
6. Find AoA θ_i and ϕ_i from the diagonal of $\mathbf{\Psi}_\mu$ and $\mathbf{\Psi}_\nu$. (Eqn. (2.36))

Note that unlike the MUSIC algorithm, the ESPRIT algorithm does not require an extra algorithm to implement a peaksearch, but instead gives a determined results. This is one of the major advantages of this algorithm in respect to MUSIC. However, an ambiguity that is present in this algorithm is the method of solving the invariance equation (as mentioned earlier).

Forward-Backward Averaging

Before applying the algorithms, the received sample data can be pre-processed in order to increase performance. One of the most common problems introduced in the eigenspace algorithms is when received signals appear to be coherent[2]. When coherent signals are received by the the array, the autocorrelation matrix will reflect this, leading to 'false' values in the eigen-decomposition causing the algorithms to fail. One of the more common methods to reduce this effect is so called Forward-backward averaging. By exploiting the fact that the antenna array is centro-symmetric one can take the average of the correlation matrix and a 'backward' correlation matrix, reducing coherency between the samples of coherent signals[14]. This operation is possible due to the fact that the steering matrix can be reversed and conjugated and still remain the same (centro-symmetry)[2].

Let $\mathbf{J}_M$ be an $M \times M$ exchange matrix (flipped identity matrix). Then the backward autocorrelation can be calculated from the regular autocorrelation matrix as:

$$\mathbf{R}_{xback} = \mathbf{J}_M \mathbf{R}_x{}^* \mathbf{J}_M \tag{2.37}$$

where the * operator denotes the complex conjugate. The forward-backward averaged auto-correlation can be calculated as follows:

$$\mathbf{R}_{xfb} = \frac{1}{2}(\mathbf{R}_x + \mathbf{R}_{xback}) = \frac{1}{2}(\mathbf{R}_x + \mathbf{J}_M \mathbf{R}_x{}^* \mathbf{J}_M) \tag{2.38}$$

Equation (2.38) can also be written using the so called 'extended data matrix' [2]:

$$\mathbf{Z} = [\mathbf{X} \quad \mathbf{J}_M \mathbf{X} \mathbf{J}_N] \tag{2.39}$$

Then the forward-backward averaged autocovariance matrix can be calculated as[2]:

$$\tilde{\mathbf{R}_{xfb}} = \frac{1}{2}\mathbf{Z}\mathbf{Z}^H = \frac{1}{2}(\mathbf{X}\mathbf{X}^H + \mathbf{J}_M \mathbf{X}\mathbf{X}^{H^*} \mathbf{J}_M) = \frac{1}{2}(\mathbf{R}_x + \mathbf{J}_M \mathbf{R}_x{}^* \mathbf{J}_M) \tag{2.40}$$

Real-Value transformation

Computation of both MUSIC and ESPRIT requires decomposition of complex valued matrices. This is a much greater computational burden than using exclusively real valued matrices. A method of making the matrices real-valued can provide further improvement to the algorithms[7].

The complex matrix that is decomposed in these algorithms is the sample autocovariance matrix. A method to make these values real-valued is by exploiting the fact that the antenna-array is centro-symmetric. This transformation is a so called unitary transformation, using a well defined unitary matrix $\mathbf{Q}$, one can find the real valued centro-symmetric matrix from a complex centro-symmetric matrix as follows[2]:

$$\mathbf{G}_{\text{real}} = \mathbf{Q}_p{}^H \mathbf{G}_{\text{complex}} \mathbf{Q}_p \tag{2.41}$$

Where Q_M is an $M \times M$ unitary matrix. The definition for when M is an even number is found in equation (2.42a), and for M is odd in equation (2.42b).[14]:

$$\mathbf{Q}_M = \frac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{I} & j\mathbf{I} \\ \mathbf{J} & -j\mathbf{J} \end{bmatrix} \tag{2.42a}$$

$$\mathbf{Q}_M = \frac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{I} & 0 & j\mathbf{I} \\ 0 & j\sqrt{2} & 0 \\ \mathbf{J} & 0 & -j\mathbf{J} \end{bmatrix} \tag{2.42b}$$

Where $\mathbf{I}$ is the identity matrix and $\mathbf{J}$ the exchange matrix.

This unitary transformation can be applied to the sample matrix of the algorithms, due to the samples being centro-symmetric (a Uniform-Rectangular-Array is centro-symmetric). When this unitary transformaiton is applied to the forward-backward averaged autocovariance matrix, one can derive an equation to do both unitary transformation and forward-backward averaging simultaneously [2].

$$\mathbf{R}_{xfb_{\text{real}}} = \mathbf{Q}_M{}^H \frac{1}{2}(\mathbf{R}_x + \mathbf{R}_{xback})\mathbf{Q}_M = \frac{1}{2}(\mathbf{Q}_M{}^H \mathbf{R}_x \mathbf{Q}_M + \mathbf{Q}_M{}^{H^*} \mathbf{R}_x{}^* \mathbf{Q}_M{}^H) = \text{Real}\left\{\mathbf{Q}_M{}^H \mathbf{R}_x \mathbf{Q}_M\right\} \tag{2.43}$$

Changing the sample matrix to a real-valued sample matrix does not reduce or increase the dimensions of the martices but does make sure they're real-valued troughout the algorithms, leading to a reduction of the computational complexity (mostly in ESPRIT)[7]. Besides a reduction in computational complexity[7], it has been shown that by implementing this unitary transformation, the ESPRIT algorithm's AoA estimates also improve significantly, due to an improvement in the underlying signal subspace estimates[7].

After unitary transformation of the sample matrix, the algorithms slightly change their implementation. Similarly to equation (2.17), the real-valued autocovariance splits into the signal-space and noise-space ($\mathbf{U}_S$ and $\mathbf{U}_N$ respectively). However, one should realize that the eigen-spaces of the autocovariance of the samples are different than the eigen-spaces of the autocovariance of the real-valued samples ($\mathbf{U}_S \neq \mathbf{E}_S$). In order to account for this change in autocovariance matrix, the algorithms should be updated as follows.

The modification of the MUSIC algorithms are exclusively in the MUSIC spatial spectrum. Since the relationship between the regular autocovariance matrix and the real-valued autocovariance matrix is simply the defined unitary matrix[17] as shown in equation (2.44).

$$\mathbf{E}_S = \mathbf{Q}_M \mathbf{U}_S \tag{2.44}$$

Meaning that a small change in the MUSIC spatial spectrum equation (eqn. (2.24)) can account for the real-valued transformation[17] as seen in equation (2.45).

$$P_{\text{MUSIC}} = \frac{1}{\tilde{\mathbf{A}}(\theta,\phi)\tilde{\mathbf{E}}_N \tilde{\mathbf{E}}_N^H \tilde{\mathbf{A}}^H(\theta,\phi)} = \frac{1}{\tilde{\mathbf{A}}(\theta,\phi)\mathbf{Q}_M \tilde{\mathbf{U}}_N \tilde{\mathbf{U}}_N^H \mathbf{Q}_M^H \tilde{\mathbf{A}}^H(\theta,\phi)} \tag{2.45}$$

The modifications for the ESPRIT algorithm to support real-value transformation are more significant. It's possible to transform all equations to real-valued equations. The real-valued steering matrices are transformed using the unitary matrix $\mathbf{Q}_M$ as given in equation (2.46).[7]

$$\mathbf{B} = \mathbf{Q}_M^H \mathbf{A} \tag{2.46}$$

Where the real-valued steering matrix $\mathbf{B}$ can be used to setup a real-valued invariance equation using transformed selection matrices $\mathbf{K}$, similarly to the invariance equations of the complex steering matrix[7], this can be observed in equation (2.47).

$$\mathbf{K}_{\mu_1} \mathbf{B} \mathbf{\Omega}_\mu = \mathbf{K}_{\mu_2} \mathbf{B} \tag{2.47a}$$

$$\mathbf{K}_{\nu_1}\mathbf{B}\boldsymbol{\Omega}_{\nu} = \mathbf{K}_{\nu_2}\mathbf{B} \tag{2.47b}$$

The transformed selection matrices **K** are derived from the regular selection matrices **J** according to equations (2.48)[28].

$$\mathbf{K}_{\mu_1} = 2\mathrm{Re}\{\mathbf{Q}_{m_x}^H\mathbf{K}_{\mu_2}\mathbf{Q}_M\} \qquad \mathbf{K}_{\mu_2} = 2\mathrm{Im}\{\mathbf{Q}_{m_x}^H\mathbf{K}_{\mu_2}\mathbf{Q}_M\} \tag{2.48a}$$

$$\mathbf{K}_{\nu_1} = 2\mathrm{Re}\{\mathbf{Q}_{m_y}^H\mathbf{K}_{\nu_2}\mathbf{Q}_M\} \qquad \mathbf{K}_{\nu_2} = 2\mathrm{Im}\{\mathbf{Q}_{m_y}^H\mathbf{K}_{\nu_2}\mathbf{Q}_M\} \tag{2.48b}$$

consequently, the invariance equations of the signal space (equations (2.33)) can also be transformed to real-valued invariance equations similarly to the real-valued steering matrix invariance equations (equations (2.47)). This invariance is given in equations (2.49)

$$\mathbf{K}_{\mu_1}\mathbf{U}_S\boldsymbol{\Upsilon}_{\mu} = \mathbf{K}_{\mu_2}\mathbf{U}_S \tag{2.49a}$$

$$\mathbf{K}_{\nu_1}\mathbf{U}_S\boldsymbol{\Upsilon}_{\nu} = \mathbf{K}_{\nu_2}\mathbf{U}_S \tag{2.49b}$$

Where the relationship between $\boldsymbol{\Upsilon}$ and $\boldsymbol{\Omega}$ is one of eigendecomposition (similarly to equation (2.30))[28].

$$\boldsymbol{\Upsilon}_{\mu} = \mathbf{T}\boldsymbol{\Omega}_{\mu}\mathbf{T}^{-1} \quad \boldsymbol{\Upsilon}_{\nu} = \mathbf{T}\boldsymbol{\Omega}_{\nu}\mathbf{T}^{-1} \tag{2.50}$$

The phase-shifts can be observed in the diagonal of $\boldsymbol{\Omega}$ similar to how the phase-shifts are observed in the diagonal of $\boldsymbol{\Phi}$ in the complex domain (Equation (2.34))[28].

$$\boldsymbol{\Omega}_{\mu} = \mathrm{diag}\{\tan(\mu_0/2), ..., \tan(j\mu_{D-1}/2)\} \tag{2.51a}$$

$$\boldsymbol{\Omega}_{\nu} = \mathrm{diag}\{\tan(j\nu_0/2), ..., \tan(j\nu_{D-1}/2)\} \tag{2.51b}$$

Where μ_i and ν_i can be used to find the elevation and azimuth according to equation (2.36). A drawback to this algorithm modification is by addition of the 'tan' function one can have difficulty pairing the found elevation and azimuth together when $\boldsymbol{\Omega}_{\mu}$ and $\boldsymbol{\Omega}_{\nu}$ are calculated independently[28]. A method to adapt for this problem is using so-called "Automatic Pairing"[28]. This technique makes use of the fact that $\boldsymbol{\Upsilon}_{\mu}$ and $\boldsymbol{\Upsilon}_{\nu}$ should have the same set of eigenvectors (Equation (2.50)). Thus, $\boldsymbol{\Omega}_{\mu}$ and $\boldsymbol{\Omega}_{\nu}$ can be solved simultaneously by creating a complex matrix from $\boldsymbol{\Upsilon}_{\mu}$ and $\boldsymbol{\Upsilon}_{\nu}$ as shown in equation (2.52).

$$\boldsymbol{\Upsilon}_{\mu} + j\boldsymbol{\Upsilon}_{\nu} = \mathbf{T}(\boldsymbol{\Omega}_{\mu} + j\boldsymbol{\Omega}_{\nu})\mathbf{T}^{-1} \tag{2.52}$$

Where μ_i and ν_i can be extracted from the diagonal of $\boldsymbol{\Upsilon}_{\mu} + j\boldsymbol{\Upsilon}_{\nu}$[28].

$$\boldsymbol{\Upsilon}_{\mu} + j\boldsymbol{\Upsilon}_{\nu} = \mathrm{diag}\{\lambda_i, ..., \lambda_{D_1}\} \rightarrow \begin{cases} \mu_i = 2\arctan(\mathrm{Re}\{\lambda_i\}) \\ \nu_i = 2\arctan(\mathrm{Re}\{\lambda_i\}) \end{cases} \tag{2.53}$$

Finally, the modified ESPRIT algorithm (Unitary ESPRIT) can be described by the following steps [28]

1. Collect sample matrix $\tilde{\mathbf{X}}$ and construct autocovariance matrix $\tilde{\mathbf{R}}_{\mathbf{X}}$.
2. Unitary transformation and forward-backward averaging of the autocovariance matrix $\tilde{\mathbf{R}}_{\mathbf{X}}$ (Eqn. (2.43))
3. Obtain Signal space $\tilde{\mathbf{U}}_S$ by eigendecomposition of $\mathbf{R}_{x\text{fb-real}}$.
4. Setup transformed selection matrices in x and y-direction ($\mathbf{K}_{\mu_1}$, $\mathbf{K}_{\mu_2}$, $\mathbf{K}_{\nu_1}$ and $\mathbf{K}_{\nu_2}$)
5. Solve invariance equation $\mathbf{K}_{\mu_1}\mathbf{U}_S\boldsymbol{\Upsilon}_{\mu} = \mathbf{K}_{\mu_2}\mathbf{U}_S$ and $\mathbf{K}_{\nu_1}\mathbf{U}_S\boldsymbol{\Upsilon}_{\nu} = \mathbf{K}_{\nu_2}\mathbf{U}_S$ for $\boldsymbol{\Upsilon}_{\mu}$ and $\boldsymbol{\Upsilon}_{\nu}$. (Eqn. (2.49))
6. Find $\boldsymbol{\Omega}_{\mu} + j\boldsymbol{\Omega}_{\nu}$ by means of eigendecomposition of $\boldsymbol{\Upsilon}_{\mu} + j\boldsymbol{\Upsilon}_{\nu}$ respectively. (Eqn. (2.52))
7. Find AoA θ_i and ϕ_i from the diagonal of $\boldsymbol{\Omega}_{\mu} + j\boldsymbol{\Omega}_{\nu}$. (Eqn. (2.53) and (2.36))

3

Simulation of Geolocation Antenna Array

The following section is about considerations and implementation of a simulation of the models described in the previous section. Before running the simulation of a model, a programming language has to be selected and the functions have to be implemented and created.

3.1. Overview of Simulation Chain

In order to maximize the flexibility and to make the simulation 'future-proof' the following considerations are to be taken into account:

- **'Architecture' of the functions**: The overall architecture documentation shall be complete but not too complex, in order to allow for future work to either extend or adapt the software.

- **Programming Language**: The programming language and software used, has to be considered future-proof. (Programming language shall be recent and high-level, but also licensing costs have to be taken into account).

- **Decoupling of functions**: The software shall decouple functionality in order to allow it to be a modular design, in-case future projects require reuse of specific functionality or addition of modules.

To comply with these considerations, the programming language selected should be a modern object-oriented language, particularly to allow for proper architecture and decoupling. Another consideration is licensing and platform dependency. Licensing costs should be reduced in order to make the software portable across multiple computer systems without the need to move a license. The software should preferably be multi-platform in order to move the software across multiple computer systems without the need for specific hardware or operating system. After weighing different options, the programming language selected is Python, due to it being very accessible between multiple platforms without the need of different compilers and having no licensing costs since it is open-source.

The architecture of the simulation is done by wrapping functionality and parameters into objects. The main object is the estimator-object, which holds the emitter-objects and the satellite-object. The satellite-object holds the AoA-estimator object and the antenna array parameters. The architecture of these objects can be found in figure 3.1. The implementation of the AoA Algorithms themselves is kept in a separate library, due to the fact that these algorithms do not distinguish between 'simulation' and actual implementation, meaning that this library could be used in actual hardware implementations in future works (decoupling of functionality).

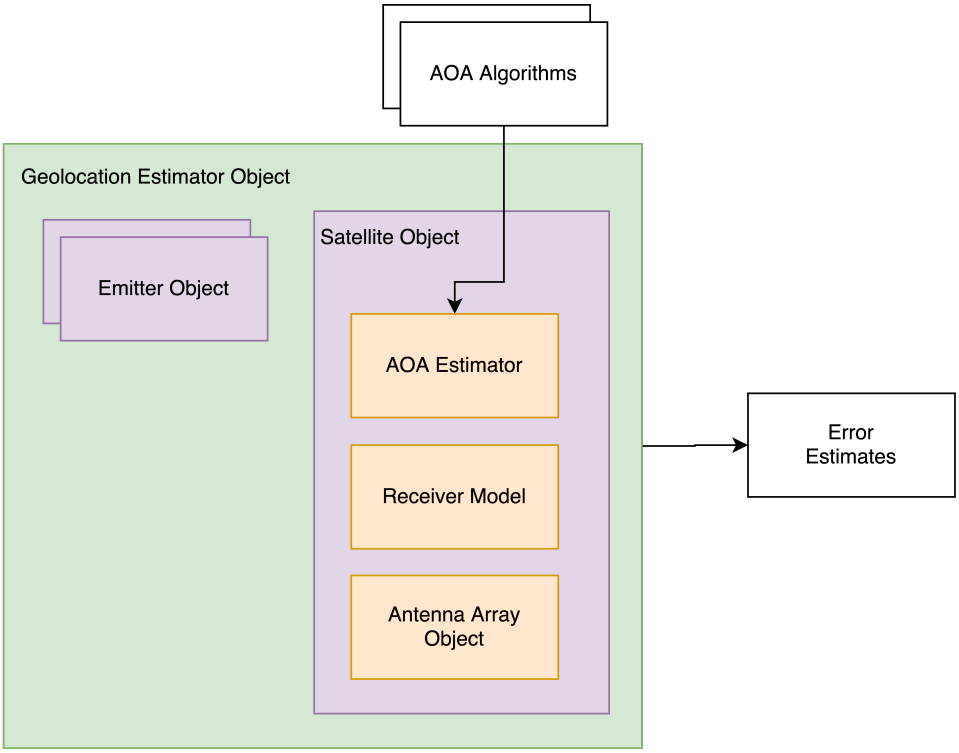


Figure 3.1: Schematic overview of software objects

3.2. Implementation

This part of the Thesis focusses on the implementation of the AOA - Algorithms as discussed per section 2.2.3.

Receive Model
The receive model is implemented according to section 2.2.3 and requires as input the parameters and dimensions of the antenna array (coordinates of elements in URA), the noise figure of the received signal, the amount of samples taken and lastly the parameters of the emitter.

Algorithm 1: Calculate received samples of Emitter

ReceiveSignal $(M_x, M_y, Lats, Lons, \lambda, N, \sigma)$
- **inputs :** Received Noise figure σ, amount of samples N, Antenna Array Geometry (M_x, M_y), Emitter Parameters (Latitudes, Longitudes, Wavelength λ)
- **outputs:** Array of Received Samples X.
- `Convert target longitudes, latitudes to Angles-of-Arrival:` (θ_i, ϕ_i);
- `Calculate steering matrix:` $\mathbf{A}(M_y, M_x, \lambda, \theta, \phi)$;
- `Generate N random signals:` $\mathbf{S}$;
- `Create received data matrix:` $\mathbf{X}_0 = \mathbf{AS}$;
- `Add Gaussian noise to received data:` $\mathbf{X} = \mathbf{X}_0 + \mathbf{N}(0, \sigma)$;

ESPRIT
The ESPRIT Algorithm[21][28] is implemented using the steps given last section. The required input is the geometry of the antenna array (URA), the received samples $\mathbf{X}$, the wavelength of the received signal λ and lastly whether to apply unitary transformation or not.

Algorithm 2: ESPRIT Algorithm

CalculateESPRIT ($\boldsymbol{M_x}, \boldsymbol{M_y}, \boldsymbol{X}, \lambda, \textit{unitaryMode}$)

inputs : Samples $\mathbf{X}$, Antenna Array Geometry ($\mathbf{M}_x, \mathbf{M}_y$) Emitter Parameter ($\lambda$), Unitary-Mode True/False

outputs: Estimated Angles-of-Arrival $\boldsymbol{\theta}, \boldsymbol{\phi}$

Calculate autocovariance Matrix: $\mathbf{R}_x = \mathbf{X}\mathbf{X}^H$;

if *unitaryMode == True* **then**

Perform Unitary Transformation: $T = \text{Real}\left\{\mathbf{Q}_M{}^H \mathbf{R}_x \mathbf{Q}_M\right\}$;

Perform Singular Value Decomposition to obtain the Signal Eigenspace: $\mathbf{U}_S$;

Calculate transformed selection matrices: $\mathbf{K}_\mu, \mathbf{K}_\nu$;

Solve invariance equations: $\mathbf{K}_{\mu_1}\mathbf{U}_S\boldsymbol{\Upsilon}_\mu = \mathbf{K}_{\mu_2}\mathbf{U}_S, \mathbf{K}_{\nu_1}\mathbf{U}_S\boldsymbol{\Upsilon}_\nu = \mathbf{K}_{\nu_2}\mathbf{U}_S$;

Perform Singular Value Decomposition to obtain intermediate: $\boldsymbol{\Omega}_\mu + j\boldsymbol{\Omega}_\nu$;

Extract phase shift terms from diagonal of intermediate: μ_i, ν_i;

else

Perform Singular Value Decomposition to obtain the Signal Eigenspace: $\mathbf{E}_S$;

Create selection Matrices that are maximum overlapping: $\mathbf{J}_\mu, \mathbf{J}_\nu$;

Solve invariance equations: $\mathbf{J}_{\mu_2}\mathbf{E}_s = \mathbf{J}_{\mu_1}\mathbf{E}_s\boldsymbol{\Psi}_\mu, \mathbf{J}_{\nu_2}\mathbf{E}_s = \mathbf{J}_{\nu_1}\mathbf{E}_s\boldsymbol{\Psi}_\nu$;

Perform Singular Value Decomposition to obtain intermediates: $\boldsymbol{\Phi}_\mu, \boldsymbol{\Phi}_\nu$;

Extract phase shift terms from diagonals of intermediates: μ_i, ν_i;

Create intermediate value from phase-shift terms: $\xi_i = \mu_i + j\nu_i$;

Get Elevation Angles: $\theta_i = \arcsin|\xi_i|$;

Get Azimuth Angles: $\theta_i = \angle\xi_i$;

MUSIC

The MUSIC Algorithm[21] requires as input the geometry of the antenna array (URA), the received samples $\mathbf{X}$, the wavelength of the received signal λ and lastly whether to apply unitary transformation or not.

Algorithm 3: MUSIC Algorithm

CalculateMUSIC ($\boldsymbol{M_x}, \boldsymbol{M_y}, \boldsymbol{X}, \lambda, \textit{unitaryMode}$)

inputs : Samples $\mathbf{X}$, Antenna Array Geomerty ($\mathbf{M}_x, \mathbf{M}_y$) Emitter Parameter ($\lambda$), Unitary-Mode True/False

outputs: Array of Received Samples X

Calculate autocovariance matrix: $\mathbf{R}_x = \mathbf{X}\mathbf{X}^H$;

if *Unitary-Mode == True* **then**

Perform Unitary Transformation: $T = \text{Real}\left\{\mathbf{Q}_M{}^H \mathbf{R}_x \mathbf{Q}_M\right\}$;

Perform Singular Value Decomposition to obtain the noise Eigenspace: $\mathbf{U}_N$;

Search for peaks in the MUSIC spatial spectrum:

$P_{MUSIC}(\theta,\phi) = (\tilde{\mathbf{A}}(\theta,\phi)\mathbf{Q}_M\tilde{\mathbf{U}}_N\tilde{\mathbf{U}}_N^H\mathbf{Q}_M^H\tilde{\mathbf{A}}^H(\theta,\phi))^{-1}$;

else

Perform Singular Value Decomposition to obtain the noise Eigenspace: $\mathbf{E}_N$;

Search for peaks in the MUSIC spatial spectrum:

$P_{MUSIC}(\theta,\phi) = (\tilde{\mathbf{A}}(\theta,\phi)\tilde{\mathbf{E}}_N\tilde{\mathbf{E}}_N^H\tilde{\mathbf{A}}^H(\theta,\phi))^{-1}$;

Match peaks with received Angles-of-Arrival: (θ_i, ϕ_i);

As mentioned earlier and observable from the MUSIC algorithm (Algorithm 3), the MUSIC algorithm requires a 2D peak-search. One could simply use a 'brute-force' approach for the peak-search, trying every possible combination of AoA to find all local maxima. However, in order to reduce the computational intensity a simple iterative neighbour checking peak-search algorithm has been implemented[3]. The algorithm is recursively called, every-time performing the peak-search with a smaller step-size. Note that the final step-size limits the amount of decimals of the fractional part of the found angles. An example of this iterative approach can be seen in figure 3.2. In this example the step-size of the search is reduced from $10^\circ \rightarrow 1^\circ \rightarrow 0.1^\circ$. The peak-search algorithm can be observed in algorithm 4. The selected final search-depth for the implementation has been decided on 4, meaning the peak search will obtain an estimation of peaks with a step-size of 10^{-4}.

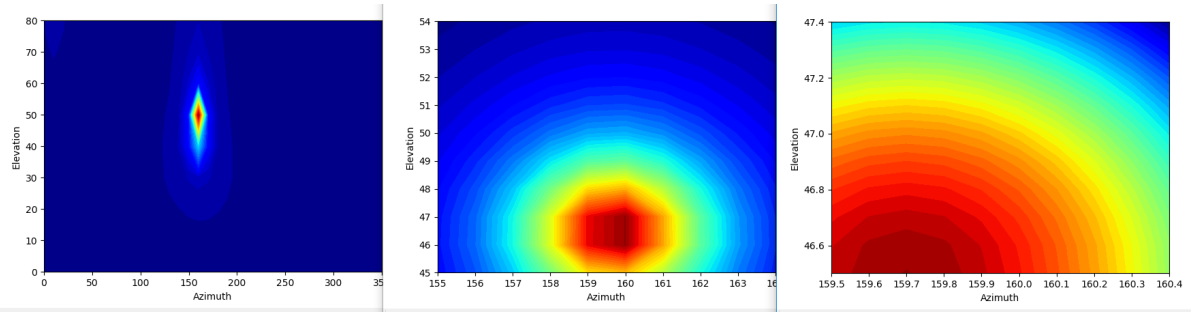


Figure 3.2: Iterative smaller search-space and step-size of Peak search algorithm step size of 10°, 1° and .1° from left to right

Algorithm 4: Peak search algorithm

SearchPeaks *($\boldsymbol{M}_x, \boldsymbol{M}_y, \boldsymbol{E}_N / \boldsymbol{U}_N, d, d_0, \lambda$, boundaries)*

inputs : Antenna Array Geometry ($\mathbf{M}_x, \mathbf{M}_y$) and Emitter wavelength (λ) in order to create the steering Matrix, Noise Space $\mathbf{E}_N / \mathbf{U}_N$, the initial boundaries of search-space (default = (0-90,0-360)), final search depth (d) and current search depth (d_0) (default = -1)

outputs: Peaks in P_{MUSIC}: ($\boldsymbol{\theta}, \boldsymbol{\phi}$)

Calculate step-size: $d_{\text{step}} = 10^{-d_0}$;

Calculate MUSIC spatial spectrum within boundaries using step-size:
$P_{MUSIC}(\text{bounds}, \mathbf{E}_N / \mathbf{U}_N, \mathbf{M}_x, \mathbf{M}_y, \lambda)$;

foreach $i, j \in$ *search-space* **do**

  Check if Peak by checking neighbours:
  $P(i,j) > P(i+1,j), P(i,j) > P(i-1,j), P(i,j) > P(i,j+1), P(i,j) > P(i,j-1)$;

  if $P(i,j) ==$ *Peak* **then**

    Calculate Elevation angle of peak: $\theta_k = \text{lower-bound} + i \cdot d_{\text{step}}$;

    Calculate Azimuth angle of peak: $\phi_k = \text{lower-bound} + j \cdot d_{\text{step}}$;

if $d_0 >= d$ **then**

  return found Angles-of-Arrival: $\boldsymbol{\theta}, \boldsymbol{\phi}$;

else

  foreach $(\theta_k, \phi_k \in \boldsymbol{\theta}, \boldsymbol{\phi})$ **do**

    Set new elevation boundary: $[\theta_k - d_{\text{step}}/2; \theta_k + d_{\text{step}}/2]$;

    Set new azimuth boundary: $[\theta_k - d_{\text{step}}/2; \theta_k + d_{\text{step}}/2]$;

    $\boldsymbol{\theta}, \boldsymbol{\phi} =$ **SearchPeaks**$(\mathbf{M}_x, \mathbf{M}_y, \mathbf{E}_N / \mathbf{U}_N, d, d_0 + 1, \lambda,$ new boundaries$)$

  return found Angles-of-Arrival: $\boldsymbol{\theta}, \boldsymbol{\phi}$;

Conversions between Longitude/Latitude to Angles-of-Arrival

Another function required for geolocation is to go from longitudes and latitudes to AoA and vice-versa. In order to do this effectively, a decision has to be made regarding reference frames. Assumed is a spherical earth and due to the nature of this project mostly being interested in locations on the earth surface and mostly dealing with longitudes and latitudes, the default frame selected is the so called Earth-Centered, Earth-Fixed reference frame (ECEF)[4]. This ECEF frame has an x-axis pointing to the Greenwich Meridian (Longitude $\lambda = 0°$), the earths equatorial plane lies entirely in the XY-plane and the x-axis points to the North-Pole. The ECEF frame can be observed in figure 3.3. Consequently, a frame of the spacecraft in which the measurements take place has to be defined. This secondary frame can be defined as a so-called North-East-Down (NED) frame, as observed in figure 3.4.

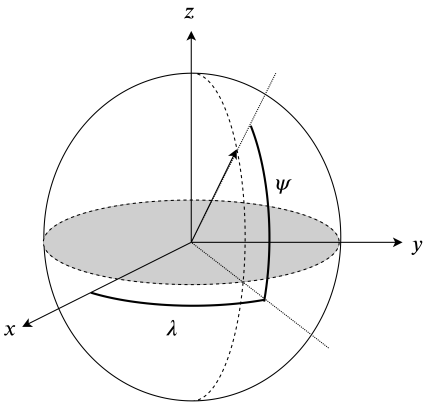


Figure 3.3: Earth-Centered Earth-Fixed frame

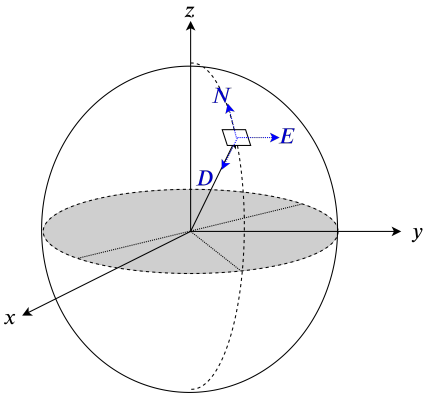


Figure 3.4: Nort-East-Down frame

An emitter on earth surface can now be placed in the ECEF frame, either as spherical coordinates or Cartesian coordinates. Due to the nature of the ECEF frame, the spherical coordinates of the emitter can simply be interchanged with longitude, latitude and radius (which is earth-radius assuming an earth-surface emitter). Using trigonometry, the location vector of emitter j can be given in Cartesian coordinates using equation (3.1):

$$\vec{\mathbf{p}}_j = R_{\text{Earth}}\cos(\lambda_j)\cos(\psi_j)\hat{\mathbf{x}} + R_{\text{Earth}}\sin(\lambda_j)\cos(\psi_j)\hat{\mathbf{y}} + R_{\text{Earth}}\sin(\psi_j)\hat{\mathbf{z}} \tag{3.1}$$

Where $\vec{\mathbf{p}}_j$ is the position vector of emitter j, R_{Earth} is the radius from the earth center ($R_{\text{earth}} = 6371$ km), λ_j is the emitter longitude and ψ_j is the emitter latitude.

The origin of the NED frame is the satellite, the location of the satellite can be described similarly to a radio emitter, namely using a longitude and latitude of the satellite projected on the Earth-surface, and a radius which can be calculated using the orbital height ($r = R_{\text{Earth}} + h$). The 'Down' vector of the NED frame can be described in the ECEF frame as the unit length inverse of the origin location vector, the 'North' vector can be calculated using trigonometry and the satellite latitude (figure 3.5). Lastly, the 'East' vector can be derived by completing the frame using the cross product. The equations to calculate the NED frame vectors in the ECEF frame can be found in equation (3.2)

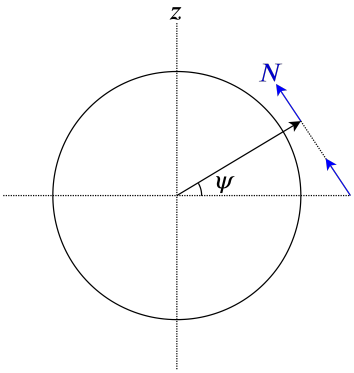


Figure 3.5: 'North' Vector in ECEF Frame

$$\hat{\mathbf{p}}_s = (R_{\text{Earth}} + h_s)\cos(\lambda_s)\cos(\psi_s)\hat{\mathbf{x}} + (R_{\text{Earth}} + h_s)\sin(\lambda_s)\cos(\psi_s)\hat{\mathbf{y}} + (R_{\text{Earth}} + h_s)\sin(\psi_s)\hat{\mathbf{z}} \tag{3.2a}$$

$$\hat{\mathbf{D}} = -\frac{\vec{\mathbf{p}}_s}{||\vec{\mathbf{p}}_s||} = -\cos(\lambda_s)\cos(\psi_s)\hat{\mathbf{x}} - \sin(\lambda_s)\cos(\psi_s)\hat{\mathbf{y}} - \sin(\psi_s)\hat{\mathbf{z}} \tag{3.2b}$$

$$\hat{\mathbf{N}} = -\cos(\lambda_s)\sin(\psi_s)\hat{\mathbf{x}} - \sin(\lambda_s)\sin(\psi_s)\hat{\mathbf{y}} + \cos(\psi_s)\hat{\mathbf{z}} \tag{3.2c}$$

$$\hat{\mathbf{E}} = \frac{\hat{\mathbf{D}} \times \hat{\mathbf{N}}}{||\hat{\mathbf{D}} \times \hat{\mathbf{N}}||} \tag{3.2d}$$

Given the geographic location of the emitter (λ_j, ψ_j), the geographic location of the satellite (λ_s, ψ_s) and the height (h_s), the Angles-of-Arrival of the incoming signal (θ_j, ϕ_j) can be found by investigating the emitter in the NED frame. The elevation of the signal (θ_j) is the angle between the vector pointing from the satellite to the emitter and the 'Down' vector. The azimuth of the signal (ϕ_j) can be defined as the angle between the 'North' Vector and the projection of the vector pointing from the satellite to the emitter projected on the 'North-East' plane. Both the angle between two vectors and a projection of a vector on a plane can be found using the dot product.

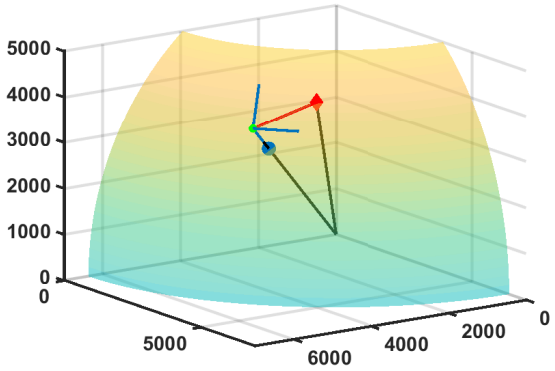


Figure 3.6: NED Frame, satellite and emitter in ECEF frame

$$\vec{\mathbf{p}}_{s,j} = \vec{\mathbf{p}}_j - \vec{\mathbf{p}}_s \tag{3.3a}$$

$$\theta_j = \arccos\left(\frac{\hat{\mathbf{D}} \cdot \vec{\mathbf{p}}_{s,j}}{||\vec{\mathbf{p}}_{s,j}||}\right) \tag{3.3b}$$

$$\phi_j = \arctan\left(\frac{\vec{\mathbf{p}}_{s,j} \cdot \hat{\mathbf{E}}}{\hat{\mathbf{p}}_{s,j} \cdot \hat{\mathbf{N}}}\right) \tag{3.3c}$$

This operation has also be done inversely. Instead of finding the respective elevation and azimuth angle from the geographic location from an emitter, one also has to find the estimated geographic location from an estimated elevation and azimuth. In figure 3.7 one can observe the estimated Angles-of-Arrival and estimated Direction vector of the Earth-Surface emitter $(\tilde{\mathbf{d}}_j)$ in the NED frame.

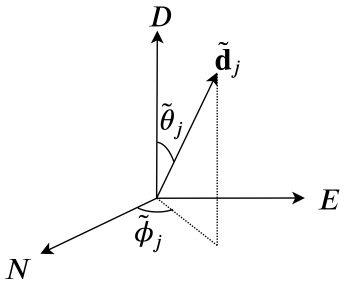


Figure 3.7: Estimated direction vector of emitter in satellite NED frame

By using trigonometry, the estimation direction $\tilde{\mathbf{d}}_j$ can be rewritten in terms of estimated AoA $(\tilde{\theta}_j, \tilde{\phi}_j)$

$$\tilde{\mathbf{d}}_j = \sin(\tilde{\theta}_j)\cos(\tilde{\phi}_j)\hat{\mathbf{N}} + \sin(\tilde{\theta}_j)\sin(\tilde{\phi}_j)\hat{\mathbf{N}} + \cos(\tilde{\theta}_j)\hat{\mathbf{D}} \tag{3.4}$$

The estimated position on the earth surface can be found by finding an intersection between a sphere (Earth) and the estimated direction vector of the emitter starting at the satellite. This Line-Sphere intersection can be extended to a quadratic form, this can be seen in equations (3.5).

$$||\vec{\mathbf{p}}_s + d\tilde{\mathbf{d}}_j|| = R_{\text{earth}}^2 \tag{3.5a}$$

$$d^2(\tilde{\mathbf{d}}_j \cdot \tilde{\mathbf{d}}_j) + 2d(\tilde{\mathbf{d}}_j \cdot \vec{\mathbf{p}}_s) + (\vec{\mathbf{p}}_s \cdot \vec{\mathbf{p}}_s) - R_{\text{earth}}^2 = 0 \tag{3.5b}$$

Solving the form of $ad^2 + bd + c = 0$ can be achieved using the quadratic formula.

$$a = \tilde{\mathbf{d}}_j \cdot \tilde{\mathbf{d}}_j = 1 \tag{3.6a}$$

$$b = 2(\tilde{\mathbf{d}}_j \cdot \vec{\mathbf{p}}_s) \tag{3.6b}$$

$$c = (\vec{\mathbf{p}}_s \cdot \vec{\mathbf{p}}_s) - R_{\text{earth}}^2 = (R_{\text{earth}}^2 + h_s)^2 - R_{\text{earth}}^2 \tag{3.6c}$$

$$d = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{3.6d}$$

Note that equations (3.6) either give:

- no solution (no possible estimated emitter for estimated direction vector)
- one solution (estimated emitter geographic location is on horizon in respect to satellite location)
- two solutions, where only the minimal solution d is valid (a solution on the surface, and another solution on the opposite side of Earth, where only the former solution is valid).

Given a solution for d using equations (3.6), the estimated coordinates for the emitter location in the ECEF frame can be found according to equation (3.7)

$$\tilde{\mathbf{p}}_j = \vec{\mathbf{p}}_s + d\tilde{\mathbf{d}}_j \tag{3.7}$$

The estimated geographic location of the emitter can be converted to estimated longitude and latitude using trigonometry.

$$\tilde{\lambda}_j = \arctan\left(\frac{\tilde{p}_{j_y}}{\tilde{p}_{j_x}}\right) \qquad \tilde{\psi}_j = \arcsin(\tilde{p}_{j_z}) \tag{3.8}$$

Linking the Angle-of-Arrival algorithms to the geographic estimation algorithms leads to a full end-to-end chain as described in figure 2.2. An example of this can be seen in figure 3.8, where the whole chain is simulated for both MUSIC and ESPRIT and for two different values of noise.

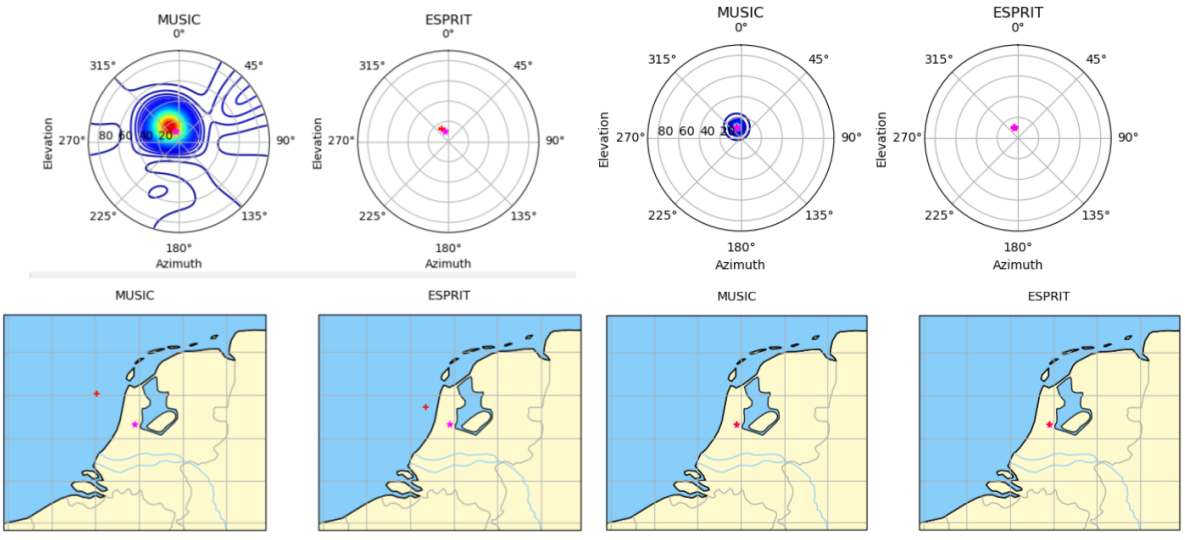


Figure 3.8: Example of AOA-Chain simulation for both MUSIC and ESPRIT for relatively high noise (0 dB SNR) and low noise (10 dB SNR)

A function has to be obtained to calculate the error between the geographic location of the estimation and the actual emitter location. This error can be found by calculating the great-circle distance between the two points using the so-called Haversine formula. The error between two points on the Earth surface can be found according to equation (3.9) [19]

$$e = 2R_{\text{Earth}} \arcsin\left(\sqrt{\text{Hav}(\psi_2 - \psi_1) + \cos(\psi_1)\cos(\psi_2)\text{Hav}(\lambda_2 - \lambda_1)}\right) \tag{3.9}$$

Where 'Hav(·)' is the Haversine equation given by equation (3.10)

$$y = \sin^2(\theta/2) \tag{3.10}$$

3.3. Simulation Verification

In order to resume analysis with the obtained simulation software, the software should be evaluated for correctness. The most critical sections of software, that should be focused on during verification is:

- Longitude and Latitude to Angle-of-Arrival Conversion.
- Angle-of-Arrival to Longitude and Latitude Conversion.
- (Unitary) ESPRIT Algorithm
- (Unitary) MUSIC Algorithm

Longitude, Latitude and Angle-of-Arrival Conversions: In order to verify the functionality of the geographical coordinates to Angle-of-Arrival conversion, one can first do a basic test, creating an output for an arbitrary input and observe if the obtained result is logical. Consequently, special cases with known outputs can be introduced in order to test the correctness, for example if the satellite and target have the same geographical coordinates, the elevation should be $\theta = 0°$, These special cases can be shown in table 3.1. The cases has been verified for geographical to elevation/azimuth conversion for a satellite height of 50km and the verification results can be seen in table 3.2. One can observe that all verification tests are passed.

Test Nr	**Satellite** (λ_s, ψ_s)	**Target** (λ_t, ψ_t)	**Angle-of-Arrival** (θ, ϕ)
1	(λ_s, ψ_s)	$(\lambda_t, \psi_t) = (\lambda_s, \psi_s)$	$(\theta, \phi) = (0°, -)$
2	(λ_s, ψ_s)	$(\lambda_t, \psi_t) = (\lambda_s, \psi_t > \psi_s)$	$(\theta, \phi) = (> 0°, 0°)$
3	(λ_s, ψ_s)	$(\lambda_t, \psi_t) = (\lambda_s, \psi_t < \psi_s)$	$(\theta, \phi) = (> 0°, 180°)$
4	(λ_s, ψ_s)	$(\lambda_t, \psi_t) = (\lambda_t > \lambda_s, \psi_s)$	$(\theta, \phi) = (> 0°, \approx 90°)$
5	(λ_s, ψ_s)	$(\lambda_t, \psi_t) = (\lambda_t < \lambda_s, \psi_s)$	$(\theta, \phi) = (> 0°, \approx 270°)$

Table 3.1: Verification cases for geographical coordinates to angle-of-arrival conversion

Test Nr	**Satellite** (λ_s, ψ_s)	**Target** (λ_t, ψ_t)	**Angle-of-Arrival** (θ, ϕ)
1	$(50°, 50°)$	$(\lambda_t, \psi_t) = (50°, 50°)$	$(\theta, \phi) = (0.00°, -69.09°)$
2	$(50°, 50°)$	$(\lambda_t, \psi_t) = (50°, 70°)$	$(\theta, \phi) = (65.69°, 0.00°)$
3	$(50°, 50°)$	$(\lambda_t, \psi_t) = (50°, 30°)$	$(\theta, \phi) = (65.69°, 180.00°)$
4	$(50°, 50°)$	$(\lambda_t, \psi_t) = (55°, 50°)$	$(\theta, \phi) = (30.35°, 88.08°)$
5	$(50°, 50°)$	$(\lambda_t, \psi_t) = (45°, 50°)$	$(\theta, \phi) = (30.35°, -88.08°)$

Table 3.2: Verification cases for geographical coordinates to angle-of-arrival conversion

However, the obtained verification cases can however also be verify the inverse conversion, by defining the input AoA and converting those to target coordinates. The verification results of this scenario can be seen in table 3.3.

Test Nr	**Satellite** (λ_s, ψ_s)	**Angle-of-Arrival** (θ, ϕ)	**Target** (λ_t, ψ_t)
1	$(50°, 50°)$	$(\theta, \phi) = (0°, 0°)$	$(\lambda_t, \psi_t) = (50.00°, 50.00°)$
2	$(50°, 50°)$	$(\theta, \phi) = (20°, 0°)$	$(\lambda_t, \psi_t) = (50.00°, 51.98°)$
3	$(50°, 50°)$	$(\theta, \phi) = (20°, 180°)$	$(\lambda_t, \psi_t) = (50.00°, 48.02°)$
4	$(50°, 50°)$	$(\theta, \phi) = (20°, 90°)$	$(\lambda_t, \psi_t) = (53.07°, 49.96°)$
5	$(50°, 50°)$	$(\theta, \phi) = (20°, 270°)$	$(\lambda_t, \psi_t) = (46.93°, 49.96°)$

Table 3.3: Verification cases for angle-of-arrival conversion to geographical coordinates

Lastly, one can check the conversion algorithms for an error or offset by coupling them directly to each-other. For example, one takes an arbitrary emitter latitude and longitude, converts this to angles-of-arrival, converts this back into estimated emitter latitude and longitude and finally checks the error in between them. When taking the satellite and emitter parameters similar to the verification tests $((\lambda_s, \psi_s) = (50°, 50°), (\lambda_t, \psi_t) = (60°, 40°))$ and converting them back and forth leads to an error of $((e_{\lambda_t}, e_{\psi_t}) = (1.42 \cdot 10^{-14}°, -7.11 \cdot 10^{-15}°))$, which is negligibly small and can be accounted to rounding errors.

ESPRIT Algorithm

The ESPRIT algorithms in the software are more complex than the conversion algorithms. Hence a more extensive approach has to be used to verify this piece of simulation. Initially one can check if the Algorithm solves to the correct solution if there is no noise on the signal. This can easily be checked by running the simulation with zero noise. Doing 100 runs of a noiseless scenario leads to the results found in figure 3.9 and 3.10

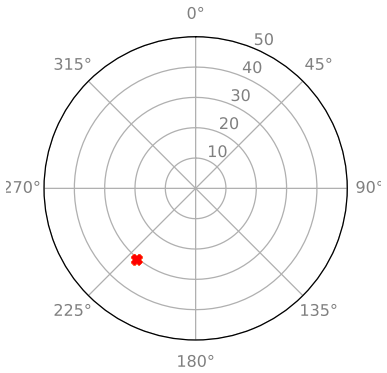


Figure 3.9: Polar plot of noiseless ESPRIT simulation

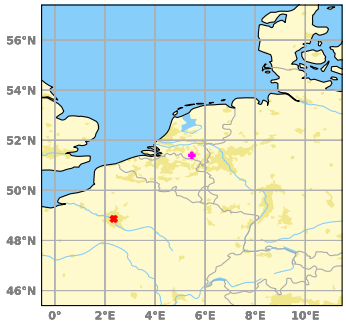


Figure 3.10: Geographic plot of noiseless ESPRIT simulation

Another verification case is that when additive white Gaussian noise is introduced to the model, the obtained estimation should form an ellipse around the real solution. This is due to the noise propagating as an uncertainty in the results leading to a distribution around the noiseless solution. This has been simulated and can be observed in figure 3.11 and 3.12

Finally, the simulation can be compared to similar already existing simulations and see if the results align with each-other. However, this step has shown to be most difficult. Since the majority of the Angle-of-Arrival simulations don't extend to geolocation and focus on one single dimension (e.g. only focus on elevation θ, not on azimuth ϕ). If one were to find a similar simulation, the parameters of the scenario are sometimes hard to match. (e.g. not all parameters are given or a different Array geometry is used). Also, not all performance evaluations of geolocation system can be assuumed to be published, due to the nature of geolocation being closely related with electronic intelligece.

A similar existing simulation results have been obtained in a paper by Zoltowski et al. [28]. In this simulation 2D Unitary ESPRIT is simulated using an URA, this is very similar to this projects simulation, however the input angles are given in spatial coordinates (u, v) which are identical to (μ', ν') (Equation (2.36)) in this paper.

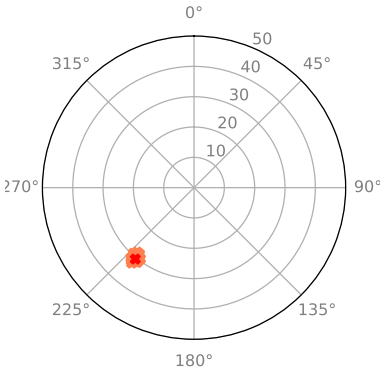


Figure 3.11: Polar plot of noiseless ESPRIT simulation

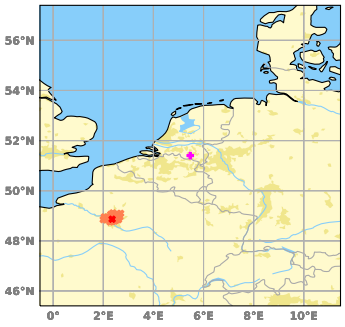


Figure 3.12: Geographic plot of noisy ESPRIT simulation (5 dB)

The simulation results published are the Root-Mean-Square-Error of the predicted spatial coordinates (μ', ν') versus the Signal-to-Noise ratio per input-source per array element. The paper displays a theoretical lower bound and experimental simulation results. This simulation scenario can be imitated in the simulation obtained in this thesis. First, the input coordinates (u, v) can be converted to AoAs (θ, ϕ) by means of equation (2.36). These AoA values can be turned into input longitudes and latitudes by using the AoA to longitude and latitude conversion software. Finally the estimated AoA of the ESPRIT algorithms can be converted back to $(\tilde{u}, \tilde{v})$. The RMSE can then be calculated according to equation (3.11).

$$RMSE = \sqrt{\frac{1}{N} \sum k = 1^{N} \{(u - \tilde{u})^2 + (v - \tilde{v})^2\}} \tag{3.11}$$

The results of this simulation, together with the results of Zoltowski et al.[28], can be seen in figure 3.13.

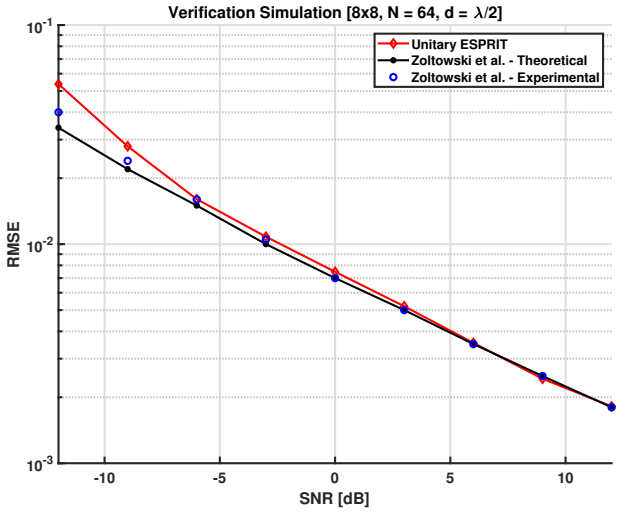


Figure 3.13: Obtained Simulation versus *Zoltowski et al.*[28]

Similarly to the ESPRIT algorithm, the MUSIC algorithm in the software are much more complex than the conversion algorithms. The approach used to verify the ESPRIT Algorithm can also be applied to the MUSIC Algorithm. Initially the correct solution in a noiseless scenario is verified. Doing 100 runs of a noiseless scenario leads to the results found in figure 3.14 and 3.15

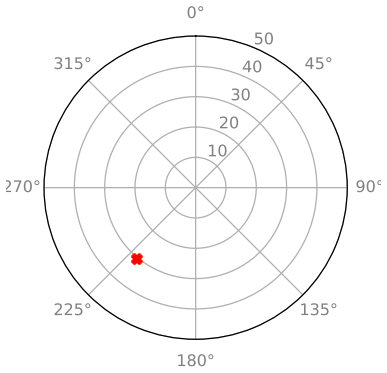


Figure 3.14: Polar plot of noiseless MUSIC simulation

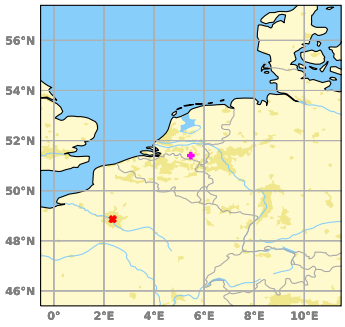


Figure 3.15: Geographic plot of noiseless MUSIC simulation

Consequently, noise is introduced in the algorithm and the simulation is run again. The obtained estimation should form an ellipse around the real solution, due to the noise propagating as an uncertainty in the results leading to a distribution around the noiseless solution. This can be observed in figure 3.16 and 3.12.

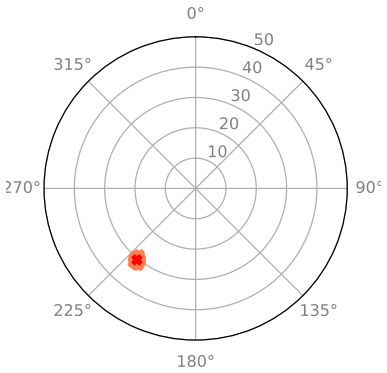


Figure 3.16: Polar plot of noiseless MUSIC simulation

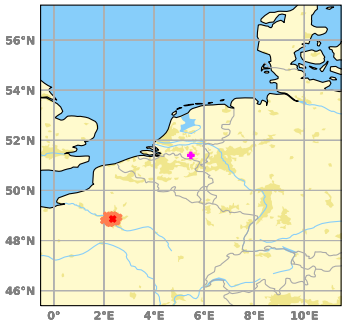


Figure 3.17: Geographic plot of noisy MUSIC simulation (5 dB)

No MUSIC algorithm simulation could be obtained where all the parameters are transparent and the figures of performance are clearly given, like in the case of the ESPRIT Algorithms. MUSIC algorithm simulations are available in 1-D (Uniform Linear Arrays), but the majority of those are unclear about peak-search step-size and proper significant digits[13][18][11][26][10]. Other simulations that do implement 2-D MUSIC algorithms, stay within analysis of the shape of the spatial spectrum without moving to accuracy results (no peak-finding)[20]. However, the ESPRIT and MUSIC algorithms share a lot of similarities, as both are dependent on the eigenspace decomposition. Comparing the MUSIC and ESPRIT algorithms after the eigen-decomposition, MUSIC is much simpler as it only compares to sub-spaces with each-other, without the need of invariance equation solving and another eigen-decomposition.

3.4. Graphical User Interface

Besides a back-end simulation architecture, a front-end graphical interface can be designed. This interface allows for the simulation framework's functionalities (calculating performance indications) to be more accessible for an end-user. By adding such an interface, the obtained simulation algorithms and software will be a fully wrapped in a product.

In order to take a systematic and system engineering approach the the design of a user interface, one can use the so called 'Planes of user experience' [6], these planes can be described in order as follows:

1. **Strategy Plane:** In this plane, the purpose of the interface is determined. It involves identifying user needs and project objectives.

2. **Scope Plane:** The scope plane defines what is and has to be defined in the interface. This plane mainly involves functional specifications/requirements and content requirements

3. **Structure Plane:** In this plane, the design on how the various features, functions and content interact with each-other. This plane mainly focuses on the information architecture

4. **Skeleton Plane:** This plane is a more concrete version of the structure plane, it holds information on how the different modules are placed and structured on the interface, and how to navigate between multiple interfaces.

5. **Surface Plane:** This plane consists of the sensory experience. The final (top) plane is the 'actual' interface.

3.4.1. Strategy

The two aspects focused on in the Strategy Plane are the user needs and the product objectives. The user needs are the goals and needs specifically to the user of the interface. Key is to understand what the user wants and what goals *they* have. The product objective embodies what the purpose and strategic interest the User-Interface has.

The specific objective decided on for this product is: *"To provide users with a performance estimate of a Space-Based Geolocation System using the Angle-of-Arrival Estimates of an Antenna Array, when provided the design parameters, in order to support the decision making process during the design synthesis in an early stage of the project"*.

When designing the interface, one should recognize that the interface has to be designed for a demographic and psychographic of users, and not a specific ideal user. Understanding the demographic, psychographic and user is part of investigating 'user needs'[6]. Due to the nature of this project being part of the RNLAF, the demographic and psychographic can be recognized in general as an employee of Section Space within the RNLAF. Meaning that the average user will have atleast basic-knowledge about space-based resources and satellites.

Another method of understanding user needs will be to look into possible Personas and Scenarios.[6]

A example persona of the RNLAF - Section Space could be a person that requires quick and accurate answers, without the need to know or read through the mathematics regarding AoA geolocation. A possible scenario could be: The RNLAF - Section Space decides to investigate Space-Based Geolocation of tracking beacon functionality for a multi-purpose satellite. Known are a satellite platform parameters and orbital parameters and Signal-to-Noise ratios. The tool can calculate what the achievable performance is of such a system when one would enter and tweak payload parameters. As such, helping in deciding whether the payload can satisfy functional requirements and aiding the decision making process whether to continue with the development of such a tracking system or not.

3.4.2. Scope

Once a clear picture is obtained of what users want and need, one can start translating the needs into specific requirements for functionality and content. The difference between these two, is that one emphasises the requirements in terms of usability and functionality, whereas content requirements focuses on requirements regarding information necessities.

REQUIREMENT	Description
REQ-GUI-FUNC-01	The GUI shall provide performance estimates using the Geolocation Simulation, Given the user Parameters
Rationale:	*Primary functionality of the interface to achieve the GUI objective.*
REQ-GUI-FUNC-02	The GUI shall show estimates in an intuitive way
Rationale:	*Showing estimation results in an intuitive way allows the user for quick judgement*
REQ-GUI-FUNC-03	The GUI shall be intuitive to use
Rationale:	*An intuitive GUI allows for an inexperienced user to use the GUI without the need to go through an extensive manual*
REQ-GUI-CONT-01	The GUI shall be managed by a content management object
Rationale:	*In order to manage all information held or created by the GUI, the information shall be wrapped in a container*

Table 3.4: Functional and Content Requirements of Graphical User Interface

The main requirements help to satisfy the GUI objective and user needs. In order to find *how* some these requirements can be achieved, sub-requirements can be made. Sub-requirements belonging to this GUI can be found in table 3.5

SUB-REQUIREMENT	Description
REQ-GUI-FUNC-01A	The GUI shall implement the Geolocation Simulation framework produced
Rationale:	*In order to provide geolocation estimates, the simulation framework produced in this thesis must be implemented in the GUI*
REQ-GUI-FUNC-02A	The GUI shall provide estimations in Root-Mean-Squared-Error (RMSE)
Rationale:	*RMSE is a very common method used to show error and should be intuitive to a person familiar with accuracy figures. Since the mean of the error should be zero, this figure also indicates the variance (distribution) of the error.*
REQ-GUI-FUNC-02B	The GUI shall present estimations in a polar plot of the satellites 'view'
Rationale:	*Showing the target and it's estimation in a polar view from the satellite's perspective allows the user to quickly see what the actual incoming angle is of the target and estimations.*
REQ-GUI-FUNC-02C	The GUI shall present estimations in a geographical plot
Rationale:	*Showing the target and it's estimation on a geographical map allows the user to see the error in terms of distance from a target where that person might be more familiar with. Showing points on a map might allow the user to more intuitively understand the error in kilometers.*
REQ-GUI-FUNC-03A	The GUI shall be limited to only one main window
Rationale:	*Presenting the GUI in only one main window allows the user to see all the information without the need to browse through different windows, menus or sub-menus. Which purpose is to increase intuitiveness of the GUI*
REQ-GUI-FUNC-03B	The GUI shall have default parameters for all inputs
Rationale:	*Showing exclusively empty boxes where information needs to be filled in might overwhelm the user, showing a default parameter gives indication on what format the parameter needs to be filled in and allows for a default version to run the simulation even if that parameter is unknown at the time.*

Table 3.5: Functional and Content Sub-Requirements of Graphical User Interface

3.4.3. Structure

After the basic requirements have been defined, one should have a clearer picture of the final product. However, one should also define on how the different objects and information of the interface form a whole. Thus, a conceptual structure for the interface should be designed.

The simulation structure itself, wraps all the functionality and information into an estimator object (figure 3.1), breaking-up this object should be avoided since it can properly hold all the simulation information in a container. Implementing the simulation framework like this doesn't only satisfy requirement `REQ-GUI-FUNC-01A`, but also `REQ-GUI-CONT-01`. The GUI should feed this estimator object the user-input, properly manage the estimator object output and finally control this estimation object to run when required. These functions can be split into several objects that can be contained within the main GUI object. This way the user only directly interacts with the parameter window, the visualization of the estimates, and with the estimator controller. An overview of this structure can be seen in figure 3.18

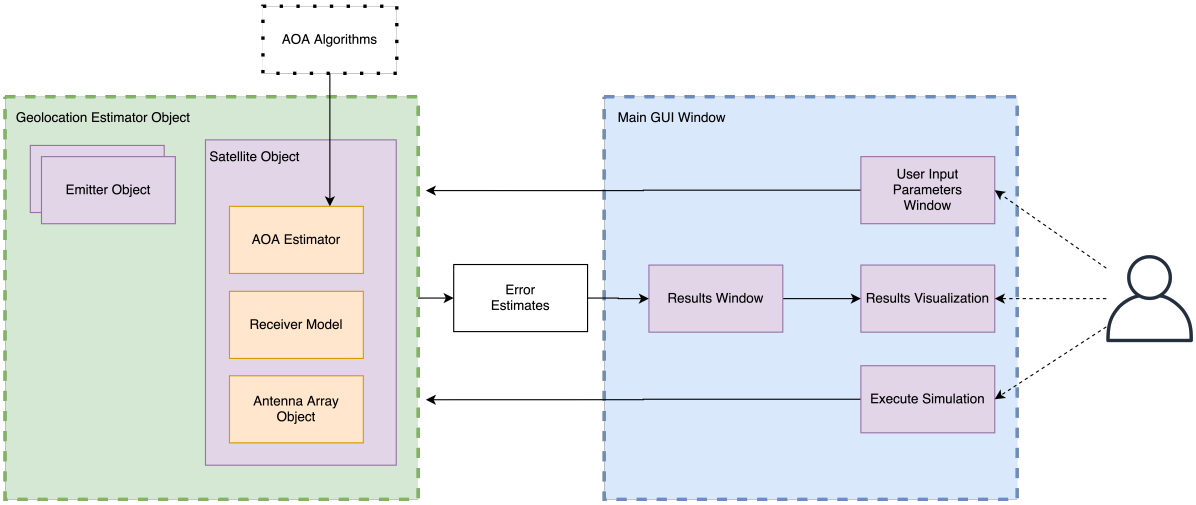


Figure 3.18: Schematic overview of GUI structure

3.4.4. Skeleton

After the basic concept structure is developed, one can further refine this concept by shaping the interface and making the structure of the front-end concrete. The GUI shall only consist of one primary window (`REQ-GUI-FUNC-03A`) and this primary window is to be populated by different objects. The objects defined are the *user input parameters window, Results Window, Simulation controller, Results Visualizer.*

Designing the user input parameters window is fairly straightforward. One should first list and categorize all input parameters available.

Parameter Description	Options
Algorithm parameters	
Algorithm	MUSIC/ESPRIT
Unitary Mode	ON/Off
Amount of Samples	-
Antenna Array Parameters	
Antenna Array Elements	-
Element Spacing	-
Satellite Parameters	-
Satellite Longitude	-
Satellite Latitude	-
Satellite Height	-
Target Parameters	-
Target Latitude	-
Target Longitude	-
Target Frequency	-
Target Signal-to-Noise	-
Amount of simulation iterations	-

Table 3.6: Available user input parameters

The parameters that have options have multiple ways to visualize this, for example a drop-down or selection '*bubbles*'. In this example selection bubbles have been selected due to the small amount of options. Parameters with no options can be visualized as an input-box or slider, in this case an input-box has been selected due to the added functionality of typing in an exact value. Finally a quick 'skeleton' is designed for the *Input Parameter* object, this can be seen in figure 3.19. Note that for input parameters in this object, a default value should be present (`REQ-GUI-FUNC-03B`).

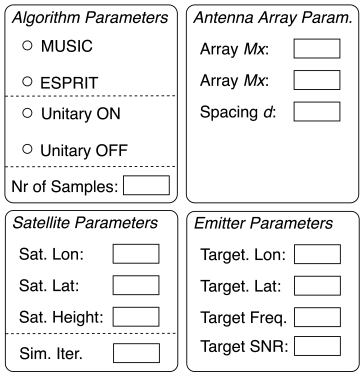


Figure 3.19: Schematic overview of *input parameters* window

The *results window* is straightforward, and should consist of a small box containing the numerical performance figures in RMSE (`REQ-GUI-FUNC-02A`). An extra required functionality which is beneficial to work efficiency is that the text should be 'select-able' to allow for 'copy-paste' of the obtained results.

The *Simulation Controller* is also a very straightforward it should essentially contain two buttons, a button to update the estimation object with new parameters, and a button to run the simulation.

The *Results Visualizer* can visualize the results in different ways. In this particular scenario, the visualizer shows the results in two ways. First, as a polar plot from the satellites 'view', allowing the user to see the incoming angle-of-arrival from the selected geographical coordinates of the emitter and satellite (`REQ-GUI-FUNC-02B`). Secondly, a map showing the geographical location of the actual emitter and the estimates, allowing the user to quickly translate the estimation error in kilometers to a more practical interpretation (For example, an estimation in Hamburg where-as actual emitter is in Berlin instead of 250 km) (`REQ-GUI-FUNC-02C`).

Lastly, all this information can be combined in order to obtain a skeletal design of the user interface. (figure 3.20)

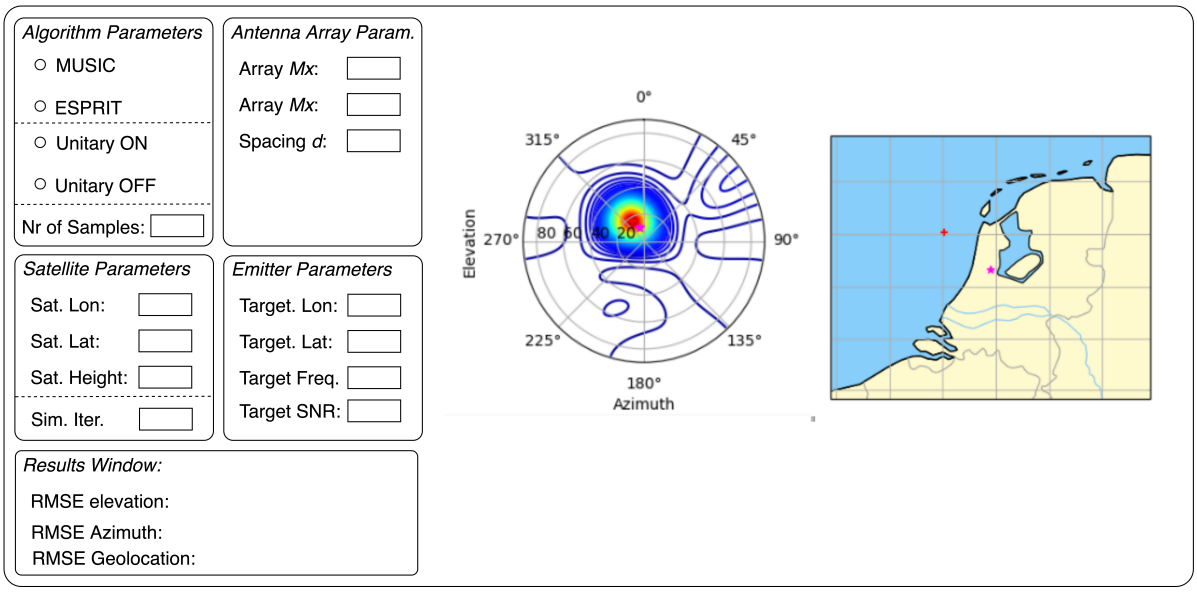


Figure 3.20: Schematic overview of GUI

3.4.5. Surface

Lastly, the obtained structure and skeleton is implemented into software, in order to obtain the product defined in the strategic plane[6].

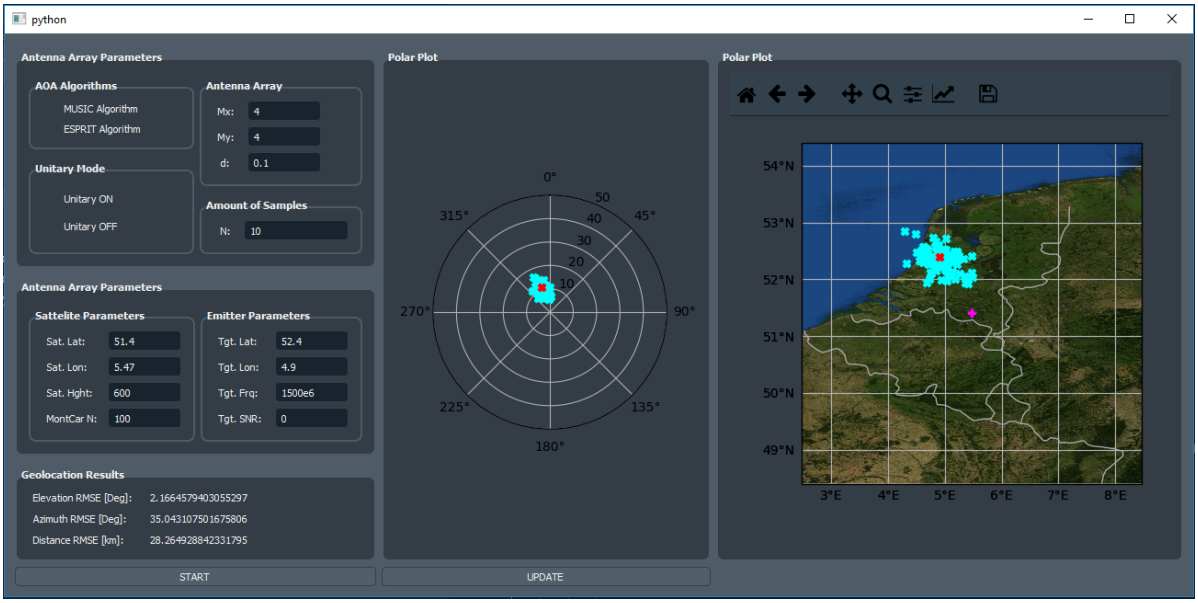


Figure 3.21: Overview of Graphical User Interface

3.5. Graphical User Interface Verification and Validation

The simulation framework itself has been verified, and from this framework a front-end GUI is produced. The final check is to validate whether this GUI is correct. A way to validate the GUI is by checking the user needs and product objective. Initially all GUI sub-requirements can be verified, this can be seen in table 3.7 ✓

SUB-REQUIREMENT	**Description**	Verification Method	Pass
REQ-GUI-FUNC-01A	The GUI shall implement the Geolocation Simulation framework produced	by design	✓
REQ-GUI-FUNC-02A	The GUI shall provide estimations in Root-Mean-Squared-Error (RMSE)	by observation	✓
REQ-GUI-FUNC-02B	The GUI shall present estimations in a polar plot of the satellites 'view'	by observation	✓
REQ-GUI-FUNC-02C	The GUI shall present estimations in a geographical plot	by observation	✓
REQ-GUI-FUNC-03A	The GUI shall be limited to only one main window	by observation	✓
REQ-GUI-FUNC-03B	The GUI shall have default parameters for all inputs	by observation	✓

Table 3.7: Functional and Content Sub-Requirements of Graphical User Interface

All main requirements should also be verified as well. A verification of the main requirements is seen in table 3.8

REQUIREMENT	**Description**	Verification Method	Pass
REQ-GUI-FUNC-01	The GUI shall provide performance estimates using the Geolocation Simulation, Given the user Parameters	by observation	✓
REQ-GUI-FUNC-02	The GUI shall show estimates in an intuitive way.	by verifying sub-req.	✓
REQ-GUI-FUNC-03	The GUI shall be intuitive to use.	by verifying sub-req.	✓
REQ-GUI-CONT-01	The GUI shall be managed by a content management object	by Design	✓

Table 3.8: Functional and Content Requirements of Graphical User Interface

Lastly, the original input of the requirements should be compared to the obtained product. Meaning that the original persona and objective should be compared to the obtained GUI in order to validate the design. The objective defined for the GUI is:

"To provide users with a performance estimate of a Space-Based Geolocation System using the AoA Estimate of an Antenna Array, when provided the design parameters, In order to support the decision making process during the design synthesis in an early stage of the project".

The persona defined for the GUI is:

"*A person that requires quick and accurate answers, without the need to know or read through the mathematics regarding AoA geolocation.*"

Comparing the objective and user need with the outcome GUI (as seen in figure 3.21), one can see that the product seems to be a tool that is of ease to use to obtain performance estimates of geolocation without the need to go through the mathematics or actual algorithms of AoA based geolocation. Therefor validating the original objective and persona. However, in order to do a true validation for both the GUI and the simulation these validation tests should be more extensive than a simple observation. In order to validate whether the user needs have been satisfied, the product should be integrated into an actual work product, and should be extensively used by multiple end-users. After-which user-data should be obtained for example by means of a questionnaire or interview. In order to fully validate the objective, the correctness of the tool needs to be further investigated, this can be done by implementing the tool into an actual product and comparing the obtained geolocation system performance with the estimated performance figures provided by the tool.

4

Sensitivity Analysis

4.1. Overview of Parameters

In this section, the obtained model and simulation framework are used to produce an sensitivity analysis of the performance of the geolocation system. Initially a baseline case is selected for the simulation, which holds the 'default' values of the design parameters. Then, the design parameters of interest and their respective range are investigated and selected.

The design parameters of interest can be divided in two categories, namely the:

- **Payload Parameters**: which hold the parameters regarding the antenna array and AoA algorithms.
- **Satellite Platform Parameters**: which hold parameters regarding the performance of the satellite platform.

and the , which hold parameters regarding the performance baseline regarding the platform.

Symbol	Parameter Description	Unit	Distribution
	Payload Parameters		
Z	Additive Noise ratio of received Signal	dB	Normal$(0,\sigma)$
N	Number of Samples taken before running the algorithm	-	Discrete Uniform
M	Amount of elements in the Antenna Array (MxM)	-	Discrete Uniform
d	Spacing between the elements of the Array relative to wavelength	-	Discrete Uniform
	Satellite Parameters		
θ_{error}	Pointing Error of satellite platform	%	Normal$(0,\sigma)$
x_{error}	Positioning Error of satellite platform	%	Normal$(0,\sigma)$

Table 4.1: Parameters investigated during sensitivity analysis

The 'baseline' parameters selected can be observed in table 4.2 and the geographical location of the 'baseline' emitter and satellite is shown in figure 4.1

Parameter	Value
SNR	10 dB
N	10
M	3
d	$\lambda/2$
MAR	0%
MPR	0%
Satellite Coordinates (λ_s, ψ_s)	Eindhoven = (5.47°, 51.4°)
Satellite Height (h_s)	600 km
Emitter Coordinates (λ_1, ψ_1)	Paris = (2.35°, 48.86°)

Table 4.2: Baseline Parameters of Analysis

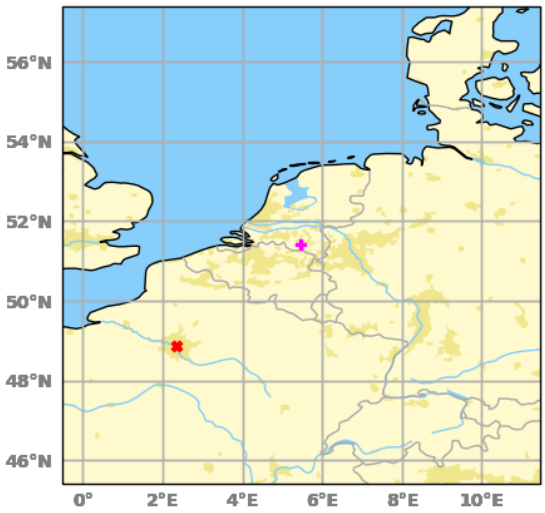


Figure 4.1: Baseline Satellite-Target Location of Analysis

4.2. Performance vs Signal-to-Noise

The noise received at the antenna is represented as the symbol **Z** (equation (2.13)). The noise is modeled as additive white Gaussian noise with a mean of zero. The amount of noise on the antenna (variance) can be derived by observing the path the signal takes from the earth-surface emitter to receiver of the satellite. A parameter commonly used to indicate the amount of noise attenuating the signal is the Signal-to-Noise Ratio (SNR), which indicates not an absolute noise power, but a noise power proportional to the signal power. Hence, the amount of noise on the signal can be derived from this SNR.

The SNR of a system is influenced by multiple (mostly physical) factors. A simplified formula to calculate the SNR can be given using equation (4.1)[12]:

$$SNR = P_{RX} - N_{RX} \tag{4.1}$$

where the P_{RX} is the received signal power in dB and N_{RX} is the received noise power in dB. Respectively, P_{RX} can be calculated using equation (4.2)

$$P_{RX} = P_{TX} + G_{TX} + G_{RX} - L \tag{4.2}$$

where P_{TX} is the transmit power of the emitter, G_{TX} and G_{RX} are the transmitter and receiver gain and finally L is the sum of all the losses. $P_{TX} + G_{TX}$ is also commonly referred to as the effective isotropic radiated power (EIRP) of the signal. The transmitter and receiver gain are commonly represented in the unit dBi, the directivity, which is the performance of the antenna in respect to an isotropic antenna (antenna that radiates all transmit power equally in all directions). The signal loss L consists of all losses in the chain, most prominently loss is the free-space.

Finally, the received noise power is calculated using equation (4.3)[12]:

$$N_{RX} = (kT_s)_{dB} B_N \tag{4.3}$$

where kT_s is the noise spectral density, calculated by multiplying the system equivalent noise temperature with the Boltzmann constant. Lastly, B_N is the noise bandwidth given in dB-Hz.
An example of calculating the SNR of a system can be as table 4.3.

Transmit Parameters	
1. Transmit Power (5W), dBW	6.99
2. Transmitter directivity, dBi	2.15
Losses	
3. Free-Space-Loss (f=1.5GHz,H=600km), dB	151.53
Receiver Parameters	
4. Receiver directivity (n=4,eff=60%), dBi	8.95
5. Noise Temperature, K	56
6. Bandwidth (50kHz), dB-Hz	46.98
7. Noise Spectral Density, dBm/Hz	-211.11
Power Summary	
8. Received Power, dBm (1+2-3+4)	-133.44
9. Received Noise, dBm (7+6)	-164.13
SNR, dB (8-9)	30.7 dB

Table 4.3: Example Calculation SNR

Common approaches to increase the SNR are to increase the directivity of the antennas and therefor the gain, reduce the equivalent noise temperature. In order to increase the directivity, a different type or bigger antenna array can be designed. In order to reduce the noise temperature, components can be selected that have a smaller noise temperature and the (preferably as early in the chain as possible) gain can be maximized (requiring more power), which will suppress noise later in the chain.

In order to model this SNR figure into the simulation, the SNR figure has to be rewritten into additive Gaussian noise $\mathbf{Z}$ (as in equation (2.13)). By rewriting SNR equation (4.1), the noise power can be obtained as $N_{RX} = P_{RX} - SNR$. If the SNR is converted from decibel scale to linear scale, the equation for the noise power can be rewritten to equation (4.4).

$$N_{RX_{\text{Lin}}} = \frac{P_{RX_{\text{Lin}}}}{SNR_{\text{Lin}}} = \frac{E[S^2]}{SNR_{\text{Lin}}} = \sigma^2_{\text{noise}} \tag{4.4}$$

Meaning that if the signal power is modeled as a value of one, the standard deviation of the Additive Gaussian noise can be given according to equation (4.5).

$$\sigma_{\text{noise}} = \sqrt{\frac{1}{SNR_{\text{Lin}}}} = \sqrt{10^{\frac{-SNR_{\text{dB}}}{10}}} \tag{4.5}$$

Lastly, the complex noise $\mathbf{Z}$ can be modeled using this noise power and equation (4.6) can be obtained.

$$\mathbf{Z} = \frac{\sigma_{\text{noise}}}{\sqrt{2}} \cdot (N(0,1) + j \cdot N(0,1)) \tag{4.6}$$

By using the baseline parameters given by 4.2, the system can be simulated and the results in figure 4.2, figure 4.3 and table 4.4 are obtained.

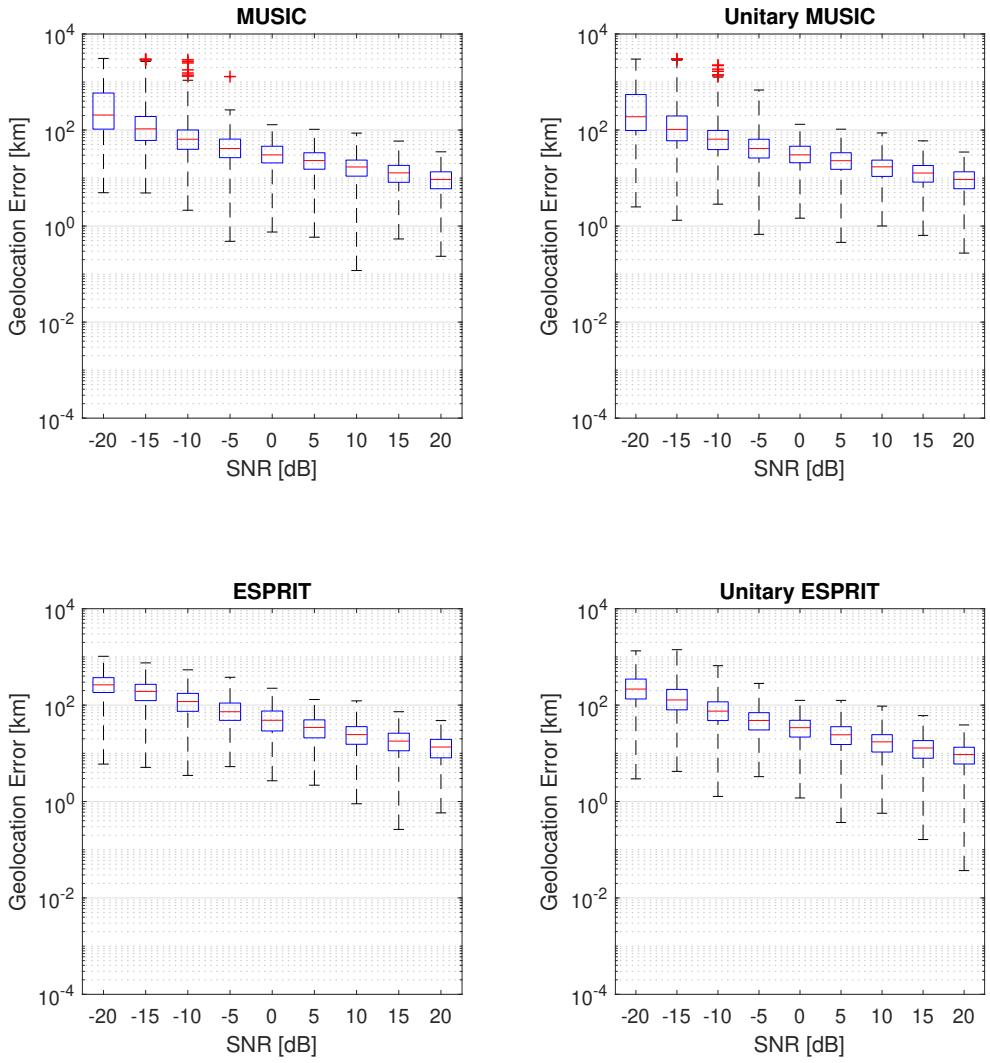


Figure 4.2: box plot of Geolocation error vs Signal-to-Noise

In Figure 4.2, the so-called box plots of the resulting Geolocation Error can be observed. From this graph one can see that by increasing or decreasing the SNR, the median value of the geolocation error shifts down, but the overall distribution of this geolocation remains roughly the same. Another observation is that the box itself is small in comparison to the total range and whisker length, meaning that the majority of the observations of the geolocation error are tightly-packed around the median. However when observing the whiskers, (especially in the ESPRIT algorithms) the distribution geolocation error is skewed to the lower end. Meaning that the the probability of an outlier of the geolocation error being much lower than the median is higher than the probability of the outlier having a much higher geolocation error.

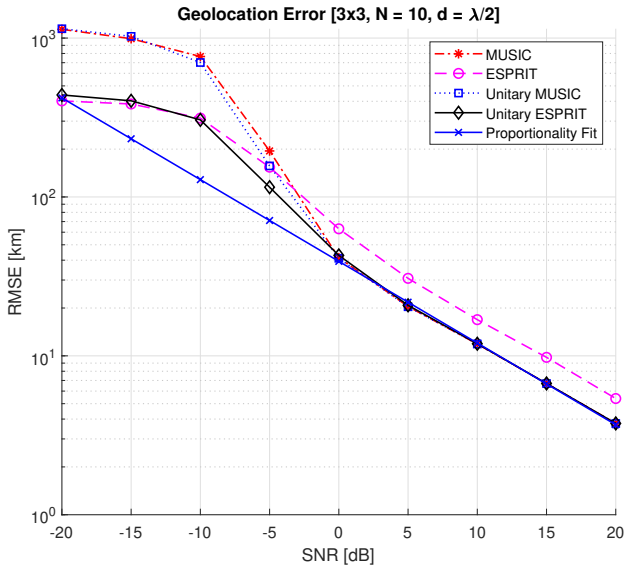


Figure 4.3: Geolocation RMSE vs Signal-to-Noise

SNR [db]:	-20.00	-15.00	-10.00	-5.00	0.00	5.00	10.00	15.00	20.00
ESPRIT: [km]	312.02	243.55	158.77	99.23	64.81	43.70	31.39	22.92	17.22
unitary ESPRIT: [km]	303.17	221.92	118.58	66.58	42.59	30.92	21.90	16.09	12.04
MUSIC: [km]	724.01	417.24	222.09	72.81	40.33	29.91	21.48	15.94	11.93
unitary MUSIC: [km]	697.83	467.10	197.76	63.47	40.04	29.80	21.48	15.95	11.93

Table 4.4: Performance of geolocation (RMSE km) vs SNR

In figure 4.3, one can observe that for the SNR > 0 dB the geolocation RMSE and the SNR appear to have a linear relationship on semi-logarithmic scale. This suggests that if (SNR > 0 dB), an increase of 1 dB in SNR leads to a decrease of 11% in of the error. This suggests a proportionality of RMSE $\propto 0.89^{\text{SNR}}$. However, in the low SNR region (SNR < 0 dB), this linear relationship appears not to hold. Besides this, in the low SNR region the ESPRIT algorithm (specifically Unitary ESPRIT) performs much better than the MUSIC algorithms. A way to rationale this difference is due to the nature of the algorithms. The MUSIC algorithm estimates the AoA by using a peak-search algorithm to find the signal's null-space that is occupied by the noise space. Having an extremely high noise (extremely low SNR), can lead to this null-space becoming unobservable. Whereas the ESPRIT algorithms uses an invariance in the signal-space that is 'contaminated' by the the noise space to find the AoA.

Besides observing RMSE of the geolocation algorithms, The algorithm run-time can also be observed over different values of the SNR. These values have been measured and can be seen in figure 4.4 and table 4.5.

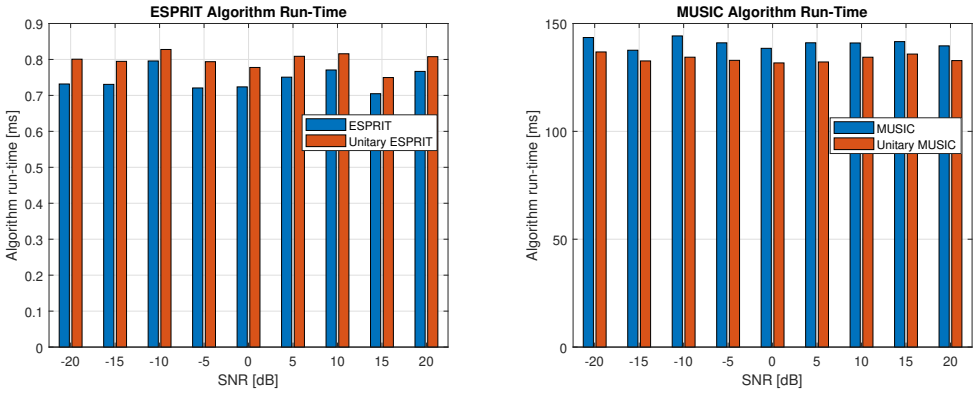


Figure 4.4: Algorithm timing vs Signal-to-Noise

SNR[dB]	-20.00	-15.00	-10.00	-5.00	0.00	5.00	10.00	15.00	20.00
ESPRIT [ms]	0.73	0.73	0.80	0.72	0.72	0.75	0.77	0.70	0.77
Unitary ESPRIT [ms]	0.80	0.79	0.83	0.79	0.78	0.81	0.82	0.75	0.81
MUSIC [ms]	143.50	137.61	144.24	141.03	138.49	141.05	140.97	141.59	139.63
Unitary MUSIC [ms]	136.81	132.65	134.38	132.93	131.72	132.17	134.37	135.83	132.81

Table 4.5: Algorithm timing vs Signal-to-Noise

It should be realized that the absolute run-time performance of the algorithms is hardware and software dependent. The hardware of the computer running the simulations influences the run-time performance by means of processor cycles per second and cycles per instruction, another commonly used method of measuring computational power is Floating Point Operations Per Second (FLOPS). The software is implemented in Python code, which is a 'dynamic' programming language, meaning that during run-time many other processes are running on the background, namely evaluating and dynamically interpreting the code and so called 'garbgage collection processes'. However, the relative difference in algorithm run-time does contain information regarding run-time sensitivity due to change in SNR. The run-time measurements stay relatively the same over different values of the SNR, suggesting that the algorithm run-time is decoupled from the SNR and remains approximately constant over different values of the SNR. This behaviour can be explained by the fact that a different noise value does not obiviously impact the excecution time of the code as the algorithms remain the same. Another observation is that the ESPRIT algorithms, have a drastically lower run-time than the MUSIC algorithm, this can be explained by the MUSIC algorithm containing a time-exhaustive peak-search algorithm.

Lastly, one can observe whether the relationship between the RMSE and the SNR (Noise) is independent of other parameters. This can be tested by changing other parameters and observing whether the behaviour of the RMSE over the SNR remains the same. The simulation is re-run with a different amount of elements, amount of samples and element spacing. The results of this simulation is seen in figure 4.5. This graph shows that when other parameters change, the behaviour of the geolocation error over SNR remains similar. This suggests that the proportionality of RMSE $\propto 0.89^{\text{SNR}}$, is independent of the other parameters, especially for higher SNR. Whereas (similar to the original simulation) the proportionality does not hold for low-SNR (SNR < 0 dB).

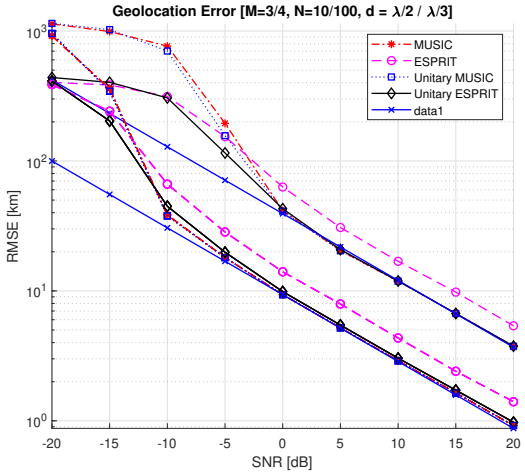


Figure 4.5: Geolocation RMSE vs SNR in semilog Scale

4.3. Performance vs Number of Samples

Another important parameter for the algorithms is the amount of samples it uses to perform the algorithm (as discussed in section 2.2.3). This parameter should be coupled to the performance of the algorithms since it increases and decreases the amount of input data to the algorithms. The geolocation algorithms can be simulated in order to obtain the performance for different values of the amount of input samples using the baseline scenario given by table 4.2. The obtained geolocation performance is given by figure 4.6, figure 4.7 and table 4.6

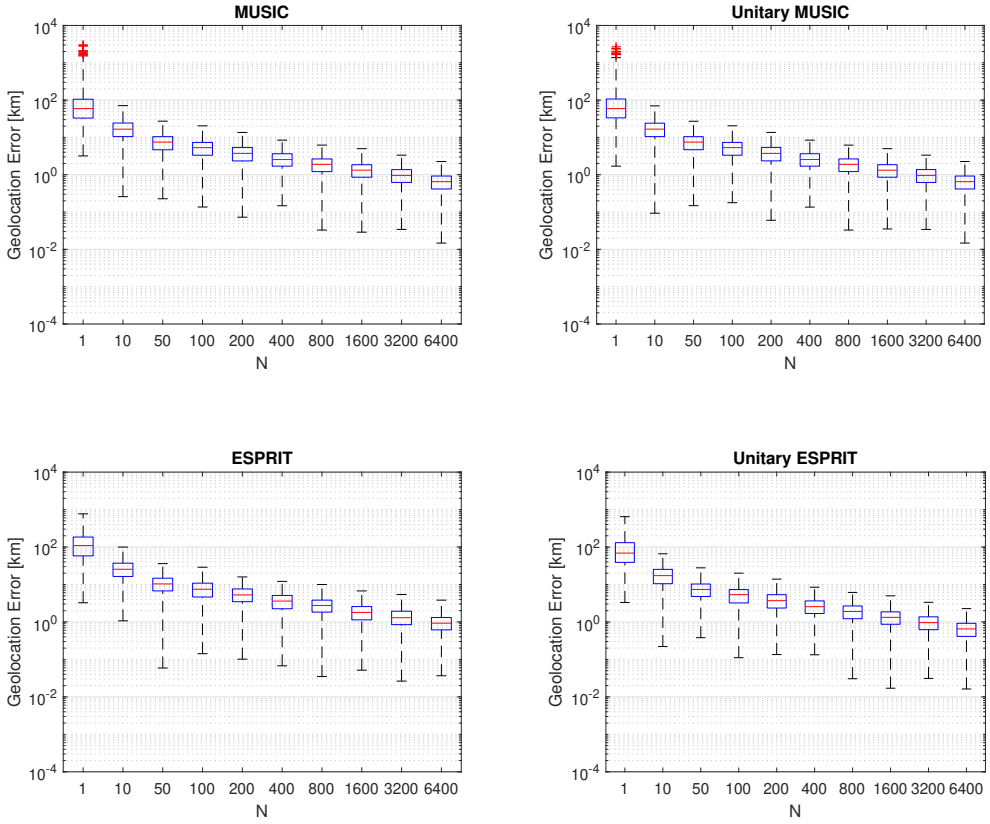


Figure 4.6: box plot of Geolocation error vs number of samples

in Figure 4.6 The box plots of the resulting geolocation error can be observed. These graphs show similarities to the SNR analysis, by increasing or decreasing the amount of samples, the median value is shifted down, but the overall range of the errors remains roughly the same. Also similar to the SNR analysis, the data appears slightly skewed downwards.

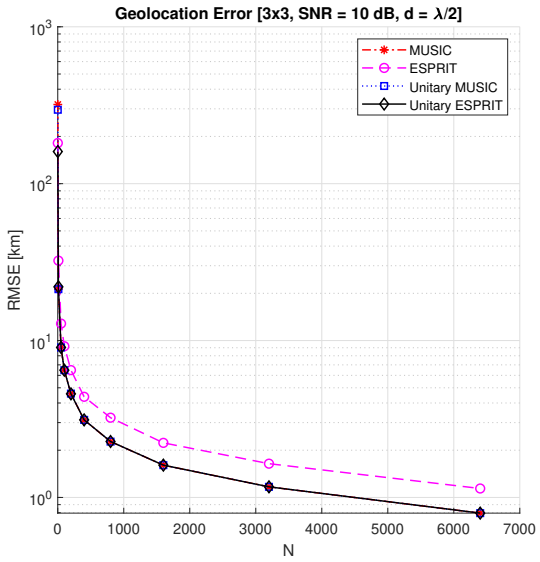


Figure 4.7: Geolocation RMSE vs Number of Samples

N [-]:	1	10	50	100	200	400	800	1600	3200	6400
ESPRIT: [km]	181.54	32.32	12.83	9.20	6.48	4.38	3.22	2.23	1.64	1.14
unitary ESPRIT: [km]	160.17	21.99	9.04	6.47	4.57	3.12	2.27	1.61	1.17	0.79
MUSIC: [km]	319.65	21.23	9.03	6.47	4.59	3.12	2.27	1.61	1.17	0.79
unitary MUSIC: [km]	295.83	21.25	9.03	6.47	4.59	3.12	2.27	1.61	1.17	0.79

Table 4.6: Geolocation RMSE vs Number of Samples

One can immediately notice that increasing the number of samples leads to a dramatic improvement in the geolocation accuracy. A jump from 1 to 10 samples can lead to an improvement of factor >7. One can also notice that the logarithmic RMSE of the geolocation seems to be of exponential relationship with the amount of samples. The decrease in the error almost appears asymptotic, this can be explained by the other factors limiting the accuracy, the error can only approach zero but never reach it. One can plot this graph on log-log scale, the resulting graph can be observed in figure 4.8

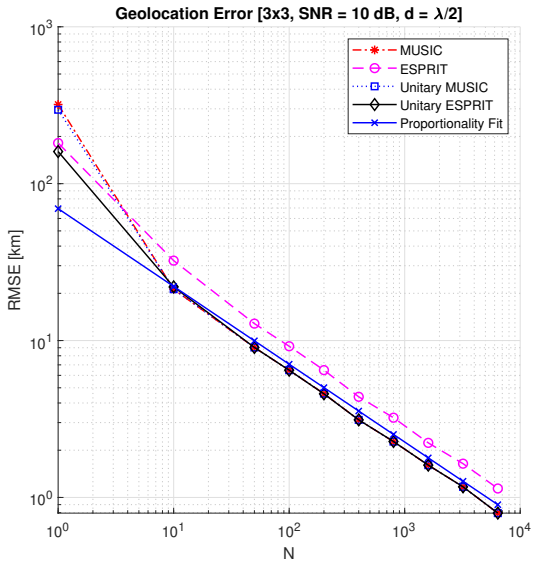


Figure 4.8: Geolocation RMSE vs Number of Samples in Log-Log Scale

When observing this log-log scaled graph, one can see that for $N > 10$ the relationship appears linear, meaning that the relationship between the RMSE of the geolocation and the amount of samples is approximately $\text{RMSE} \propto N^{-0.46}$ ($\frac{\Delta \log \text{RMSE}}{\Delta \log N} \approx -0.46$). Another aspect one notices on this scale, is that for single snapshot measurements $N = 1$, the ESPRIT algorithms perform better.

Increasing the amount of samples, might also alter the run-time of the algorithms, plotting the algorithm run-time over the amount of samples leads to the results represented in figure 4.9 and table 4.7

N [samples]:	1	10	50	100	200	400	800	1600	3200	6400
ESPRIT: [ms]	0.79	0.81	0.76	0.87	0.85	0.93	0.92	0.98	1.11	1.66
unitary ESPRIT: [ms]	0.82	0.82	0.75	0.96	0.95	1.03	1.03	1.10	1.25	2.15
MUSIC: [ms]	134.53	135.76	134.23	146.29	178.02	175.83	176.34	178.15	176.65	173.47
unitary MUSIC: [ms]	130.63	137.13	130.40	155.15	154.11	156.59	153.72	154.60	154.64	157.56

Table 4.7: Algorithm run-time vs number of samples

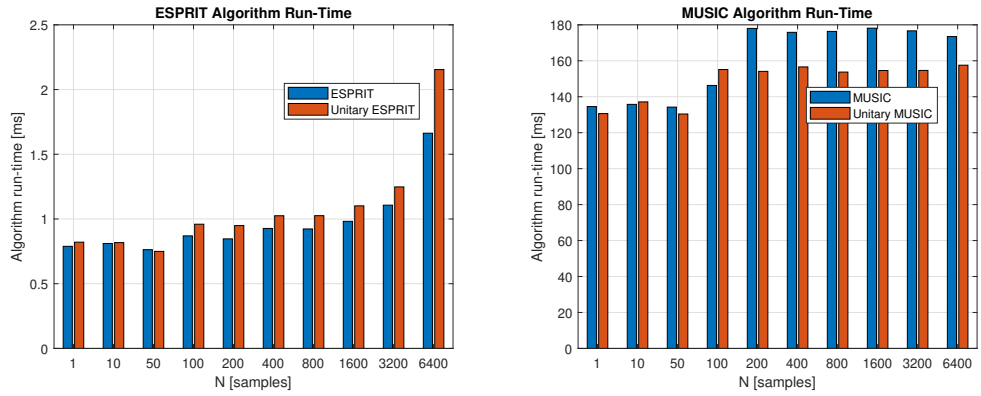


Figure 4.9: Algorithm run-time vs number of samples

From the run-time one can observe, that an increase in the amount of samples leads to an increase of run-time. The increase in the number of samples leads to a bigger calculation of the autocovariance, causing an

initial increase in the run-time for both algorithms. The increased amount of samples has the biggest impact on the ESPRIT algorithms as the run-time appears to double, whereas the increase in the MUSIC algorithms is approximately ≈25% from 1 to 6400 samples.

It is also investigated whether this proportional behaviour of the RMSE of the geolocation is independent of the other parameters. The simulation is re-run with a different amount of elements, Signal-to-Noise and element spacing. The results of this simulation is seen in figure 4.10. One can observe that for $N > 10$ the proportionality of RMSE $\propto N^{-0.46}$ holds for different parameters of the array, suggesting that the effect of N on the performance is independent on the other parameters.

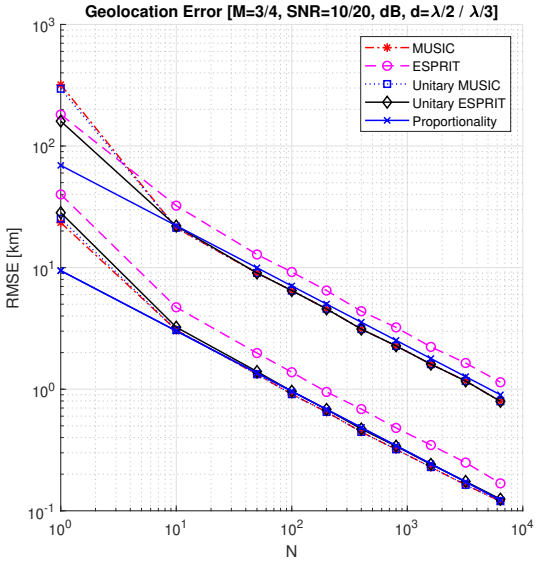


Figure 4.10: Geolocation RMSE vs Number of Samples in Log-Log Scale

Knowing this information one might be inclined to use the algorithms with an extremely high amount of samples (N > 1000), however, this is impossible due to practical limits. One of the limits is memory, assuming every sample is a complex 32-bit floating point number the memory costs are 8 bytes per sample per array element (4 + 4 bytes). Assuming the baseline scenario (3 x 3 Antenna Array) the amount of storage required quickly reaches a big amount. ($N = 10,000 \rightarrow$ 720kB)
Finally, another limit to obtaining a big amount of samples is the act of sampling itself. The sampling of a signal in orbit is not an *instantaneous* function, meaning the satellite cannot take all samples at the same time immediately. The movement of the satellite during sampling 'contaminates' the data due to the angle of arrival changing during measurements. However, even though the sampling gets contaminated by a shift in the AoA, it might still allow for an increase in the performance. The speed at ones does sampling is commonly limited by the Analog-to-Digital converter of the antenna array, by simulating the sensor array while in movement, one can find an optimal amount of samples fitting the sampling-rate of the satellite.

The position of the satellite can be extrapolated over time by assuming a circular orbit and by either fixing the location of the ascension node or the inclination. Using this information one can find a plane that contains the origin, satellite and ascension-node (in an earth-centric frame) by means of cross product (equation (4.7)).

$$n_{plane} = v_{satellite} \times v_{ascension-node} \tag{4.7}$$

The entire circular orbit is contained in this plane. The center-point of the orbit is the origin and the circular orbit intersects the satellite position. By using these properties the circular orbit can be found. By using the angular velocity and the circular velocity (equation (4.8) & (4.9) [4]) one can propagate the orbit for time $t = \tau$, note that this method does not take into account orbital perturbations[4], hence it would only hold for small τ. An example of using this method to extrapolate a circular orbit from a satellite position and an ascension-node longitude can be found in figure 4.11

$$v_{circular} = \sqrt{\frac{GM}{r}} \tag{4.8}$$

$$v_{circular} = \omega_{circular} r \tag{4.9}$$

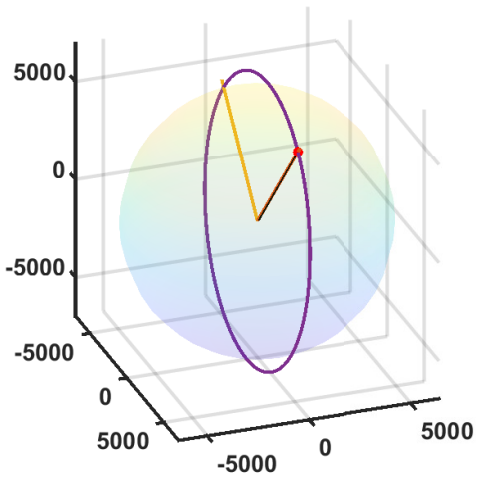


Figure 4.11: Propagation of circular orbit

For example, taking 200 Samples with a sample-rate of 10 Hz takes (200/10Hz) 20 seconds. An orbital height of 600 km has an angular velocity of approximately $\omega_{circular}$ = 0.0011 rad/s, this leads to an angular shift of 1.24°. Assuming the case that a satellite is currently above Eindhoven (Longitude = 5.47, Latitude = 51.4) and an emitter is situated in Amsterdam (Longitude = 4.9, Latitude = 52.4), the approximate drift on ground can be observed in figure 4.12.

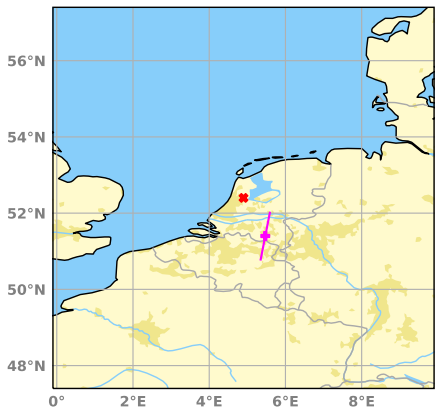


Figure 4.12: Movement of satellite during 20 seconds measurement

In order to visualize the inaccuracy caused by the satellite movement during sampling, previous' simulation can be reevaluated, incorporating satellite movement in the model. The three cases simulated are an extremely low sample rate (10 Hz), an average sample-rate (1 kHz) and a sample rate readily available in

commercial Analog-to-Digital converters (1.6 MHz). The results of this simulation can be observed in figure 4.13 and on log-log scale in figure 4.14

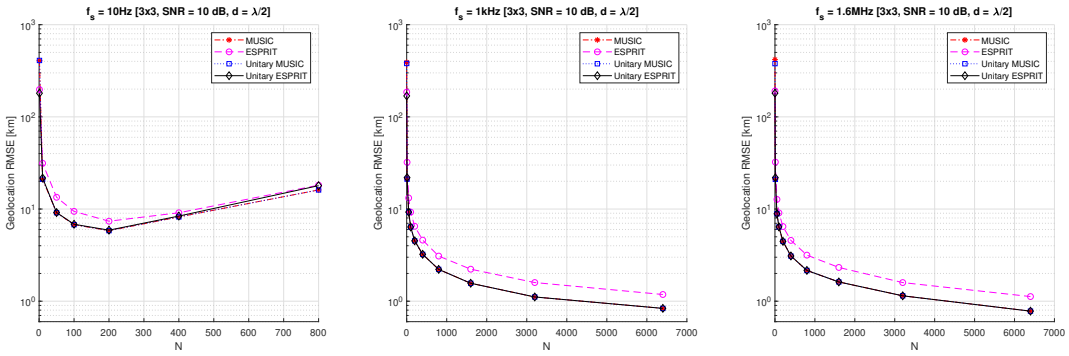


Figure 4.13: Geolocation RMSE vs Number of Samples including drift

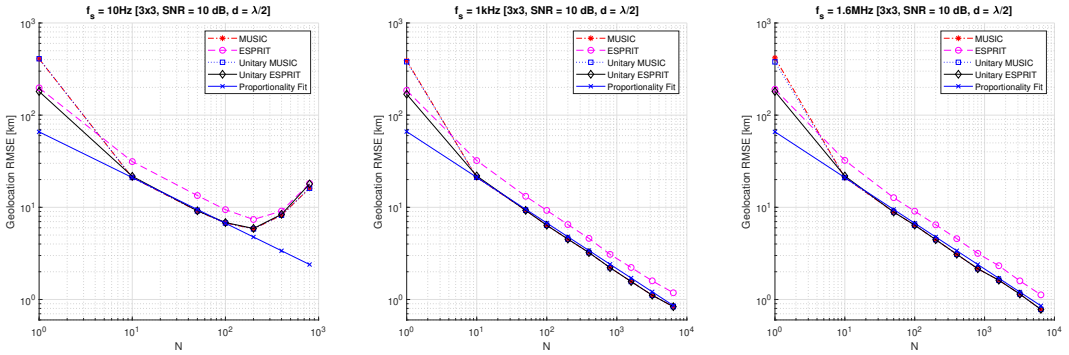


Figure 4.14: Geolocation RMSE vs Number of Samples including drift on Log-Log scale with proportionality approximation

From these results it is observed that if the sampling rate of the antenna array is high enough, inaccuracies introduced by movement of the satellite is not going to noticeably influence the accuracy of geolocation for $N < 6400$. However, when using a low sampling rate, an increase in samples might actually reduce the performance of the system. This happens when the performance loss due to drift 'contamination' is bigger then the performance gain when the number of samples is increased. In the log-log scale figure (4.14), the approximation of proportionality defined earlier (RMSE $\propto N^{-0.46}$) is plotted as well, and one can see that if $f_s \to \infty$ the RMSE behaviour converges to this proportionality.

4.3.1. Performance vs Number of Elements

The number of elements in an antenna array should also influence the geolocation accuracy. However, the number of elements in the array cannot be arbitrarily selected. The number of eigenvectors of the autocovariance matrix should be equal to the number elements in the array. (autocovariance is full-rank see section 2.2.3). In order for the algorithms to solve the angle-of-arrival, the eigenspace should be big enough to distinguish the noise space and signal space. Meaning that the eigenspace should at-least be the size of the signal-space +1. If the eigenspace is of the size M^2 (where M^2 is the number of elements in the array and M the number of elements per dimension) and the signal-space is of the size N (where N is the number of incoming signals), this constraint can be shown by equation (4.10)

$$M^2 \geq N + 1 \tag{4.10}$$

When the amount of elements in the array increase, the amount of samples and the size of the eigenspace also increase, hence leading to a greater performance. However, the amount of elements in an antenna array has an upper-bound by means of physical complexity. A higher amount of elements has a much greater implementation complexity and power requirement.

In figure 4.15, 4.16 and table 4.8 one can see the obtained results from simulating the antenna array for multiple values of the amount of elements, ranging from a total of 4 elements to 49 elements.

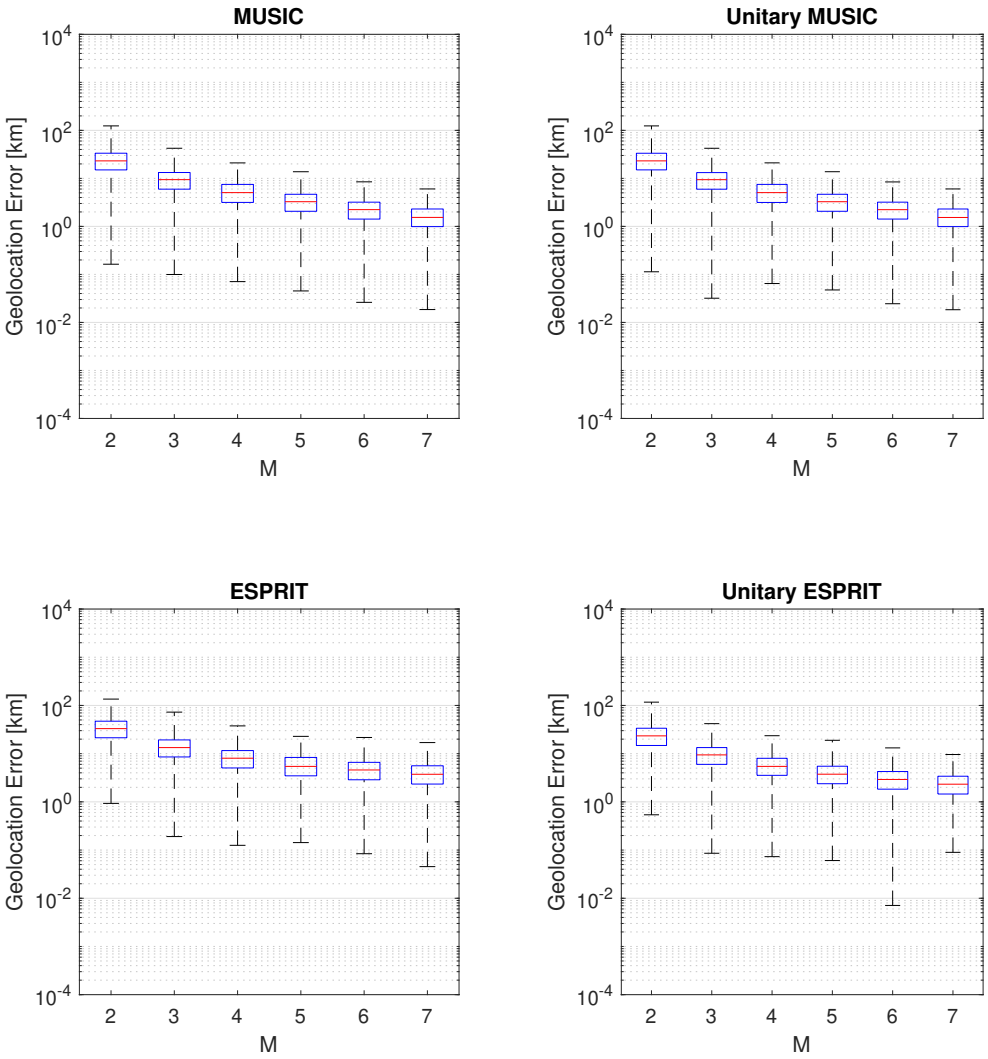


Figure 4.15: box plot of Geolocation error vs number of elements

Figure 4.15 presents the box plots of the resulting Geolocation Error. This graph is again similar to the number of elements and noise analysis. Increasing or decreasing the amount of elements causes the median value to either shift up or down, but the overall range of the errors remains roughly the same. Also similar to the other analyses, the data appears slightly skewed downwards.

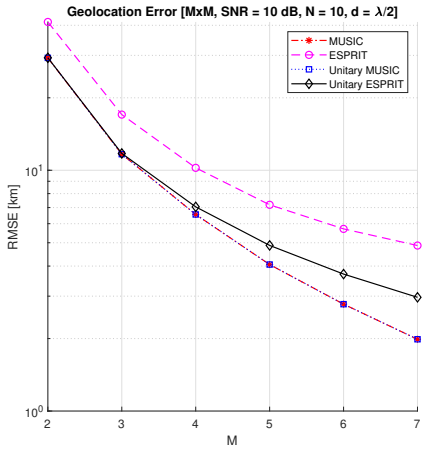


Figure 4.16: Geolocation RMSE vs Number of Elements in antenna array

M [-]:	2	3	4	5	6	7
ESPRIT: [km]	41.30	17.03	10.24	7.19	5.71	4.87
unitary ESPRIT: [km]	29.32	11.75	7.05	4.87	3.71	2.97
MUSIC: [km]	29.30	11.67	6.55	4.06	2.78	1.99
unitary MUSIC: [km]	29.30	11.66	6.55	4.06	2.78	1.99

Table 4.8: Geolocation RMSE vs Number of Elements in antenna array

It is observed that for both ESPRIT and MUSIC algorithms, the RMSE decreases with an increasing amount of elements, however when $M > 3$ the MUSIC algorithm starts performing significantly better then the ESPRIT algorithm. One could rationale this by closely investigating the data used by the algorithms to calculate the AoA. The data exploited by the ESPRIT algorithm is the signal subspace, every element added to the calculation should add one row to this subspace. However, the MUSIC algorithm exploits the noise-subspace, and every element added increases the noise subspace with one row *and* one column. Meaning that an increase of elements has a much bigger impact on the data exploited by MUSIC then by ESPRIT. The geolocation RMSE vs the number of elements can also be plotted on logarithmic scale on both axis, giving figure 4.17. One can observe that the performance of the MUSIC algorithm on this scale is shown as linear, suggesting a proportionality of $\text{RMSE}_{\text{MUSIC}} \propto M^{-2.15}$. This proportionality is also represented in the figure.

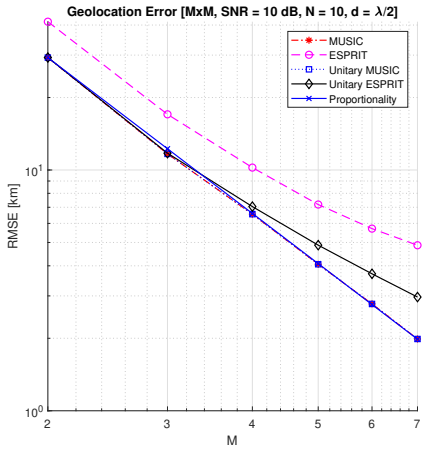


Figure 4.17: Geolocation RMSE vs Number of Elements in antenna array on log-log scale with approximation line

The behaviour of the run-time of the algorithms can also be observed as a function of M. The run-time of the

algorithm for the different values of M can be found in figure 4.18 and table 4.9.

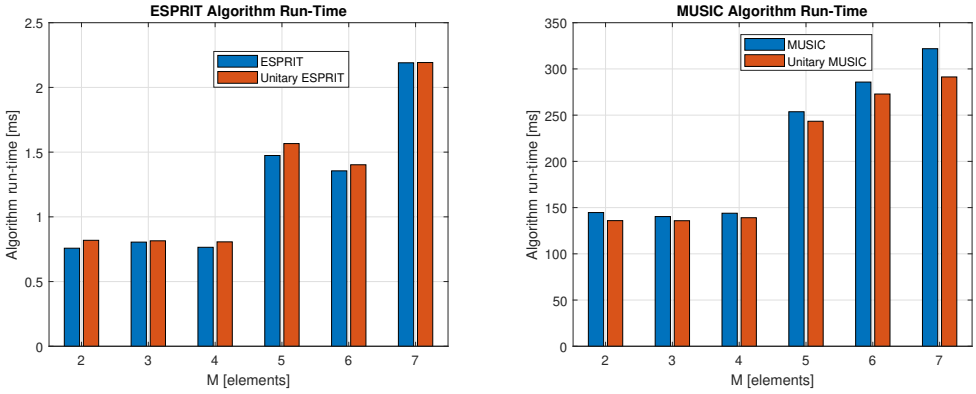


Figure 4.18: Algorithm timing vs Number of elements

M[-]	2	3	4	5	6	7
ESPRIT [ms]	0.76	0.80	0.76	1.47	1.36	2.19
Unitary ESPRIT [ms]	0.82	0.81	0.81	1.57	1.40	2.19
MUSIC [ms]	144.64	140.29	143.92	253.68	285.78	321.87
Unitary MUSIC [ms]	135.91	135.81	139.05	243.40	272.86	291.32

Table 4.9: Algorithm timing vs Number of elements

It can be observed that the algorithm timing is coupled to the size of the array. Increasing M leads to a significant increase in both the run-time of the MUSIC algorithms and the ESPRIT algorithms. The increase in the run-time seems to be proportional. However, due to the large run-time of the MUSIC algorithms (also in the baseline case) this increase is much more significant in absolute numbers (increase in ESPRIT algorithms: $\approx$ 1.5ms, increase in MUSIC algorithms: $\approx$ 150ms)

The system can also be simulated for multiple impinging signals on the antenna array. This simulation can be observed in figure 4.19

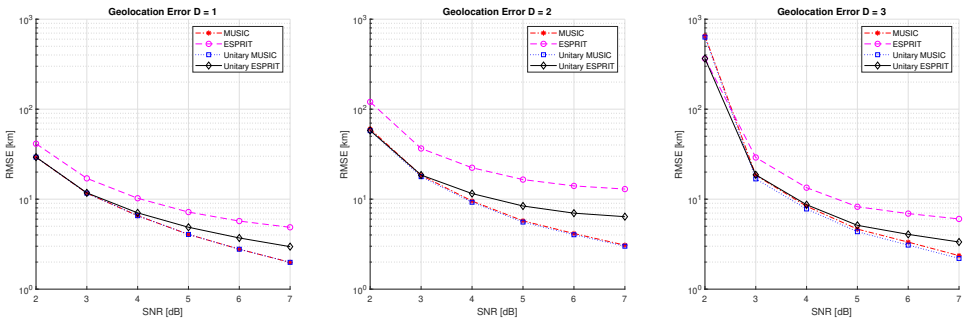


Figure 4.19: Geolocation RMSE vs Number of Elements in antenna array

City	**Longitude**	**Latitude**
Berlin	13.40°	52.527°
London	−0.12°	51.51°

Table 4.10: Geographic coordinates of two additional earth-surface emitters

It can be observed that for a higher amount of incoming signals, the performance reduces. The cause of this can be shown more clearly by observing the spatial spectrum of the MUSIC algorithm. When the amount of

incoming signals increase, but the amount of elements remains the same, the noise-space reduces in size. When the workable noise-space reduces in size the peaks in the spatial spectrum become much less defined, until it is reduced in such a degree that the spatial spectrum peaks start to 'merge'. When this happens, the peak finding algorithms misses or wrongly distinguishes peaks. This process is shown in figures 4.23 and 4.22 for two incoming signals and figures 4.25 and 4.24 for three incoming signals. Another noticeable property is that if the amount of incoming signals increase, the MUSIC algorithms still surpass the ESPRIT algorithms in performance, around $M = 3$ when $D = 1$, $M = 4$ when $D = 2$ and $M = 4$ when $D = 3$. This further suggests that the reason that the performance of the MUSIC algorithms overtake the ESPRIT algorithms is due to a bigger amount of input data in the MUSIC algorithms for a bigger M. An approach to find the point where the MUSIC algorithms overtake the ESPRIT algorithms in performance, is to assume that the suggestion of causality of the MUSIC data-set being bigger than the ESPRIT data-set is correct. The size of the signal-space is namely D where D is the amount of signals, and the size of the noise-space can be modeled as $P - D$, where P is the amount of elements in the array ($P = M^2 - D$). This means that the noise-space size will overtake the signal-space size (the MUSIC data-set wil overtake the ESPRIT data-set) at $P > 2D$ or $M^2 > 2D$. The obtained simulation results and figures of MUSIC spatial spectra (figure 4.23, 4.22, 4.25, 4.24) seem to support the obtained hypothesis. This means that the MUSIC algorithms overtake the ESPRIT algorithms when the amount of columns describing the spatial spectrum is similar to the amount of peaks in the spatial spectrum.

One can also investigate whether the performance behaviour of the geolocation (due to the amount of array elements) is independent on the other parameters. When increasing the amount of samples and increasing the Signal-to-Noise ratio, the results in figure 4.20 are obtained. One can see that the overall performance is much better (as expected), but the overall shape and slope of the curves remain the same, suggesting that the behaviour of the performance caused by the array elements is independent of other parameters. This different case is also plotted on log-log scale and the same proportionality as in the original case (RMSE $\propto M^{-2.15}$) is represented around the new case, using this one can observe that the behaviour of the RMSE of the MUSIC algorithms is almost identical, and that the ESPRIT algorithms actually seem to approach this behaviour.

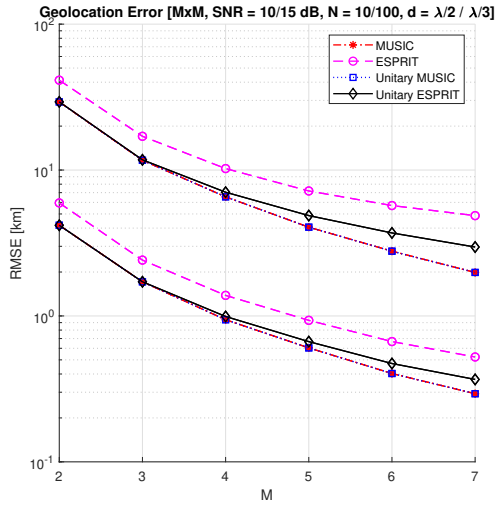


Figure 4.20: Geolocation RMSE vs Number of Elements in antenna array with different configuration

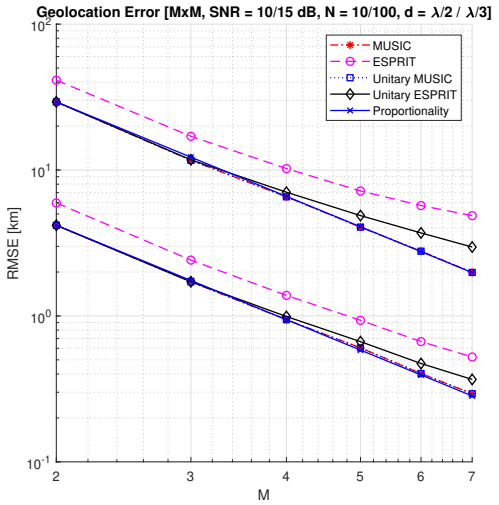


Figure 4.21: Geolocation RMSE vs Number of Elements in antenna array with different configuration on log scale

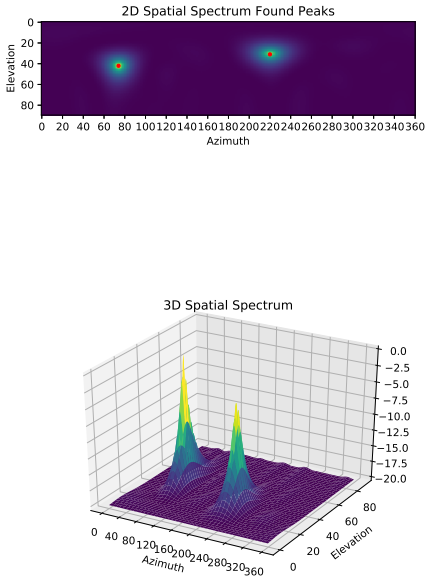


Figure 4.22: two incoming signals on a 5x5 array, peaks are clearly separate

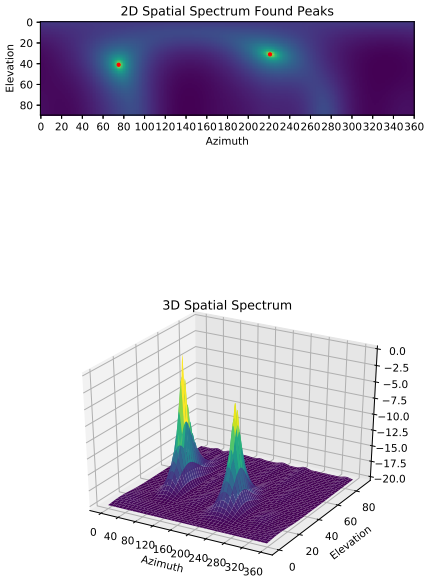


Figure 4.23: two incoming signals on a 2x2 array, peaks are slightly merging

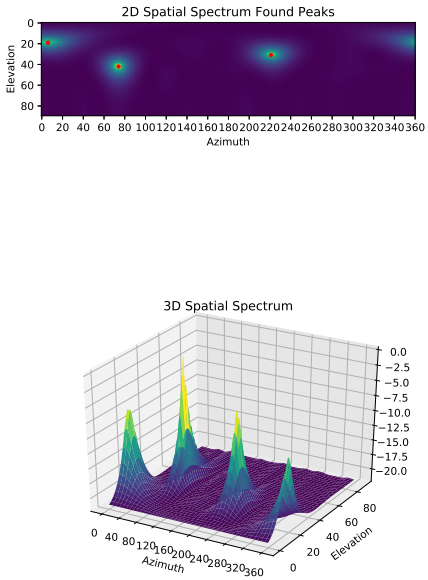


Figure 4.24: three incoming signals on a 5x5 array, peaks are clearly separate

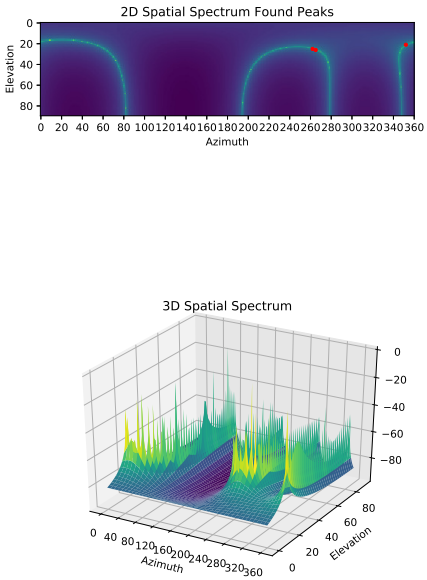


Figure 4.25: three incoming signals on a 2x2 array, peaks are clearly merging

4.3.2. Performance vs Element Spacing

The spacing between the antenna array elements also influences the performance of the antenna array. The reason why the AoA can be extracted from antenna array samples is essentially due to the phase difference per antenna array element (section 2.2.3), when one decreases the elemental spacing, this phase difference is reduced and when someone increases the elemental spacing, this effect is exaggerated. Thus one could hypothesize that the bigger the elemental spacing, the bigger the phase shift effect, the better the performance. However, when the phase difference of a signal is bigger then half a wavelength, ambiguity is introduced in the measurements. Meaning that the maximum performance of geolocation should be when the element spacing equals half-wavelength, whereas an element spacing greater than half-wavelength would degrade the performance drastically

The performance of the geolocation is simulated for different values of the element spacing d using the baseline parameters given by table 4.2. The outcomes of this simulation can be observed in figure 4.26, 4.27 and table 4.11

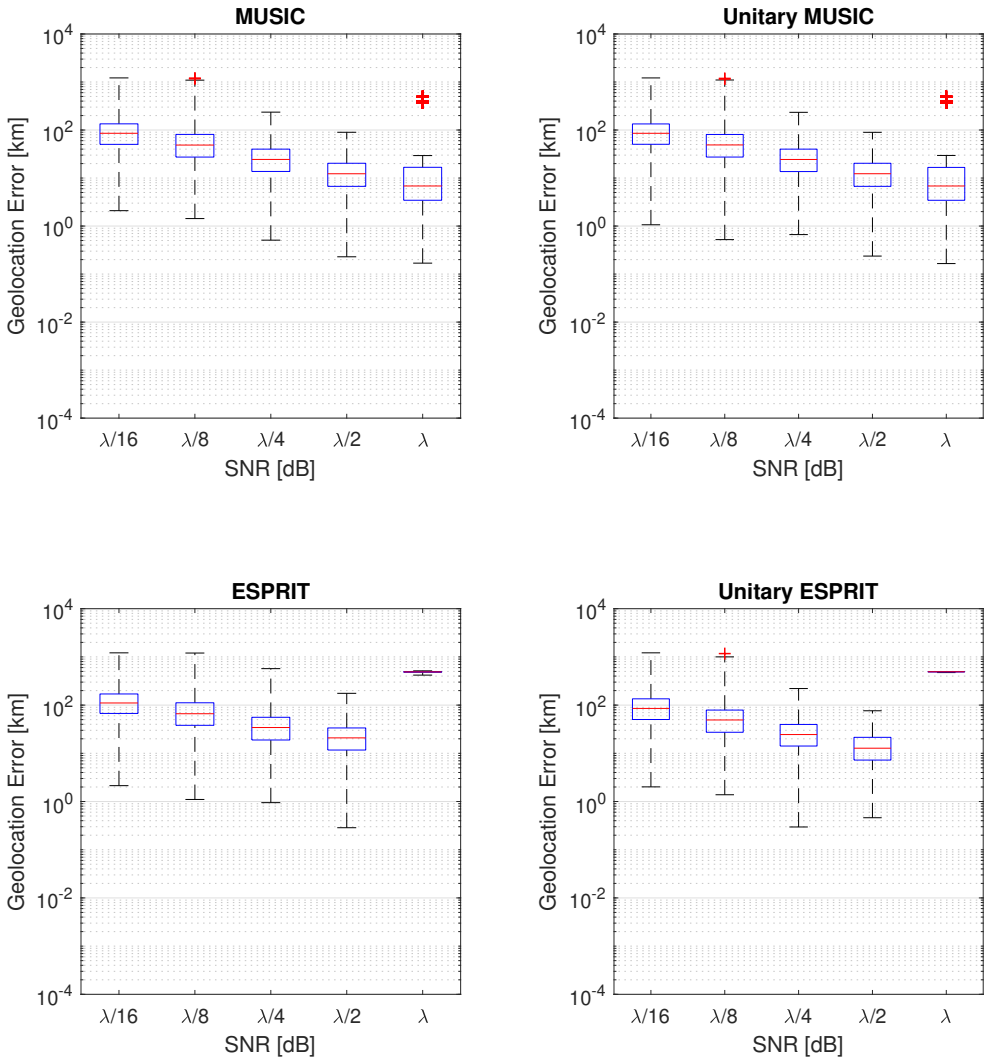


Figure 4.26: box plot of Geolocation error vs element spacing

Figure 4.26 presents the box plots of the resulting Geolocation Error. The behaviour of the boxplots is again

similar to the other analyses. Increasing or decreasing the element spacing, causes the median geolocation error to either shift up or down. The overall range of the errors remains roughly the same and the distribution of the error appears slightly skewed downwards. It is observed that when the elements spacing exceeds $\lambda/2$, the ESPRIT algorithms performance extremely degrades, whereas the MUSIC algorithms is still able to produce some accurate results, however it still degrades.

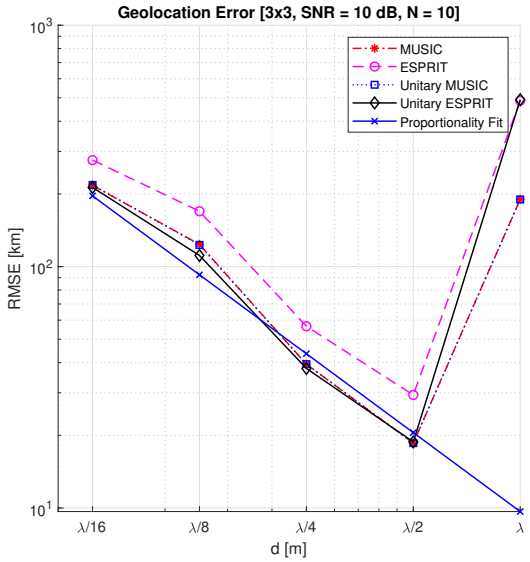


Figure 4.27: Geolocation RMSE vs Element Spacing

M [-]:	$\lambda/16$	$\lambda/8$	$\lambda/4$	$\lambda/2$	λ
ESPRIT: [km]	276.14	169.57	56.61	29.39	486.72
unitary ESPRIT: [km]	212.73	111.20	37.87	18.79	491.03
MUSIC: [km]	216.79	123.67	39.42	18.57	190.09
unitary MUSIC: [km]	217.97	122.97	39.44	18.57	189.70

Table 4.11: Geolocation RMSE vs element spacing

One can observe that for an element spacing $< \lambda/2$ MUSIC and unitary ESPRIT preform similarly. The RMSE reaches a minimum (the performance reaches a maximum) at an element spacing of $d = \lambda/2$. One can observe that the RMSE is linear in log-log scale for $d = \lambda/2$, suggesting a proportionality. This proportionality has been calculated as RMSE $\propto d^{-1.086}$ and is shown in the figure. When the element spacing exceeds this ($d > \lambda/2$) ambiguity is added to the phase results and the performance reduced drastically as expected. However, the ESPRIT performance reduces more drastically then the MUSIC algorithms.

The run-time of the algorithms is also analyzed over a change of element spacing d. This can be found in figure 4.28 and table 4.28. From these results one can observe that the algorithm run-times appear to be decoupled to the element spacing. This suggests that a change in element-spacing would not lead to a change in run-time.

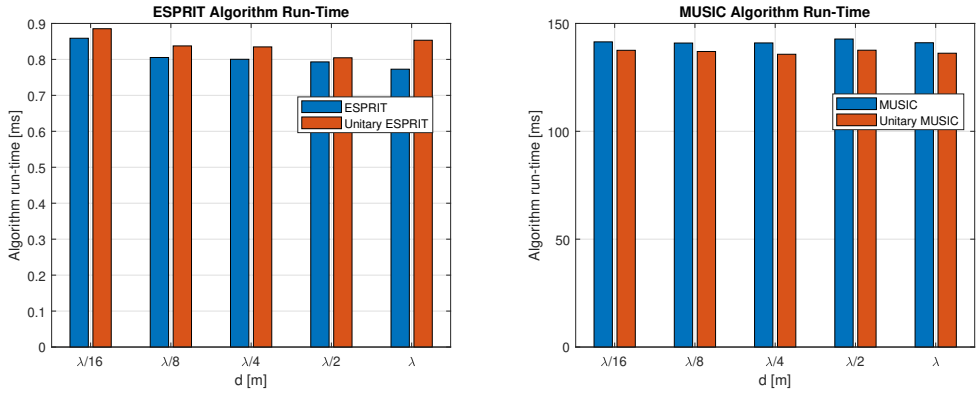


Figure 4.28: Algorithm timing vs Element Spacing

M[-]	$\lambda/16$	$\lambda/8$	$\lambda/4$	$\lambda/2$	λ
ESPRIT [ms]	0.86	0.81	0.80	0.79	0.77
Unitary ESPRIT [ms]	0.89	0.84	0.83	0.80	0.85
MUSIC [ms]	141.46	140.92	140.97	142.78	141.06
Unitary MUSIC [ms]	137.56	137.01	135.72	137.60	136.20

Table 4.12: Algorithm timing vs Element Spacing

It is investigated whether the behaviour of the performance of the geolocation due to element spacing is independent on other parameters. This is achieved by changing other parameters and comparing the behaviour. The system has been simulated again but with an increased the amount of elements, Signal-to-Noise and number of samples, leading to the results in figure 4.29. As one can observe, the linear behaviour on log-log scale still approximately holds, this suggests that the proportionality RMSE $\propto d^{-1.086}$ would hold independently of the other parameters.

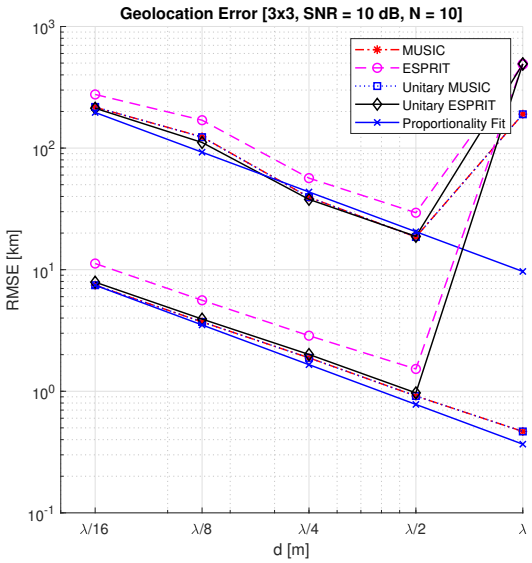


Figure 4.29: Geolocation RMSE vs Element Spacing

A trade-off in the geometric design of the antenna array, is element spacing versus number of elements. When more elements occupy the same space, they are more compactly packed, hence decreasing the element spacing. An increase in elements should allow for an increase of performance, however the decrease of element spacing should cause a decrease of performance. Observing the proportionals calculated for these

parameters(RMSE $\propto d^{-1.086}$ and RMSE $\propto M^{-2.15}$) one can derive that increasing the amount of elements would cause a greater performance gain than the performance loss due to a decrease of element spacing.

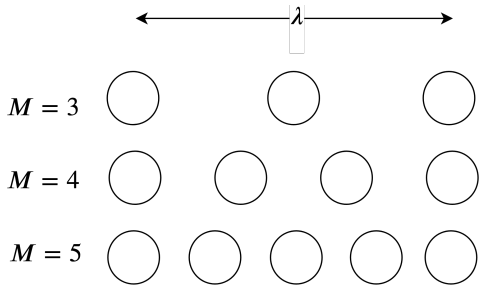


Figure 4.30: Schematic of increase of M causing decrease in d

A scenario of this decrease in element spacing can be observed in figure 4.30. In this particular case, the element spacing is optimal ($d = \lambda/2$) when the amount of elements is $M = 3$. This particular scenario has been simulated in order to compare whether an increase in M but decrease in d would cause a performance gain or loss. The results of this simulation can be observed in table 4.13 and figure 4.31. One can observe that an increase of elements indeed causes an increase of performance (nonetheless that a decrease in spacing causes a small loss) as expected.

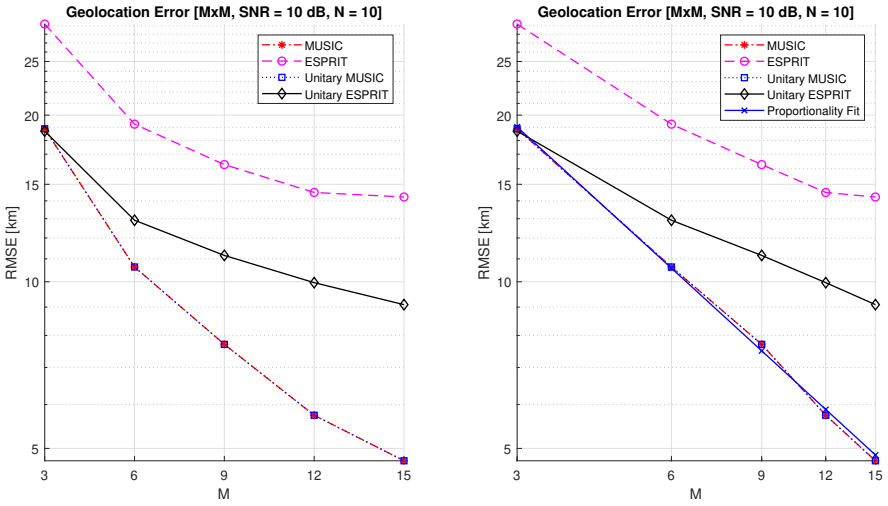


Figure 4.31: Geolocation RMSE vs increasing M, decreasing d

M [-]:	3.00	6.00	9.00	12.00	15.00
ESPRIT: [km]	29.21	19.26	16.28	14.50	14.23
unitary ESPRIT: [km]	18.71	12.92	11.16	9.96	9.09
MUSIC: [km]	18.89	10.64	7.71	5.74	4.75
unitary MUSIC: [km]	18.89	10.63	7.71	5.74	4.75

Table 4.13: Geolocation RMSE vs increasing M, decreasing d

When plotting on log-log scale, the performance shows a linear relationship as expected. The slope of this relationship can be estimated by using the obtained proportionalities found for parameter d and M. The estimation of the slope is $(-2.15 + 1.086 =) - 1.0640$, the slope approximated from the simulation is -0.8478 which is within reasonable range of the estimation.

It is important to note that there are methods available for increasing performance when the element spacing

exceeds $\lambda/2$. An example of these methods is antenna array interpolation, by creating extra 'virtual' antenna array elements in-between the largely spaced real antenna array elements, one can reduce the RMSE. [8]

4.4. Performance vs Satellite Platform Parameters

Besides parameters of the geolocation payload affecting the performance of the geolocating process. The satellite platform itself also influences this performance. Satellite parameters affecting the geolocation process consists of extra added noise and inaccuracies. The two major additions of inaccuracies are those in the pointing (noise is added to the satellite orientation during AoA to Geographic location conversion) and in the position (extra noise is added to the location of the satellite during the Angles-to-Arrival to Geographic location conversion).

4.4.1. Performance vs Satellite Pointing Accuracy

Pointing accuracy of a satellite is mostly given as a 3σ figure. When using a complete Commercial-of-the-shelf Attitude Determination and Control System (ADCS) solution, the 3σ ranges from $0.1° - 1°$ degrees.

The pointing error angels can be modeled as an uncertainty in the orientation of the NED-frame. One can simulate this error by multiplying the NED-frame vectors by a rotation matrix with it's x- y- and z-angles a random distribution given by the 3σ figure. However, due to the pointing error angle being relatively small and the satellite orbital height being relatively large, an easy approximation can be done to obtain the added geolocation error induced by this uncertainty without a full simulation. This approximation is represented in one dimension in equation, where θ_{error} is the added pointing uncertainty in radians and h_{sat} the satellite height. (4.11) an figure 4.32

$$x_{\text{error}} = \theta_{\text{error}} \cdot h_{\text{sat}} = \sigma_{\text{pointing}} \cdot N(0,0) \cdot h_{\text{sat}} \tag{4.11}$$

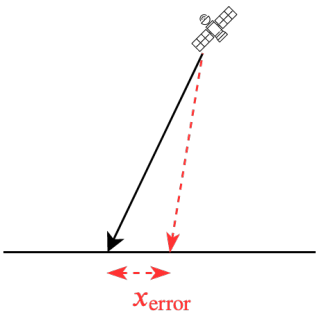


Figure 4.32: Error in pointing propagating to geographic error

This approximation needs to be extended as the uncertainty is in three dimensions. This means that the total approximated additive geolocation error caused by pointing uncertainty can be calculated according to equation (4.12)

$$e_{\text{error}} = \sqrt{x_{\text{error}}^2 + y_{\text{error}}^2 + z_{\text{error}}^2} = \sqrt{3}\sigma_{\text{pointing}} \cdot h_{\text{sat}} \cdot N(0,0) \tag{4.12}$$

Using this estimate, one can state that the total added RMSE error can be given by equation (4.13).

$$\text{RMSE}_{\text{pointing}} = \sqrt{3}\sigma_{\text{pointing}} \cdot h_{\text{sat}} \tag{4.13}$$

The RMSE added to the system due to a uncertainty in the pointing has been calculated for different values of 3σ and compared to a simulated value. This can be seen in table 4.14 and figure 4.33, one can observe that the approximated additive RMSE is very close to the 'actual' simulated value.

Accuracy Figure 3σ	Approximated RMSE	Simulated RMSE
0°	0	0
.2°	0.2828	0.2892
.4°	0.5657	0.5696
.6°	0.8485	0.8738
.8°	1.1314	1.1620
1°	1.4142	1.4299

Table 4.14: Distribution in pointing accuracy and how-much additive RMSE in geolocation

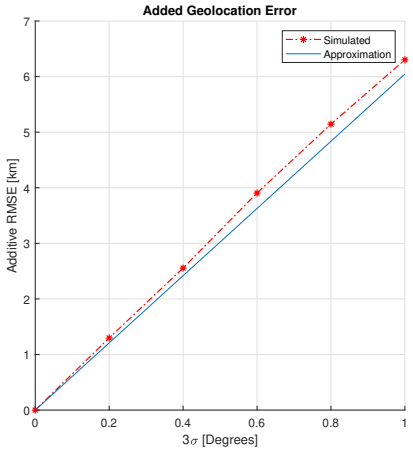


Figure 4.33: Additive Geolocation RMSE vs Pointing uncertainty 3σ

4.4.2. Performance vs Satellite Positioning Accuracy

Positioning accuracy of a satellite can also be given as a figure of variance. This can be for example standard deviation σ. This positioning uncertainty can be simulated by adding a distribution of error to the position of the origin of the NED frame (satellite position vector). Similarly to the case of pointing accuracy, an approximation can be obtained by first observing the additive error in one dimensional. The result of this one dimensional approximation can be seen in figure 4.34 and equation (4.14).

$$x_{\text{error}} = x'_{\text{error}} = \sigma_{\text{position}} \cdot N(0,0) \tag{4.14}$$

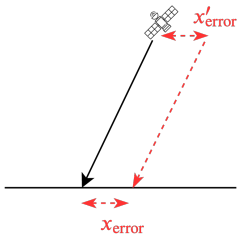


Figure 4.34: Error in position propagating to geographic error

This approximation can now be extended to three dimensions, since there is uncertainty in the x-, y- and z-direction. However, an error in the z-direction (height), should have minimal to no impact on the system due to the emitter always being assumed as earth surface. Hence, the error can be approximated according to equation (4.15) and (4.16)

$$e_{\text{error}} = \sqrt{{x'_{\text{error}}}^2 + {y'_{\text{error}}}^2 + {z'_{\text{error}}}^2} = \sqrt{{x'_{\text{error}}}^2 + {y'_{\text{error}}}^2} \tag{4.15}$$

$$e_{\text{error}} = \sqrt{2}\sigma_{\text{position}} \cdot N(0,0) \tag{4.16}$$

Meaning that the approximated added RMSE due to positioning error can be given by equation (4.17).

$$\text{RMSE}_{\text{position}} = \sqrt{2}\sigma_{\text{position}} \tag{4.17}$$

Simulating and calculating the added RMSE error for different values of σ one can observe whether the approximation is close to the 'real' simulated losses. The results of this simulation can be seen in table 4.15 and figure 4.35, one can observe that the approximated additive RMSE is very close to the 'actual' simulated value.

Accuracy Figure σ	Approximated RMSE	Simulated RMSE
$0km$	0	0
.3°	1.2092	1.2975
.5°	2.4184	2.5553
.7°	3.6276	3.9049
.9°	4.8368	5.1449
1.1°	6.0460	6.3019

Table 4.15: Distribution in pointing accuracy and how-much additive RMSE in geolocation

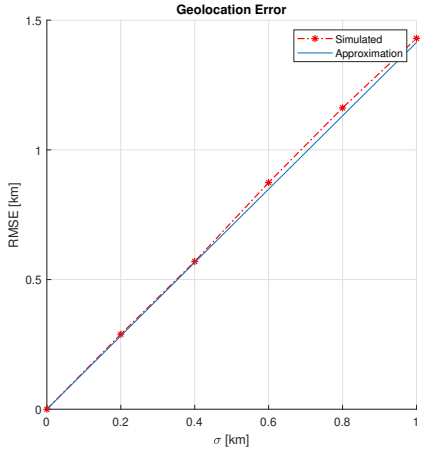


Figure 4.35: Additive Geolocation RMSE vs positioning uncertainty σ

4.5. Summary of Sensitivity Analysis

After the analysis, all parameters are listed and their respective optimization approach and effect on RMSE is given. The optimization approach can either be to maximize, minimize, or to set the value to an optimum. This overview can be seen in table 4.16

Symbol	Parameter Description	Optimization	RMSE Proportionality
	Payload Parameters		
Z	Additive Noise ratio of received Signal	minimize (maximize SNR)	$\propto 0.89^{\text{SNR}}$
N	Number of Samples	optimum(based on f_s and orbit)	$\propto N^{-0.46}$
M	Amount of elements ($M \times M$)	maximize	$\propto M^{-2.15}$
d	antenna array element spacing	optimum ($d = \lambda/2$)	$\propto d^{-1.086}$
	Satellite Parameters		
θ_{error}	Pointing Error of satellite platform	minimize	$+\sqrt{2}\sigma_{\text{position}}$
x_{error}	Positioning Error of satellite platform	minimize	$+\sqrt{3}\sigma_{\text{pointing}} \cdot h_{\text{sat}}$

Table 4.16: Investigated parameters with their respective effect and optimization approach

The performance of the geolocation can be represented as afigure of merit, an example of this is to take the inverse of the RMSE. This figure of merit can be graphed for the design parameters versus the design

parameter value. By using this graph one could identify which design parameter has the biggest impact on the figure of merit. This graph can be seen in figure 4.36.

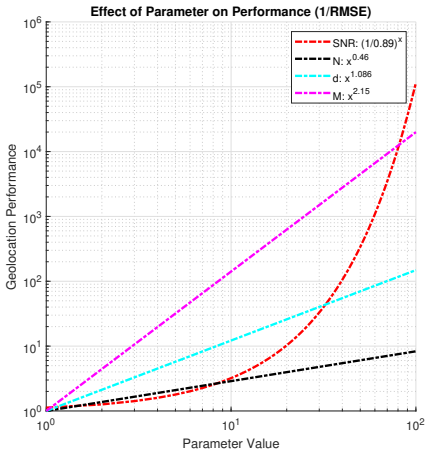


Figure 4.36: Effect on design parameter x on geolocation performance

Using this one can see that the influence of the SNR on the performance of the geolocation on log-log scale is exponential, whereas the other payload parameters are linear. This is expected behaviour due to the found proportionalities and the linearity on semi-log scale instead of log-log scale. However, one should realize that the unit of the SNR parameter is in dB, which is an exponential unit by itself. When excluding the effect of the SNR, the importance of the design parameters can be ranked based on influence on figure of merit which is M, followed by d, followed by N.

4.6. Recommendation of Design Synthesis

Using the analysis and prioritization provided this chapter, one can define the steps to maximize the performance of the URA Array for AoA based Geolocation. Assuming that the satellite platform, orbital parameters and the antenna element technology are known, one can take the following steps in order to maximize performance.

1. Investigate the frequency (range) of the emitters of interest, since this frequency (range) defines the target wavelength.

2. Calculate the maximum allowed element spacing to $d = \lambda_{min}/2$.

3. Maximize Amount of array elements $M \times M$ based on maximum available power, available complexity or available surface Space keeping the maximum allowed element spacing into account.

4. Space the obtained elements as optimally as possible to eachother ($d \rightarrow \lambda_{min}/2$).

5. Make sure that $M \times M > D$ (Equation (4.10)) or design is infeasible.

6. Find optimum point of amount of samples N by simulating with respective sample frequency and orbital parameters.

7. If quick run-time is a requirement → select Unitary ESPRIT Algorithm.

8. If run-time is not a requirement and ($M^2 > 2D$), the MUSIC algorithm should be selected.

These steps can be translated into a flowchart, leading to figure 4.37.

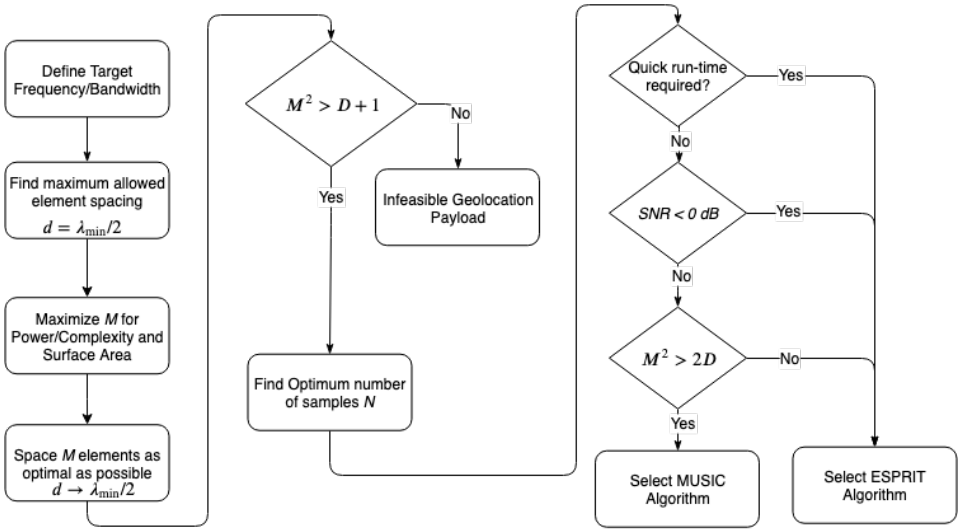


Figure 4.37: Overview of design steps to maximize Geolocation performance

5

Test Case: The design of a geolocation antenna array

The obtained sensitivity analysis and design synthesis approach can be used to design an antenna array payload for a satellite mission. As a demonstration, an example of the design process for a proposed mission is showcased in this chapter.

5.1. Proposed mission scenario

The mission proposed involves space-based geolocation of earth-surface interference sources in order to find unlicensed emitters broadcasting on a specific reserved frequency. The mission consist of multiple assets, one of which is a smallsat in low sun synchronous low earth orbit. This satellite's purpose is to do global course geolocation of interference sources of a specific known frequency. When an interference source is roughly localized, local authorities and other more responsive assets can take further action. Below one can find an example of the requirements set by the mission provider. The antenna array payload designed for this mission is the so called 'GLAP' meaning 'Geolocating Array Payload'. It is assumed that no more then two interference sources would cluster up to each-other in close geographic proximity.

REQUIREMENT	Description
REQ-GLAP-TRACK-01	The GLAP shall geolocate interference sources from space
REQ-GLAP-TRACK-02	The GLAP shall geolocate with a Root-Mean-Square Error of less or equal to 100km
REQ-GLAP-TRACK-03	The GLAP shall geolocate interference sources of 10dBm or more
REQ-GLAP-TRACK-04	The GLAP shall be able to geolocate up to four interference sources simultaneously
REQ-GLAP-TRACK-05	The GLAP shall geolocate interference sources with a frequency of 3 GHz
REQ-GLAP-MECH-01	The GLAP shall not exceed the dimensions of a 6U surface payload
REQ-GLAP-PWR-01	The GLAP shall not exceed power usage of 15W when active

Table 5.1: Mock Mission Requirements

During the early design phase of the satellite payload within this 'mock scenario', the antenna geometry selected are so called patch-antennas (microstrip antenna), Due to the slim design making it easy to integrate on a cubesat surface and it's easy availability in this specific project. Another advantage of this particular antenna is that it's directivity pattern is maximum at $\theta = 0$.

5.2. Satellite and Antenna Array Parameters

Following the previous' chapter's defined steps, initially the wavelength is investigated.

1. The only frequency of interest is $f_c = 3GHz$, which wavelength is $\lambda = \frac{c}{f} = \frac{3 \cdot 10^8}{3 \cdot 10^9} = 10$ cm
2. The optimal (and maximal) element spacing can be calculated to $d_{\text{optimal/maximal}} = \lambda/2 = 5$ cm.

3. The surface space available is 6U (30cm x 20cm) surface (REQ-GLAP-MECH-01). Initially the dimensions of a single antenna element should be investigated. An example of a simple inset feed patch antenna can be seen in figure 5.1.

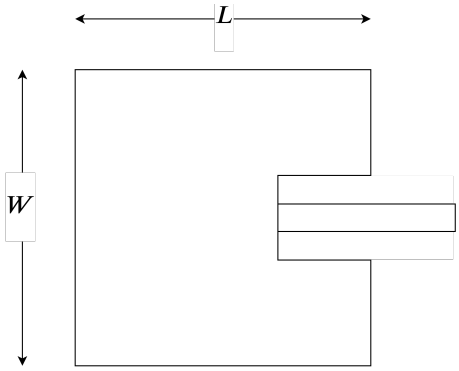


Figure 5.1: schematic of inset feed patch antenna

The primary dimensions of such a patch antenna are the width (W) and length (L). The sizing of these dimensions are primarily based on the designed sensitive frequency (the center frequency of the signal) of the permittivity of the dielectric substrate the patch antenna is printed on. A simple initial value for W and L can be empirically approximated using equation (5.1) and (5.2)[16].

$$W = \frac{\lambda}{2\sqrt{0.5\epsilon_r + 1}} \tag{5.1}$$

$$L = \frac{c}{2f_c\sqrt{\epsilon_r}} \tag{5.2}$$

where λ is the wavelength, c is the speed of light, f_c is the designed center frequency and ϵ_r is the substrate permittivity.

A common substrate used for patch antennas is an FR4 substrate, which permittivity is $\epsilon_r = 4.4$[16]. Subsequently, a patch antenna on this substrate designed for a center frequency of 3 GHz (REQ-GLAP-TRACK-05) will have the dimension of approximately $W = 30.4$ mm and $L = 23.8$ mm. Assuming a power consumption of 0.5W (only Receive Capability) per element, implementing 24 antenna array elements results in a power consumption of 12W, leaving 3W for a data processing unit. (REQ-GLAP-PWR-01)

4. When positioning the elements on the surface, one should make sure to leave space for possible connectors, while trying to approach the optimal element spacing (50 mm). An optimally spaced antenna array can be designed for a 6U surface using a URA of 6×4 elements which also allows for enough spacing for connectors and wires. This antenna array design can be seen in figure 5.2.

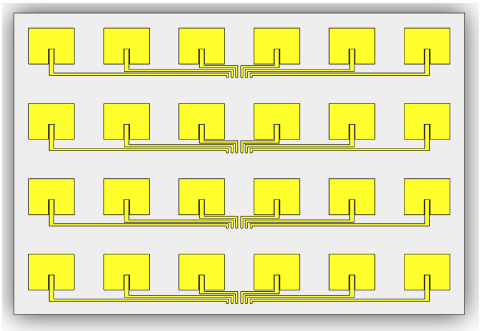


Figure 5.2: Proposed Design of Patch Antenna Array

5. 24 Elements is sufficient for geolocating four targets simulaneously, as 24 elements theoretically allows for up to 23 targets. (REQ-GLAP-TRACK-04)

6. A commercial ADC converter can easily sample the signals with a sample frequency of >2MHz, meaning that due to the orbital-height, the sampling rate and satellite movement drift will not have a big impact on the performance (figure 4.13). Therefor, the amount of samples should be maximized and depends on available processing resources and broadcast time of the interference source. For example, if the interference source only broadcasts for 5ms when in view, the maximum amount of obtainable samples is $N = f_s \cdot T = 8000$.

7. No statement is made about maximum allowed processing time, and thus is assumed that this is not of importance. If processing time ends up being an important factor, one should use the Unitary ESPRIT Algorithm

8. Due to the high amount of elements in the array compared to the maximum amount of expected incoming signals (`REQ-GLAP-TRACK-04`), the MUSIC is the algorithm should give the best performance. $(6 \times 4 > 2 \cdot 4)$

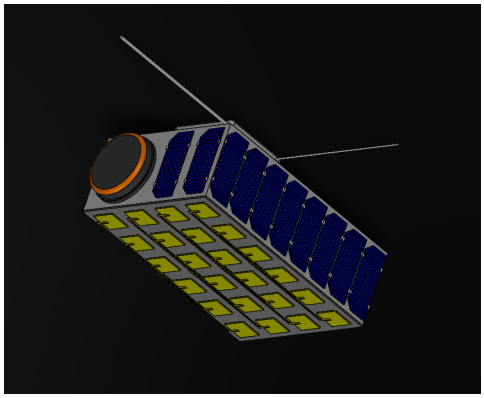


Figure 5.3: Example Design of geolocation Satellite using the GLAP

The last unknown parameter for the simulation is the Additive Noise Z, calculated from the SNR. However this can be calculated in similar fashion to chapter 4. Two important parameters for this are however, the antenna element directivity and the equivalent noise temperature. These elements can be found by either extensive simulation or analysis of the complete chain, or by measurements after implementation. For now an assumption will be made of a directivity of 5 dBi (which is considered average for patch antennas [16]) and an equivalent noise temperature of 100K (which is considered normal for satellite systems [25]). The SNR calculated is the worst-case scenario, meaning that the incoming AoA of the interference source is $\theta = 30°$ which is the maximum assumed angle, thus:

- has the longest path and thus the maximum free-space loss

- is considered the half-angle of the antenna array, thus causing a pointing loss of 3 dB

Transmit Parameters	
1. Transmitted EIRP(10W)	10 dBW
Losses	
3. Free-Space-Loss (f=3GHz,H=600km,θ=30°)	158.67 dB
4. Pointing Loss	3 dB
Receiver Parameters	
5. Receiver directivity (Patch antenna)	5 dBi
6. Noise Temperature	100 K
7. Max Bandwidth (50kHz)	47 dB-Hz
8. Noise Spectral Density	-208.6 dBW/Hz
Power Summary	
9. Received Power (1-3-4+5)	-149.68 dBm
10. Received Noise (7+8)	-161.6 dBm
SNR (9-10)	11.9 dB

Table 5.2: Calculation SNR for Mission Scenario

5.3. Performance of Proposed Satellite

Lastly, the AoA tool can be used to evaluate the performance of the geolocation process. The antenna array parameters obtained in previous' section are given by table 5.3.

Symbol	Parameter Description	Value
	Payload Parameters	
Z	Additive Noise ratio of received Signal	SNR = 11.9 dB
N	Number of Samples	8000
M	Amount of elements ($M \times M$)	4×6
d	antenna array element spacing	5 cm

Table 5.3: Selected GLAP design parameters

These design parameters can be entered in the tool shown in figure 5.4. The tool also requires a satellite and emitter location, in this example the location selected are shown in table 5.4. Finally, the predicted RMSE of the geolocation system are given in table 5.5.

Symbol	Parameter Description	Value
	Location Parameters	
(λ_s, ψ_s)	Satellite Coordinates	Eindhoven = (5.47°, 51.4°)
h_s	Satellite Height	600 km
(λ_1, ψ_1)	Emitter Coordinates	Paris = (2.35°, 48.86°)

Table 5.4: Selected scenario location Parameters of Analysis

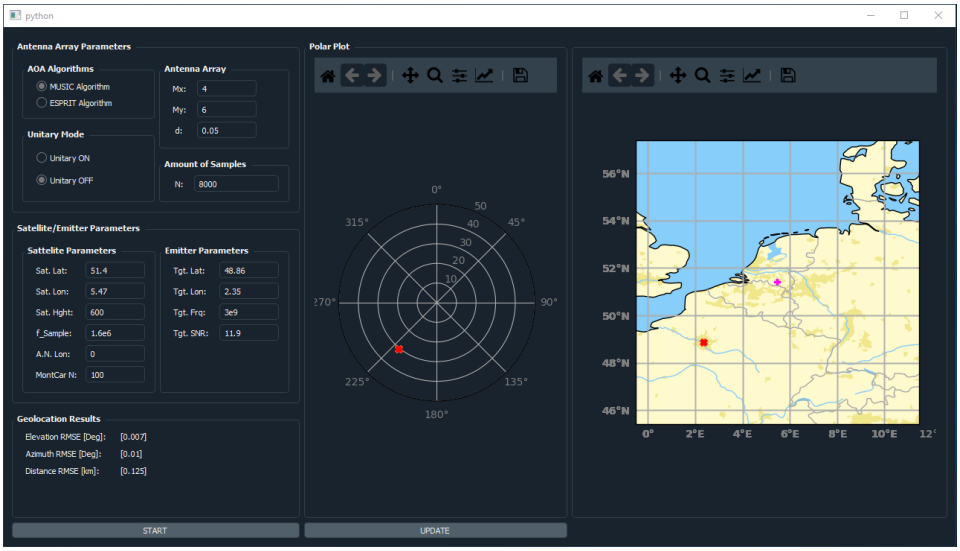


Figure 5.4: Analysis of Geolocation Accuracy using the AOA Tool

Error Description	RMSE Value
Elevation Angle	0.007°
Azimuth Angle	0.01°
Distance to Target	125m

Table 5.5: Results of RMSE after simulation

6

Conclusion

6.1. Summary and Key Outcomes

The research question of this thesis is to estimate the performance figures of Space-based Geolocation using AoA, and to investigate the effect is of different design parameters on this performance. The algorithms required for geolocation are investigated, explored and a simulation framework is produced. The simulation framework produced contains a front-end GUI allowing a user to interact with the simulation framework without thorough understanding or knowledge about the Mathematics or specific algorithms. By using the obtained simulation framework, multiple design parameters have been investigated and clear intuitive correlations are found between design parameters and geolocation accuracy. These correlations are put together in an overview for quick access. By using the obtained knowledge of the effects caused by design parameters on the performance of geolocation, a recommended procedure is obtained on how to synthesise an optimal design for a geolocation payload. The following statements can be made regarding the comparison between the ESPRIT and MUSIC algorithms and the simulation parameters.

- The effect of the payload parameters (Z, N, M and d) on the geolocation performance can be approximated.
- The Payload parameter importance, excluding the SNR, based on influence on performance is M, followed by d, followed by N.
- An array with more elements always performs better than an array with better spaced but less elements. (More elements is always preferred over better spacing)
- ESPRIT / Unitary ESPRIT performs much better at low SNR (SNR < 0dB).
- Unitary ESPRIT outperforms ESPRIT in *all* (simulated) cases. Hence, when the array is cyclo-symmetric, unitary transformation should always be performed.
- MUSIC performs much better then ESPRIT when the array has a higher amount of elements and not a high amount of incoming signals ($M^2 > 2D$)
- Unitary transformation produces no significant benefit to the MUSIC algorithm when dealing with exclusively incoherent signals.
- ESPRIT algorithms allow for much faster run-time then MUSIC Algorithms.
- The ESPRIT algorithms perform better on single-snapshot measurements.

6.2. Recommendations for Future Work

This project is only an initial exploitative research in space-based geolocation. Further study is necessary in order to fully obtain the most optimal performance for space based geolocation. Approaches for this further study can be to:

- Create or develop an actual geolocation payload that uses the proposed algorithms, in order to fully validate the obtained accuracy figures.
- Investigate more versions of the MUSIC and ESPRIT algorithms (such as root-MUSIC).
- Investigate more (pre-)processing schemes, such as spatial smoothing and transformations to the Beamspace domain [2]
- Investigate different AoA algorithm such as Maximum-Likelihood algorithms.
- Extend the simulation by introducing more Noise figures and parameters, such as earth oblateness and implementation losses.
- Investigate different Antenna Array Geometries, such as a uniform circular array, a filled circular array or a more three dimensional L-shaped array.
- Extend the algorithms to include multiple observations (from maybe different agents) using, for example, Kalman Filters or Particle Filters (State Observers).
- Extend the algorithms to fuse multiple satellite agents, flying in constellation or formation.
- investigate different geolocation techniques such as TDOA and FDOA.

Bibliography

[1] Taken koninklijke luchtmacht. `https://www.defensie.nl/organisatie/luchtmacht/taken`. Accessed: 16-04-2019.

[2] Zhizhang Chen, Gopal Gokeda, and Yiqiang Yu. *Introduction to Direction-of-arrival Estimation*. Artech House, 2010.

[3] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2009.

[4] Howard D Curtis. *Orbital mechanics for engineering students*. Butterworth-Heinemann, 2013.

[5] Tanuja S Dhope. Application of music, esprit and root music in doa estimation. *Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia*, 2010.

[6] Jesse James Garrett. *The elements of user experience: user-centered design for the web and beyond*. Pearson Education, 2010.

[7] Martin Haardt and Josef A Nossek. Unitary esprit: How to obtain increased estimation accuracy with a reduced computational burden. *IEEE transactions on signal processing*, 43(5):1232–1242, 1995.

[8] Per Hyberg, Magnus Jansson, and Björn Ottersten. Array interpolation and doa mse reduction. *IEEE Transactions on Signal Processing*, 53(12):4464–4471, 2005.

[9] Chen Jian, Shuxun Wang, and Lin Lin. Two-dimensional doa estimation of coherent signals based on 2d unitary esprit method. In *2006 8th international Conference on Signal Processing*, volume 1. IEEE, 2006.

[10] Rameshwar Kawitkar. Performance of different types of array structures based on multiple signal classification (music) algorithm. In *2009 Fifth International Conference on MEMS NANO, and Smart Systems*, pages 159–161. IEEE, 2009.

[11] Ramdas Kumaresan and Donald W Tufts. Estimating the angles of arrival of multiple plane waves. *IEEE Transactions on Aerospace and Electronic Systems*, (1):134–139, 1983.

[12] Wiley J Larson and James Richard Wertz. Space mission analysis and design. Technical report, Torrance, CA (United States); Microcosm, Inc., 1992.

[13] Tukaram Baburao Lavate, VK Kokate, and AM Sapkal. Performance analysis of music and esprit doa estimation algorithms for adaptive array smart antenna in mobile communication. In *2010 Second International Conference on Computer and Network Technology*, pages 308–311. IEEE, 2010.

[14] Darel A Linebarger, Ronald D DeGroat, and Eric M Dowling. Efficient direction-finding methods employing forward/backward averaging. *IEEE Transactions on Signal Processing*, 42(8):2136–2145, 1994.

[15] Mingqian Liu, Junlin Zhang, Jie Tang, Fan Jiang, Peng Liu, Fengkui Gong, and Nan Zhao. 2-d doa robust estimation of echo signals based on multiple satellites passive radar in the presence of alpha stable distribution noise. *IEEE Access*, 7:16032–16042, 2019.

[16] A Pandey. Practical microstrip and printed antenna design. *Norwood, MA: Artech House*, 2019.

[17] Marius Pesavento, Alex B Gershman, and Martin Haardt. Unitary root-music with a real-valued eigendecomposition: A theoretical and experimental performance study. *IEEE transactions on signal processing*, 48(5):1306–1314, 2000.

[18] MP Priyadarshini and R Vinutha. Comparative performance analysis of music and esprit on ula. In *2012 International Conference on Radar, Communication and Computing (ICRCC)*, pages 120–124. IEEE, 2012.

[19] C Carl Robusto. The cosine-haversine formula. *The American Mathematical Monthly*, 64(1):38–40, 1957.

[20] R Roy, A Paulraj, and T Kailath. Comparative performance of esprit and music for direction-of-arrival estimation. In *ICASSP'87. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 12, pages 2344–2347. IEEE, 1987.

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

[22] Michael Rubsamen and Alex B Gershman. Direction-of-arrival estimation for nonuniform sensor arrays: from manifold separation to fourier domain music methods. *IEEE Transactions on Signal Processing*, 57 (2):588–599, 2009.

[23] Ralph Schmidt. Multiple emitter location and signal parameter estimation. *IEEE transactions on antennas and propagation*, 34(3):276–280, 1986.

[24] Ralph Otto Schmidt. A signal subspace approach to multiple emitter location and spectral estimation. 1982.

[25] International Telecommunications Union. *ITU Handbook on Satellite Communications, Spanish Edition*. John Wiley & Sons, 2002.

[26] Pascal Vallet, Xavier Mestre, and Philippe Loubaton. Performance analysis of an improved music doa estimator. *IEEE transactions on signal processing*, 63(23):6407–6422, 2015.

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

[28] Michael D Zoltowski, Martin Haardt, and Cherian P Mathews. Closed-form 2-d angle estimation with rectangular arrays in element space or beamspace via unitary esprit. *IEEE Transactions on Signal Processing*, 44(2):316–328, 1996.