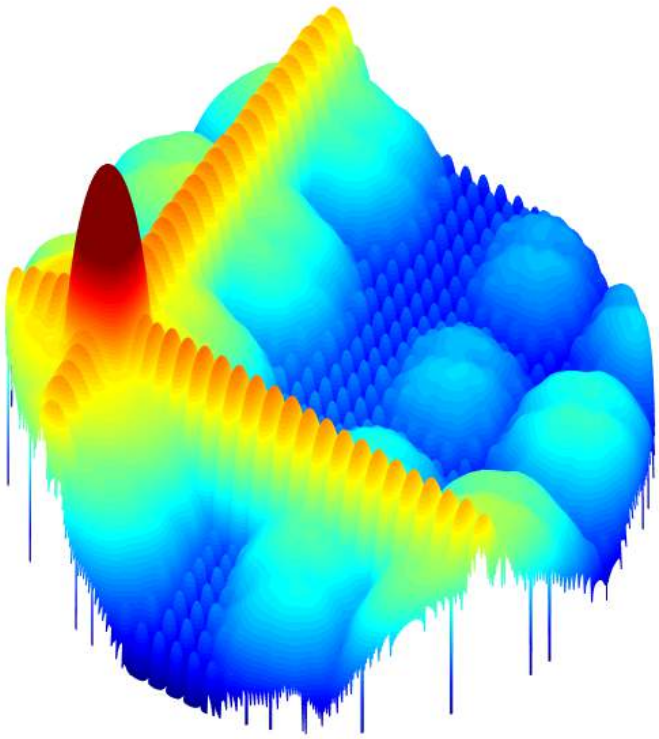
CHALMERS
UNIVERSITY OF TECHNOLOGY



Compensation of Element Position Errors in Electrically Scanned Arrays

Limits of beamforming calibration in analog and digital antenna arrays with scan dependant errors

Master's thesis in Wireless, Photonics and Space Engineering

JOSEF YDREBORG

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2019

Master's thesis 2019:30

Compensation of Element Position Errors in Electrically Scanned Arrays

Limits of beamforming calibration in analog and digital antenna arrays with scan dependant errors

JOSEF YDREBORG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Division of Communication and Antenna Systems
Antenna Systems Group
Chalmers University of Technology
Gothenburg, Sweden 2019

Compensation of Element Position Errors in Electrically Scanned Arrays
Limits of beamforming calibration in analog and digital antenna arrays with scan dependant errors
JOSEF YDREBORG

Supervisor: Bengt Svensson, SAAB AB
Examiner: Jian Jiang, Electrical Engineering, Chalmers University of Technology

Master's Thesis 2019:30
Department of Electrical Engineering
Division of Communication and Antenna Systems
Antenna Systems Group
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Radiation pattern constructed in Matlab of a 32x32 sensor array with translational subarray errors and a scan angle of 45 degrees in θ and ϕ.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2019

Compensation of Element Position Errors in Electrically Scanned Arrays
Limits of beamforming calibration in analog and digital antenna arrays with scan dependant errors
JOSEF YDREBORG
Department of Electrical Engineering
Chalmers University of Technology

Abstract

The consequences of mechanical translational errors on the radiation pattern in sensor and antenna arrays are investigated and discussed. A method for restoring the radiation pattern to the ideal case has been tried and the limits of its capabilities explored in different scenarios reflecting real antenna configurations. A proof of concept has been established by manufacturing an intentionally erroneous antenna array and applying investigated calibration methods. Conclusion is that the possibilities of compensating for mechanical errors is greatly enhanced with increased digitalisation of the array antenna system. Analog systems may achieve compensation in a smaller solid angle, while digital systems can achieve compensation over most of the half sphere of the array scanning area.

Keywords: antenna, array antenna, sensor array, signal processing, beamforming, optimization, antenna measurements.

Acknowledgements

I would like to thank my supervisor Bengt Svensson for his patience and willingness to always help out with my master thesis project. I would also like to thank my examiner Jian Yang for his positive attitude and for guiding me through the process of performing a master thesis at Chalmers University of Technology. The path to a degree has been a long and arduous journey and without my close friends at Chalmers it would have been a significantly more difficult time. Thank you for always making my time as a student a pleasure and a joy.

Josef Ydreborg, Gothenburg, July 2019

Contents

List of Figures

List of Tables

1
Introduction

1.1 Background

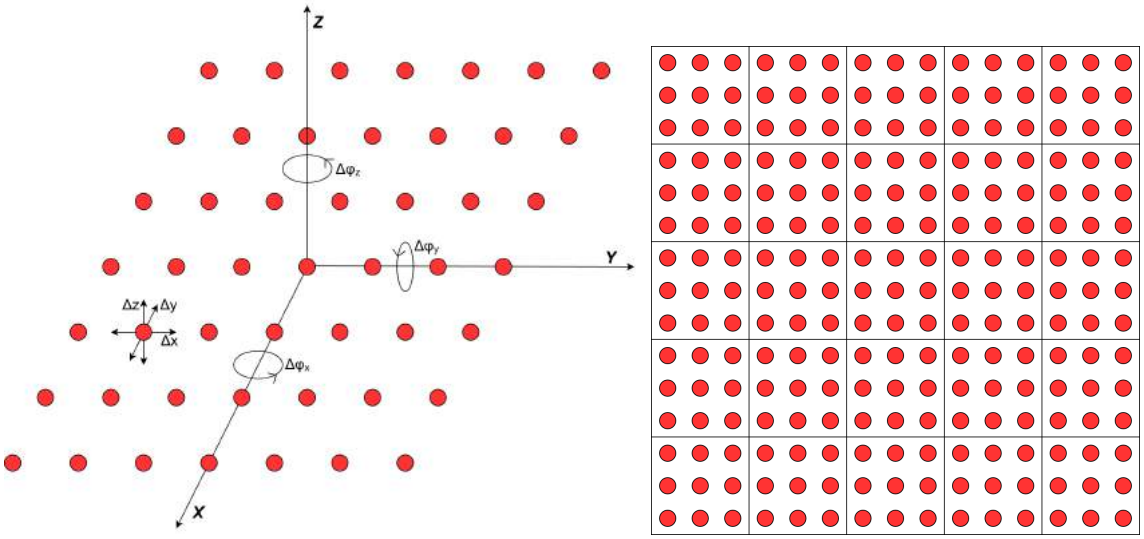
During operation an antenna array should exhibit a desired radiation pattern with sufficiently low sidelobes, realized through carefully chosen beamforming weights on the elements. However, when manufacturing an antenna array, mechanical errors are produced due to the manufacturing tolerances. These errors, in turn, have negative effects on the antenna radiation pattern produced by the array. Historically, the possibillities of mitigating these undesirable effects have been limited due to the analog nature of previous antenna arrays. However, with the rise of digitally defined array antennas, signal processing methods become more viable due to faster processing speeds and larger storage capabilities. As such, the tolerances can be maintained with increased performance or they can be reduced to lower the manufacturing costs. This report investigates the effects of the errors that occur due to mechanical tolerances and tries to mitigate them through signal processing methods.

1.2 Problem Description

Due to tolerances in the production of sensor arrays and wear due to usage, there will be a number of mechanical errors. These include the following (illustrated in Figure 1.1a):

- Translational errors
  - Δx, along the x-axis
  - Δy, along the y-axis
  - Δz, along the z-axis
- Rotational errors
  - $\Delta\varphi_x$, around the elements x-axis
  - $\Delta\varphi_y$, around the elements y-axis
  - $\Delta\varphi_z$, around the elements z-axis
- Structural distortion

This means that elements can be allocated away from the designed positions in three dimentions and also rotated around any of its three axes. Structural distortion means that the antenna shape is altered in the manufacturing or during use. An example of a distortion is when the antenna is bent creating a correlated rotational error across all elements.



(a) **(b)**

Figure 1.1: a) Possible mechanical errors in an array antenna
b) Example of a sensor array with 3x3 subarrays

In the case of smaller linear arrays it can be assumed that the elements have uncorrelated individual position errors, but when manufacturing larger arrays a more modular approach can be expected. This means that most large arrays are made up of subarrays that are manufactured separately and then assembled to form the complete antenna. As a consequence the subarrays are subject to the same types of errors as the elements. However, in this case the errors are correlated between all the elements in a subarray. An example of subarrays is shown in Figure 1.1b. The effects on the antenna performance due to mechanical errors are manifold. The most important effect is that the radiation pattern is altered. Additionally, there is a change of coupling between the elements. The main challenge with mitigating effects from mechanical tolerances is that the phase errors produced are angle dependent. This means that they cannot be adjusted with a simple phase shift at each antenna element and a more sophisticated method must be found.

1.2.1 Restrictions

This report will focus on handling translational errors and only briefly discuss the other types of errors. This is due to that the translational errors have a relatively higher effect on the radiation pattern of the antenna [1]. However, the approach layed out later in the report should be applicable to any mechanical errors. Additionally, there is reason to assume that an element's position error will affect the position of the neighboring element, but this problem is neglected in this work for simplicity. There is also the problem of coupling effects, but these are also neglected.

1.3 Methodology

The report is organized in three main segments. These are; error characterisation, error compensation, and trial antenna testing, as seen in Figure 1.2. The goal is for these three segments to form a partial guideline for what is important to determine before manufacturing an antenna array, given that the calibration method presented is used. Additionally, the methods handled in this report will serve as a good reference when a similar investigation with other parameters is attempted.

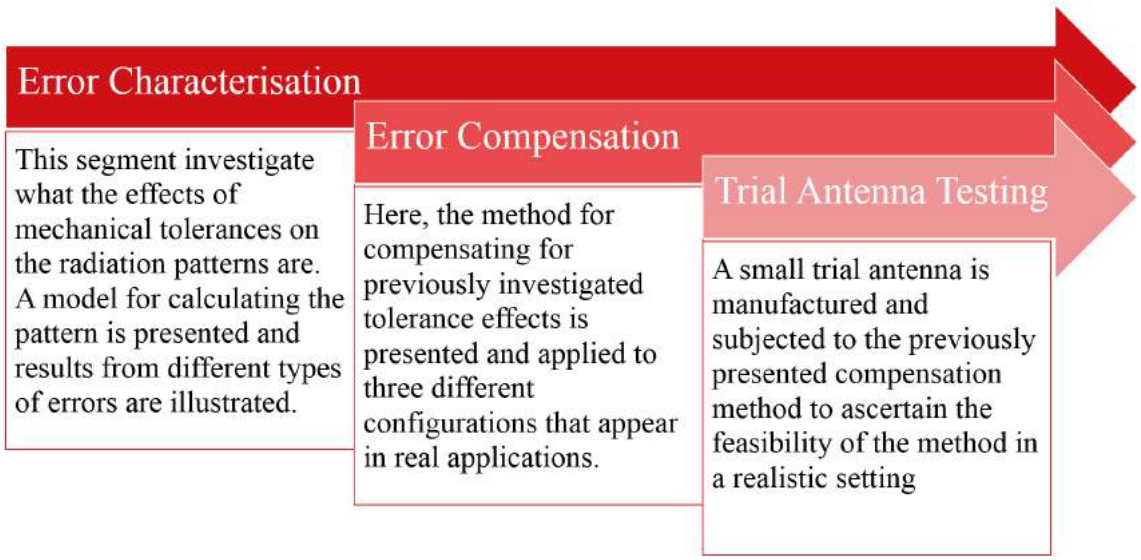


Figure 1.2: Structure of the report in three parts; error characterisation, error compensation, trial antenna testing.

2

Theory

2.1 Error Characterisation

2.1.1 Array Antenna Theory

The report mostly uses uv-coordinates which can be described based on spherical coordinates θ and ϕ and with $u^2 + v^2 + w^2 = 1$ as a requirement, as shown in Equation 2.1.

$$\begin{aligned} u &= sin(\theta)cos(\phi) \\ v &= sin(\theta)sin(\phi) \\ w &= cos(\theta) = \sqrt{1 - u^2 - v^2} \end{aligned} \tag{2.1}$$

The model used for calculating the radiation pattern of the antenna array can be expressed as [2][3][4]

$$G(u, v) = g(u, v) \sum_{n=1}^{N} A_n e^{j\Phi_n} e^{jk(x_n u + y_n v + z_n w)}, \tag{2.2}$$

where k is the wave number in free space, (x_n, y_n, z_n) are the cartesian coordinates for element n, and $A_n e^{j\Phi_n}$ is the excitation of the n^{th} element where A_n is the amplitude and Φ_n is the phase. This means that each element is modified by a complex number prior to operation. The amplitude excitation of each element is dictated by a weighting scheme chosen to create the desired beamforming. In our case a Taylor taper is used and is discussed in section 2.1.2. This will create a pencil beam at the scanning angle (u_0,v_0). The scanning phase of the n_{th} element is set as

$$\Phi_n = k(x_n u_0 + y_n v_0). \tag{2.3}$$

In analog systems the excitations of phase and amplitude are the parameters that can be modified to compensate for element position errors. When designing an array it is important to know what element spacing is required to avoid grating lobes. If we assume that the half power beamwidth (HPBW) of the grating lobes are λ/L we can express the element spacing criteria as [3]

$$d \leq \frac{\lambda}{1 + |cos(\alpha_0)| + (\lambda/L)}, \tag{2.4}$$

where α_0 is the maximum desired elevation scanning angle, and L is the length of the array. Furthermore the directivity of the designed antenna can be described by [3]

$$D = e_{grt} e_{pol} e_{ill} cos(\theta_0) D_{max}, \tag{2.5}$$

where e_{grt}, e_{pol}, and e_{ill} is the grating lobe, polarisation, and illumination efficiencies. θ_0 is the elevation scanning angle and D_{max} is the maximum possible directivity described by [3]

$$D_{max} = \frac{4\pi}{\lambda^2} A, \tag{2.6}$$

where A is the area of the array. It is worth noting that the directivity in Equation 2.5 is excluding the directivity of the antenna elements, assuming isotropic elements. This can be rectified by multiplying the directivity of the element type with the directivity of the array. Moving on from directivity the actual antenna will have a gain expressed by [3]

$$G = e_r e_{abs} D, \tag{2.7}$$

where e_r and e_{abs} are the mismatch factor and radiation efficiency.

2.1.2 Taylor Weighting Scheme

In order to achieve a certain SLL a Taylor line source is used to get the weighting values for each antenna element. For a linear array the values can be calculated through [5]

$$w(x) = F(0,\, A,\, \bar{n}) + 2 \sum_{m=1}^{\bar{n}-1} F(m,\, A,\, \bar{n})\, cos\left(\frac{2m\pi x}{L}\right), \tag{2.8}$$

where x is the position of the element in the intended dimension and L is the length of the antenna in the same dimension. $\bar{n}$ is a correcting integer and is set for maximum efficiency with a monotonic amplitude taper. The function $F(m, A, \bar{n})$ is defined as [5]

$$F(m,\, A,\, \bar{n}) = \frac{[(\bar{n}-1)!]^2}{(\bar{n}-1+m)!(\bar{n}-1-m)!} \prod_{n=1}^{\bar{n}-1} [1 - m^2/z_n^2], \tag{2.9}$$

where

$$\begin{aligned} z_n &= \pm\sigma(A^2 + (n-1/2)^2)^{1/2} && for\ 1 \leq n \leq \bar{n} \\ &= \pm n && for\ \bar{n} \leq n \leq \infty \end{aligned} \tag{2.10}$$

$A = (1/\pi) cosh^{-1} r$, where r is the desired ratio between the main lobe and the SLL and $\sigma = \bar{n}/[A^2 + (\bar{n} - 1/2)^2]^{1/2}$. For more infomation on how these equations are derived please see [5]. In order to find the suitable weight for an element in a uniform planar array (UPA) the weight function of two dimensions is separable as $g(x) \cdot g(y)$ for the x,y-positions of each element creating a Taylor sheet source.

2.1.3 Error Modeling / Statistics

All mechanical tolerances are assumed to follow a Gaussian probability distribution function (PDF) with zero mean and can be expressed as [6]

$$P_{norm}(x \mid \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} exp\Big(- \frac{(x-\mu)^2}{2\sigma^2} \Big), \tag{2.11}$$

where μ is the mean value and σ is the standard deviation. Each position error gives rise to a phase error that can be derived from Equation 2.2 and is expressed as

$$\delta\Phi = k(\delta x \cdot u + \delta y \cdot v + \delta z \cdot w), \tag{2.12}$$

where δx, δy, and δz are the position errors in the x, y, and z dimension. Using this information we can derive the total standard deviation of the phase error based on the standard deviation of the mechanical errors of each axis. Provided that the standard deviations follow $\sigma^2 << 1$ we can conclude that [1]

$$\sigma_\Phi^2 = k^2(\sigma_x^2 \cdot u^2 + \sigma_y^2 \cdot v^2 + \sigma_z^2 \cdot w^2), \tag{2.13}$$

The challenge with the phase error is that it varies with direction as seen in the changing variables (u,v,w) in Equation 2.12. This is called as 'direction dependant phase errors' and it makes the errors more difficult to compensate for. Even though the tolerances follow Equation 2.11 the resulting gain variance for a certain viewing angle, when subject to phase errors, does not. This quantity follows a so called Ricean distribution [5] and is modelled as [7][8]

$$P_{rice}(x \mid \nu, \sigma) = \frac{x}{\sigma^2} exp\Big(- \frac{x^2 - \nu^2}{2\sigma^2}\Big) I_0\Big(\frac{x\nu}{\sigma^2}\Big), \tag{2.14}$$

where σ is the scale parameter, ν is a noncentrality parameter, and I_0 is the Bessel function of the first kind with order zero. The Ricean PDF can, for errors much larger than the expected value, approximate a Rayleigh PDF [5]. This PDF is described by [6]

$$P_{ray}(x \mid \sigma) = \frac{x}{\sigma^2} exp\Big(- \frac{x^2}{2\sigma^2}\Big), \tag{2.15}$$

where σ is the scale parameter. As can be clearly seen, Equation 2.15 is Equation 2.14 with the noncentrality parameter $\nu = 0$. If the errors are much smaller than the expected value, then Equation 2.14 approximates Equation 2.11 [5].
Finding the standard deviation can, for normal distributions, be done by calculating the sample standard deviation from a number of samples according to [6][9]

$$s = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N-1}}, \tag{2.16}$$

where N is the number of samples, x_i is the i^{th} sample, and $\bar{x}$ is the average of all the samples. The standard deviation for the amplitude of the radiation pattern at different viewing angles can then be used to determine an upper bound with a certain probability of being beneath this bound. For a specific PDF this probability can be calculated by a cumulative distribution function (CDF). Given the Gaussian PDF the CDF is described by [6]

$$P_{CDF}(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-t^2/2}, \tag{2.17}$$

where x is the number of standard deviations above the mean. Using Equation 2.17 we can calculate the probability of being beneath the mean plus one, two, and three standard deviations to be 0.8413, 0.9772, and 0.9987 respectively. However, these probabilities are only for one viewing angle, or one point on the sphere. When considering all sidelobes together, a deeper analysis is required. See [5] for more details on this.

2.2 Error Compensation

There are several methods for compensating for direction dependent phase errors and here one of them will be presented. When you have a plane wave incoming with some direction of arrrival (DOA) in the surrounding sphere you can model it as [10]

$$\mathbf{x}(t) = \mathbf{a}_{mod}(\theta)s(t) + \boldsymbol{\eta}(t), \tag{2.18}$$

where $\mathbf{x}(t)$ is the received signal, $\mathbf{s}(t)$ is the actual signal, and $\boldsymbol{\eta}(t)$ is noise. $\mathbf{a}_{mod}(\theta)$ is the erroneous array response of the antenna with mechanical imperfections. It is dictated by Equation 2.2 and will vary for each element depending on its position and the DOA. We can write the erroneous array response as [10]

$$\mathbf{a}_{mod}(\theta) = \mathbf{Q}\mathbf{a}(\theta), \tag{2.19}$$

where $\mathbf{a}(\theta)$ is the ideal array response and Q is a correction matrix. In other words, the errouneous array response is assumed to be the ideal array response modified by the correction matrix, Q. The correction matrix is an $N \times N$ matrix, where N is the number of elements in your array. We know the ideal steering vector prior to calibration and must find the correction matrix in order to calibrate the array. If we want to choose Q in order to minimize the difference between the measured radiation pattern of an antenna and the modified ideal array response we will get an optimization problem of the following configuration called Global Calibration. [10]

$$\hat{\boldsymbol{Q}} = arg\min_{\mathbf{Q}} \|\mathbf{A}_{meas}(\boldsymbol{\theta}_{cal}) - \mathbf{A}_{mod}(\boldsymbol{\theta}_{cal})\|_F \tag{2.20}$$

The matrices $\mathbf{A}_{meas}(\boldsymbol{\theta}_{cal})$ and $\mathbf{A}_{mod} = \mathbf{Q}\mathbf{A}(\boldsymbol{\theta}_{cal})$ are the array response of each element arranged in columns for each of the calibration angles, $\boldsymbol{\theta}_{cal}$, for the measured antenna and the modified ideal array response respectively. $\boldsymbol{\theta}_{cal}$ forms a grid in uv-space or on the sphere surrounding the array. The resolution of the calibration grid needs to be finer than half of the half power beamwidth (HPBW) of the antenna. In other words, the calibration grid needs to be finer for larger arrays since they have narrower beamwidths. The goal is to choose the arguments of $\mathbf{Q}$ so that the Frobenious norm of the difference between the measured array response and the modified ideal array response is as small as possible. In general, there is no perfect solution where the Frobenius norm is zero, but if we treat the problem as a least squares problem of the form $\boldsymbol{xA} = \boldsymbol{B}$ with the solution $\boldsymbol{x} = \boldsymbol{B}/\boldsymbol{A}$ the solution to Equation 2.20 should be

$$\hat{\boldsymbol{Q}} = \mathbf{A}_{meas}(\boldsymbol{\theta}_{cal})/\mathbf{A}_{mod}(\boldsymbol{\theta}_{cal}). \tag{2.21}$$

This only works if the whole uv-space is sampled. Otherwise the ranks of the array response matrices will be severely deficient. If such is the case an optimization algorithm could be deployed instead. If some directions are more important to correct than others a weighting matrix could be added to Equation 2.20 to change the results of the Frobenius norm. This is called Local Calibration and the optimization problem is expressed as [10]

$$\hat{\boldsymbol{Q}} = arg\min_{\mathbf{Q}} \|(\mathbf{A}_{meas}(\boldsymbol{\theta}_{cal}) - \mathbf{A}_{mod}(\boldsymbol{\theta}_{cal}))\boldsymbol{W}^{1/2}\|_F, \tag{2.22}$$

where $\mathbf{W}$ is a diagonal matrix with the dimensions $M \times M$ where M is the number of calibration angles. As such each calibration angle is multiplied with a weight lending more significance to some angles than others. One important scenario is to find the DOA of a signal using DOA estimation algorithms such as MUSIC. In this case it is important that the directions close to the scanning angle are correctly calibrated. This means that we could shape our weighting matrix to give more significance to angles close to the scanning angle which would lead to the diagonal of the weighting matrix being described by [10]

$$w_j = exp(-hD_j^2), \quad D_j = |\theta_{cal,j} - \theta|, \tag{2.23}$$

where D_j is the distance to the scanning angle for the j^{th} calibration angle and h is a parameter determining the width of the weighting function. Using a weighting matrix has the consequence that the matrices no longer have full rank when trying to solve the problem through a least square method. However, if the weighting function is sufficiently narrow the problem can be solved by only calculating the diagonal values of the correction matrix Q. In other words, we treat the problem as a set of N independent equations. The calculation needed to solve this problem is [10]

$$\hat{q}_i(\theta) = \frac{\sum_{j=1}^{M} \boldsymbol{A}_{ij}^* w_j(\theta) \boldsymbol{A}_{meas,ij}}{\sum_{j=1}^{J} \boldsymbol{A}_{ij}^* w_j(\theta) \boldsymbol{A}_{ij}}, \quad i = 1, ..., N, \tag{2.24}$$

where $\hat{q}_i(\theta)$ is the i^{th} diagonal argument of the correction matrix and (i,j) denotes the indices of the matrices where N is the number of elements and M is the number of calibration angles. Once the correction matrix is estimated the recieved signal can be corrected as [10]

$$\boldsymbol{x}_{corr}(t) = \hat{\boldsymbol{Q}}^{-1} \boldsymbol{x}(t), \tag{2.25}$$

where $\boldsymbol{x}_{corr}(t)$ is the corrected signal. This correction can also be made by modifying the beamfroming weights and can be calculated as [10]

$$\boldsymbol{v} = (\hat{\boldsymbol{Q}}^{-1})^* \boldsymbol{a}(\theta_0), \tag{2.26}$$

where $\boldsymbol{a}(\theta_0)$ are the beamforming weights for the array without mechanical errors. The virtual array approach of Equation 2.25 and 2.26 is designed to produce an optimal beampattern, but does not handle noise very well. If we want to maximize the signal-to-noise ratio (SNR) we can modify the beamforming weights with [11]

$$\boldsymbol{v} = \hat{\boldsymbol{Q}} \boldsymbol{a}(\theta_0), \tag{2.27}$$

but here the beampattern is not as good as for Equation 2.26. If both channel noise and a poor beampattern are problems in a system a proper compromize between optimizing beampattern and maximizing SNR must be found [11].

3
Methods

3.1 Error Characterisation

When looking at previous literature the most significant degradation in the radiation pattern is the sidelobe level (SLL) [1]. The purpose of this section is to find out where and why the sidelobes are increased and how much they are increased for different translational errors. Of course, sources for effects on SLL is not limited to mechanical tolerances, and later in the report a discussion of how the findings can be combined with other error sources are discussed. Examples of other errors affecting the radiation pattern are internal phase and gain errors in the array (channel errors), coupling between elements, radome effects, edge effects, and interference from other radiation sources close to the array.

What distinguish the mechanical errors in the array elements and subarrays is direction dependence. This means that we can remove error effects perfectly in one direction, but since sidelobes are present throughout the whole space, the other sidelobes will not be compensated for. Therefore, an optimal compromise between manufacturing tolerances and compensation methods need to be found for each specific use case [1], which is why it is important to have a good picture of what the effects or mechanical errors are.

From papers as early as [12] from 1958, we know that rotational errors are significantly less impactful than translational errors. We also know that larger arrays with more elements experience less negative effects from individual element errors. As such, it is reasonable to assume that the greatest impact on performance in smaller arrays come from the translational errors of individual elements and in larger arrays from the translational errors in subarrays. For larger arrays, one can imagine that the subarrays are the antenna elements of a smaller array with more complicated radiation patterns. However, this assumption only works if individual elements errors are sufficiently small.

3.1.1 Antenna Array Setup

All of the investigation is done through simulations in MATLAB and for this segment a rectangular uniform planar array (UPA) is considered. The setup is summarized in Table 3.1. The results should also be broadly applicable to other planar arrays such as triangular or circular. In order to evaluate both correlated and uncorrelated errors the 32x32 element array will be made up of 4x4 element subarrays. Additionally, to

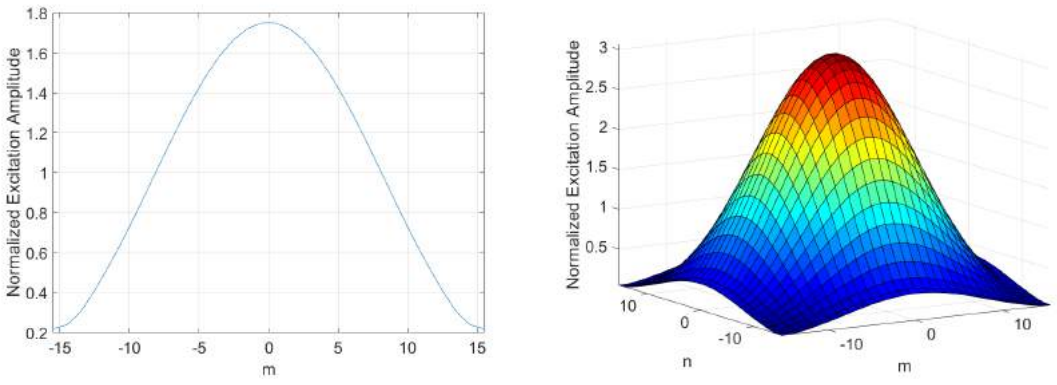
avoid grating lobes the nominal element spacing will be half of the wavelength in the dimensions of the plane. A relatively strict sidelobe level (SLL) will be maintained by applying a 40 dB Taylor taper. By this way, error effects will be amplified and easier to measure. The weighting scheme is illustrated in Figure 3.1. All lengths will be represented as the electrical length (in terms of the wavelength), but the main frequency span of interest is X-band, since this is the area where tolerances start to be more difficult to maintain. Lastly, the individual element radiation pattern is set to represent a realistic pattern comparable to real antenna elements as

$$g(u, v) = \sqrt{cos\theta} = \sqrt{w}, \tag{3.1}$$

for this particular case. A non-isotropic antenna element would also have a certain directivity. However, in this case the important results are the normalized directive gain of the antenna and not the maximum gain it can produce.

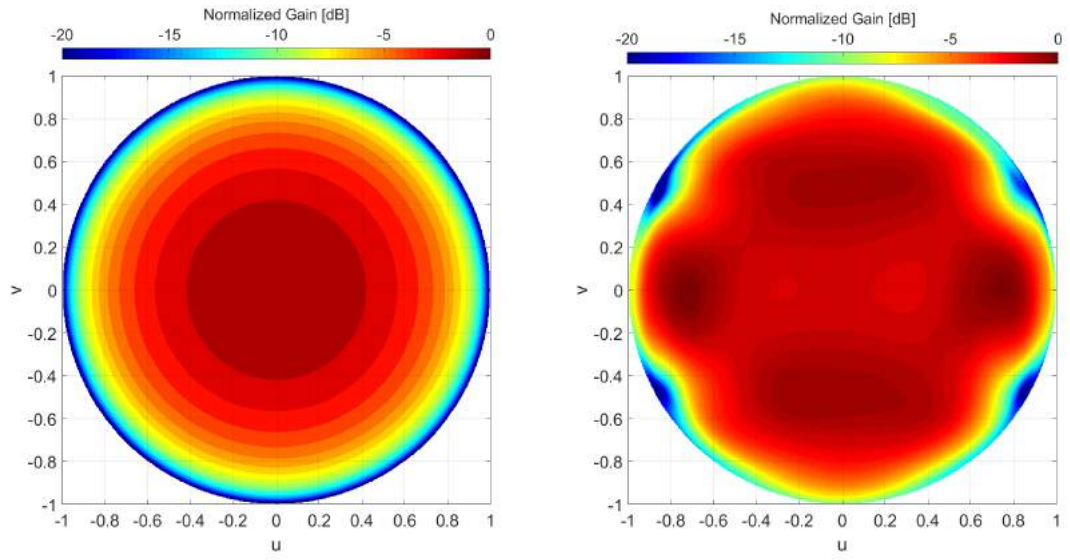
Table 3.1: Simulation setup for the antenna array.

Array type	Rectangular Uniform Planar Array (UPA)
Array size	32x32 (1024) elements
Subarray Size	4x4 (16) elements
Element spacing x-axis	$\lambda/2$
Element spacing y-axis	$\lambda/2$
Element radiation pattern	$\sqrt{cos(\theta)}$ (see Figure 3.2a)
Weighting scheme	Taylor 40 dB (see Figure 3.1)
Frequency	X-band



(a) **(b)**

Figure 3.1: 40 dB Taylor weighting scheme for (a) a 32 element ULA (b) a 32x32 element UPA.



(a) **(b)**

Figure 3.2: Radiation patterns of a) simulated $\sqrt{cos(\theta)}$ element and b) isolated trial antenna element (from HFSS simulation, see Figure 3.4a.)

3.1.2 Error Modeling Setup

The type of errors considered are translational errors in x, y, and z dimensions for both individual elements and subarrays. Rotational errors are not considered in this work since they are usually negligible compared to translational errors [12][1]. Additionally, it is assumed that the mechanical error of an element or subarray does not affect the position of the neighboring element or subarray. My investigation will focus on characterizing the effects caused by tolerances with a standard deviation of one percent of the wavelength, which is roughly comparable with realistic tolerances and will therefore constitute an appropriate reference [1]. All tolerances are assumed to follow a Gaussian distribution with zero mean. The effects of each error type will first be considered individually and then combinations of errors will be investigated. Each type of error will be simulated with boresight scanning and a scanning angle of $(\theta, \varphi) = (45, 45)$. Converting to uv-coordinates using Equation 2.1 we get two scanning angles $(u_0, v_0) = (0, 0)$ and $(u_0, v_0) = (0.5, 0.5)$.

3.1.3 Results Presentation

In order to get a comprehensive picture of the effects that translational errors have, the results of the model simulations will be presented in two formats. Firstly, the average radiation pattern based on 25 iterations of a 0.01λ mechanical standard deviation on a configuration of possible error types is calculated and presented as a contour plot. In this way we can see where the influence of that particular error type is most significant and to what extent it is affecting the SLL. Secondly, there will be cross sections through the main lobe of the significant sidelobes based on 500 iterations of the array with the same mechanical standard deviation. These will

illustrate the first three resulting standard deviations of the radiation pattern for each direction based on Equation 2.16. The assumption here is that the radiation pattern errors have an approximate normal distribution for the chosen mechanical standard deviation. The types of errors that will be presented are:

- Element errors
  - Δx errors
  - Δy errors
  - Δz errors
  - $(\Delta x, \Delta y, \Delta z)$ errors
- Subarray errors
  - Δx errors
  - Δy errors
  - Δz errors
  - $(\Delta x, \Delta y, \Delta z)$ errors
- $(\Delta x, \Delta y, \Delta z)$ errors for both elements and subarrays

Lastly, an analysis of the PDF of the points of the radiation pattern due to different sizes of mechanical standard deviations will be done in order to determine where we can assume different probability distributions when calculating the chances of exceeding a certain SLL.

3.2 Error Compensation

In order to compensate for the errors presented in section 3.1, we need to find a calibration scheme that can calibrate specific antennas to remove these errors. However, due to the impossibility of compensating for all directions optimally, there is a need for finding a solution as close as possible to the goal pattern which is the desired pattern for the application. This is what is presented in section 2.2 and for which the results will be based on. Furthermore, it is important to remember that the types of errors vary widely between different antenna/sensor elements or types. The type of element also affects what types of subarray configurations are available or sensible. As such, there might be some mechanical tolerances that are very difficult to minimize and others that are naturally very small. As we shall see later there is a proof that reducing the number of different types of mechanical errors makes it easier to compensate for the mechanical errors. Another very important point is that arrays are in different levels of digitization. Some antennas employ full digital beamforming, while others have a completely analog weighting of the antenna elements. Then there are hybrid antennas where the subarrays are fully digital, while the elements within each subarray are analog. The type of antenna will limit to what extent the method in section 2.2 can be applied. Therefore, there will be an investigation into what the effectiveness for different types are. The template will be the same as the one presented in Table 3.1 and will investigate the following scenarios (illustrated in Figure 3.3):

- Analog beamforming
  - Only the complex excitation of each element is available. Therefore the Q-matrix of section 2.2 can only use the diagonal arguments. This will limit the possibilities of compensating to local calibration.
- Digital beamforming
  - The array response can be processed after receiving the signal while using the full Q-matrix for compensation. As such, both local and global calibration will be viable.
- Hybrid beamforming
  - A two step process can be employed where the elements within each subarray will be corrected in an analog fashion while the subarrays will be treated as an array of digital antennas. Local calibration and a hybrid global calibration is available

In order to investigate these scenarios the limits of each type of calibration must be found. Therefore, the analysis will be based on how large mechanical errors can occur before the SLL is degraded more than 2 dB from the 40 dB taper anywhere in uv-space. This safety margin is set due to the variations in the calibration results that occur when calculating the least square of the optimization problem and therefore the goal SLL is effectively -38 dB. Additionally, this criteria will be enforced within an viewing angle of 60 degrees around the scanning angle for global calibration (Equation 2.20). This corresponds to a solid angle of $\Omega = \pi sr$ and $\sqrt{(u-u_0)^2+(v-v_0)^2} \leq 0.866$. For local calibration (Equation 2.22) the goal is

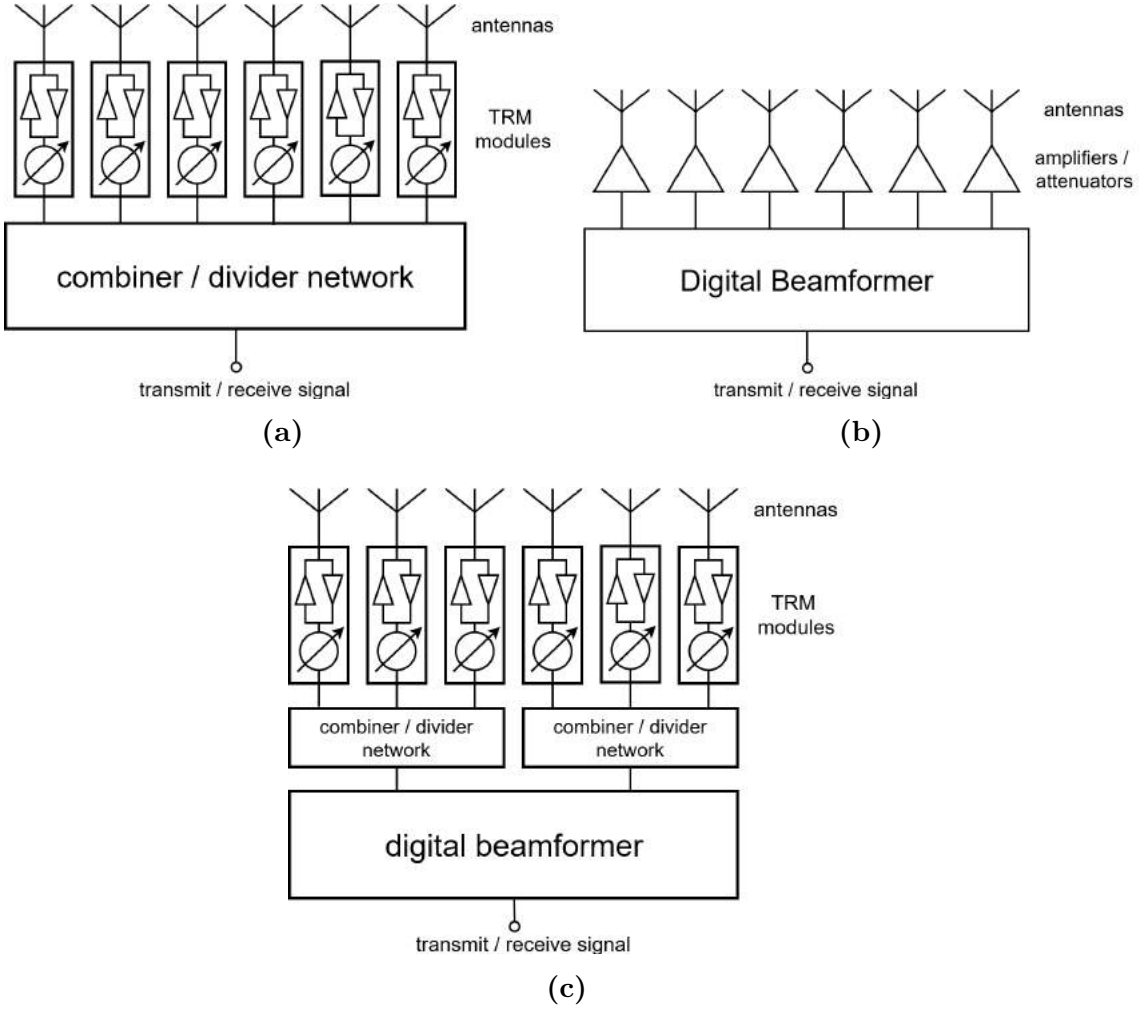


Figure 3.3: Three array configurations capable of using a) Analog beamfoming, b) Digital beamforming, c) Hybrid beamforming.

to calibrate the angles close to the scanning angle. As such the angle criteria here will be set to within 12 degrees of the scanning angle which correspond to a solid angle $\Omega = 0.137sr$ and $\sqrt{(u-u_0)^2 + (v-v_0)^2} \leq 0.208$. To simplify the process all mechanical translation errors must be whole percentages of the wavelength regardless of the combination of error types. When calibrating an array using a hybrid beamforming there can be several configurations between individual elements and subarrays. In other words, a suitable combination of global and local calibration must be chosen for a global beamforming goal. For a local beamforming goal it is clear that fully analog beamforming is sufficient as discussed in section 2.2. The possible combinations are

- Global subarray calibration + Global inter subarray calibration
- Global subarray calibration + Local inter subarray calibration
- Local subarray calibration + Global inter subarray calibration
- Local subarray calibration + Local inter subarray calibration

Notice that the global beamforming in the subarrays will only have the diagonal arguments of the correction matrix available and as such there is reason to believe that local calibration for the subarrays might be superior when trying to achieve global calibration for the entire antenna. Finally, the robustness of the method when scanning the beam without changing the calibration matrix will be investigated. The goal is to find the maximum deviation from the scanning angle from which the correction matrix is calculated that will incur a degradation in SLL lower than 1, 2, and 3 dB respectively. From the results the number of correction matrices required for a certain scanning space can be calculated.

3.3 Trial Antenna

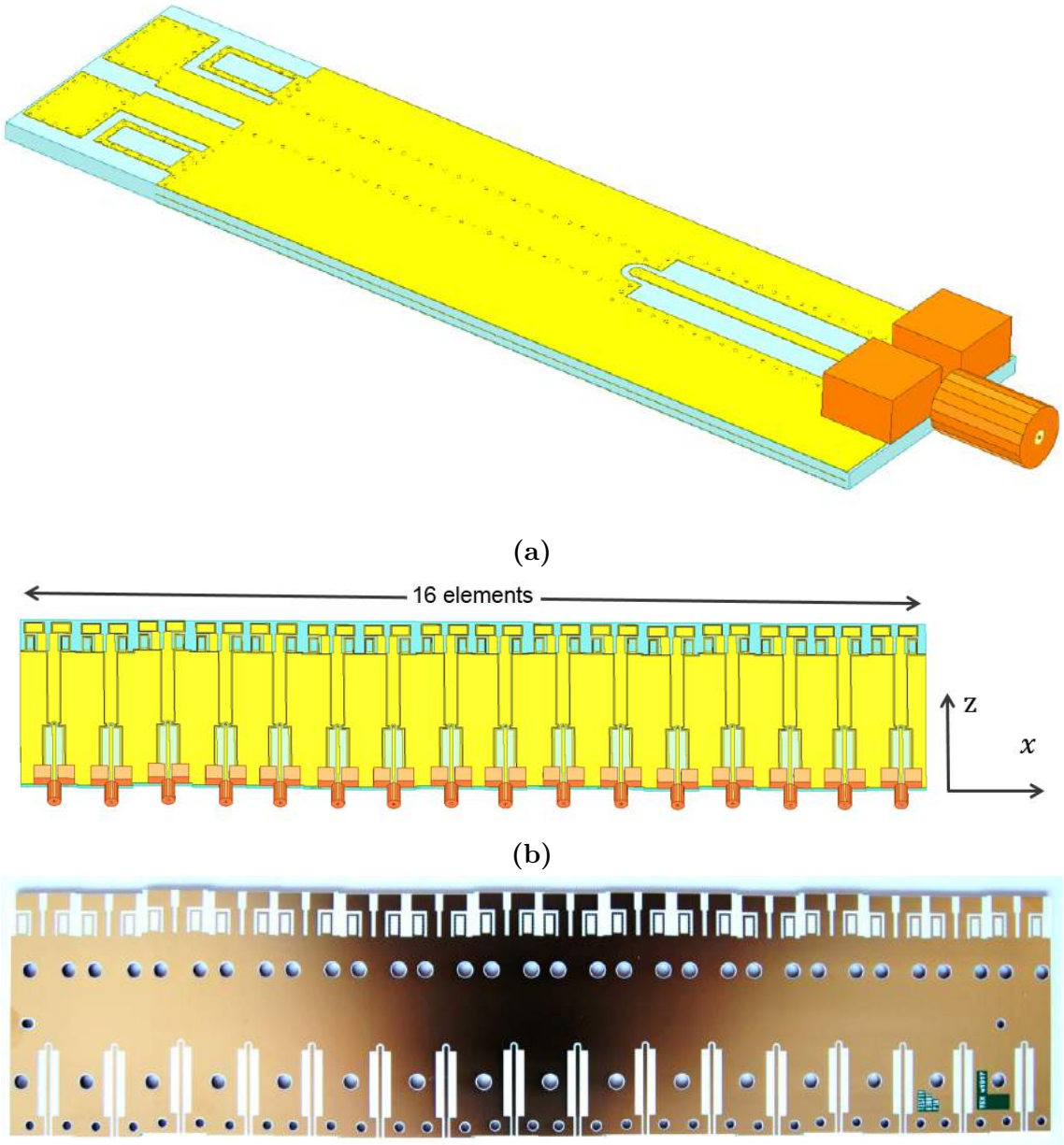
A trial antenna is constructed for the purpose of testing the methods investigated in prior sections of this report. A summary of the trial antenna parameters is seen in Table 3.2. The test antenna is a 16 element uniform linear array (ULA) and is shown in Figure 3.4. It is designed and simulated using the EM-simulation program Ansys HFSS and, using the design, later manufactured. This enables tests of the method on a smaller array, but the effects on larger antennas with subarrays will be left for further research. There are deliberate errors in the design of the trial antenna and these are based on one iteration of a ULA produced from MATLAB code based on the theory laid out in section 2.1. The positions of each element compared with the ideal position are presented in Table 3.3 and the element radiation pattern is shown in Figure 3.2b. The manufactured antennas' reflection coefficients are measured with a vector network analyzer (VNA) and the radiation pattern is measured in a compact antenna test range (CATR). The range used is the SAAB A15 CATR. See Appendix B for pictures of the measurement setup. To establish a most optimal calibration scenario as a reference, a calibration made based on the element radiation pattern from HFSS without any coupling between the elements is performed. The measured antenna is then calibrated and the results are compared with the optimal calibration results.

Table 3.2: Trial antenna configuration.

Array type	Uniform Linear Array (ULA)
Array size	16 elements
Element spacing	$\lambda/2$ (for 10 GHz)
Element radiation pattern	see Figure 3.2b
Weighting scheme	Taylor 30 dB
Frequency	9, 9.25, 9.5, 9.75, 10 GHz
Standard deviation	$(\sigma_x, \sigma_y, \sigma_z) = (0.01\lambda, 0.00\lambda, 0.01\lambda)$

Table 3.3: Trial antenna element positions in cartesian coordinates with ideal positions as reference. All units are in millimeters.

Index	Ideal X	Real X	Error X	Ideal Z	Real Z	Error Z
1	-112.42	-112.09	-0.33	0	0.43	0.43
2	-97.43	-97.10	-0.33	0	0.09	0.09
3	-82.44	-82.70	0.26	0	0.06	0.06
4	-67.45	-67.43	-0.02	0	0.48	0.48
5	-52.46	-52.83	0.37	0	-0.24	-0.24
6	-37.47	-37.81	0.34	0	0.21	0.21
7	-22.48	-22.49	0.01	0	0.25	0.25
8	-7.49	-7.04	-0.45	0	-0.07	-0.07
9	7.49	7.26	-0.23	0	0.06	0.06
10	22.48	22.60	-0.12	0	-0.35	-0.35
11	37.47	37.41	0.06	0	-0.34	-0.34
12	52.46	52.80	-0.34	0	0.03	0.03
13	67.45	67.13	0.32	0	0.22	0.22
14	82.44	82.45	-0.01	0	0.78	0.78
15	97.43	97.60	-0.17	0	-0.20	-0.20
16	112.42	112.75	-0.33	0	0.06	0.06



(a)

(b)

(c)

Figure 3.4: a) Trial antenna element design, b) Trial antenna array design, c) Manufactured trial antenna.

4
Results

4.1 Error Characterisation

Before presenting the results of the analysis a few parameters of the simulated array must be established. Firstly, we must make sure that there are no grating lobes. Using Equation 2.4 and setting the maximum scanning angle at 60 degrees we calculate the requirement $d \leq 0.64\lambda$ which is larger than half the wavelength which was the nominal element spacing. As such there is no need to modify the initially set element spacing. The HPBW of the simulated array is $\Delta u_{3dB} = 0.078$ based on the results of the MATLAB simulation and correspond to $\Delta\theta_{3dB} = 4.47°$. The directivity of the simulated array at boresight scanning is $D = 60.21$ dB and the maximum available directivity is $D_{max} = 64.13$ dB. This leaves a loss of 3.92 dB due to efficiencies. Since the grating lobe efficiency $e_{grt} = 1$ due to the nominal $\lambda/2$ element spacing and we assume a perfect polarisation efficiency, $e_{pol} = 1$, the only efficiency left is the illumination efficiency. This is not unity due to the applied 40 dB Taylor taper, and turns out to be $e_{ill} = 0.4053$. Moving on to the gain of the system we similarly assume that the mismatch factor and radiation efficiency is unity, since they bear little importance to the error characterisation analysis. As such the directivity and gain is equal in this theoretical case. The parameters are summarized in Table 4.1.

Table 4.1: Parameters of array based on array setup from Table 3.1

Max element spacing	$d_{max} = 0.64\lambda$
Half Power Beamwidth	$\Delta\theta_{3dB} = 4.47°$
Directivity	$D = 60.21$ dB
Maximum possible directivity	$D_{max} = 64.13$ dB
Illumination efficiency	$e_{ill} = 0.4053$

The optimal radiation pattern, subject to Table 3.1 and Table 4.1, can be seen in Figure 4.1 and 4.2. There are two principal cross sections with sidelobes centering at the scanning angle. These values are the most important for the characterisation.

4.1.1 Element Position Errors

Errors in x-position is characterised by an increasing variance with larger viewing angle |u| as can be seen in both Figure 4.3a and 4.4. Examining Figure 4.3b, 4.4, 4.5

and 4.6, we can conclude that the standard deviation does not change with scanning angle. Additionally, the effects are mitigated somewhat due to the element patterns effects at larger viewing angles. This type of error would be a bigger problem with more isotropic antenna elements [1], but due to the more realistic element pattern, the larger viewing angles have slightly lower amplitude than smaller viewing angles. Y-position element errors exhibits the same errors as x-position errors, but with an inversion of the u and v coordinates. Consequently, the errors increase with larger |v| angle instead, since the radiation pattern in H-plane is more isotropic.

Figure 4.7, 4.8 and 4.9 illustrates that the effects of z errors are larger for small viewing angles and decreases with larger viewing angles. These errors are also symmetrical and will give the same results for both u and v axes. Due to the realistic element radiation pattern, the effect on SLL is significantly larger than for x,y-position errors.

Combining all element errors means that all scanning angles exhibit variance close to the variance of the type of element error that dominates that viewing angle. For angles close to the main lobe, the variance is similar to the the variance seen for z-position element errors. For large viewing angles the variance is more like for x,y-position errors. In conclusion, the SLL does not increase noticably when all element errors are combined. The results for combined element errors can be seen in Figure 4.10, 4.11, and 4.12.

4.1.2 Subarray Position Errors

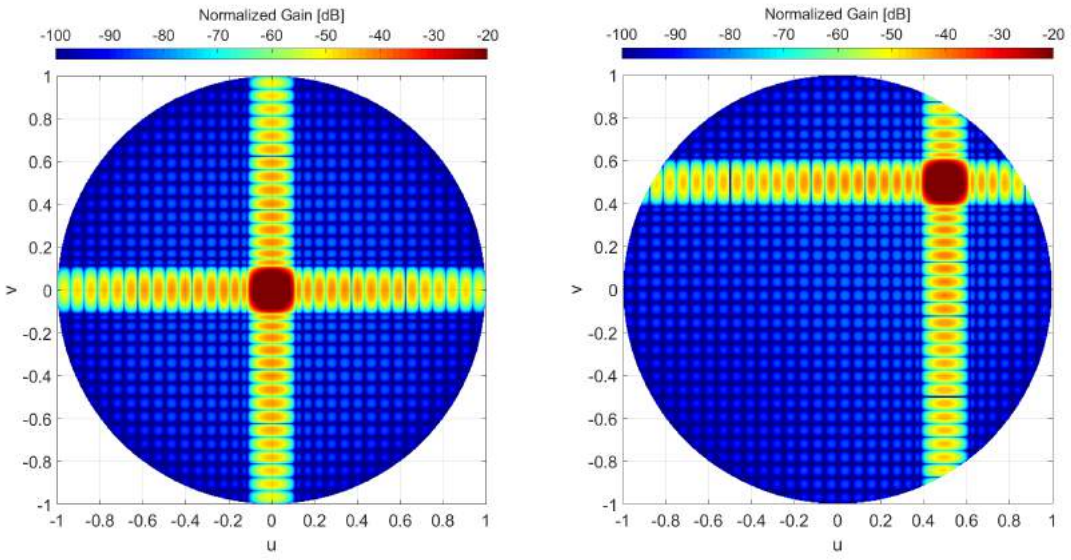
Figure 4.13, 4.14, 4.15 and 4.16 shows that subarray errors introduce a periodicity to the standard deviation of the radiation pattern. The nature of this periodicity depends on the configuration of the subarray, or in other words, follow the radiation pattern of the subarray. When looking at the boresight scanning in Figure 4.13a, the peaks of the periodicity display the same amplitude as for an x-position element error. However, if the scanning angle |u| is larger, as in Figure 4.13b, the peaks are larger than for element errors, which is due to grating lobes created by the subarray errors. Y-position subarray errors will exhibit the same errors as x-position errors, but with an inversion of the u and v coordinates. This will not be true if the array or the element beam pattern are not symmetric.

The z value error results can be seen in Figure 4.17, 4.18, and 4.19. The z-position subarray errors results in a similar periodicity to x,y-position subarray errors. Furthermore, the same effect as for z-positon element errors where the effect decreases with larger viewing angles is also seen here. This type of error has the largest effect on the SLL (except for very large scanning angles) with large side lobes close to the main lobe.

Similar to the combined element errors, the combined subarray errors creates variance with the largest effects dominating for each viewing angle. The periodicity stays the same, but the peaks are the maximum from the individual subarray x,y,z error results as can be seen in Figure 4.20, 4.21, and 4.22.

4.1.3 Combined Element and Subarray Errors

Including all types of errors does not make a significant difference in SLL compared to the combined subarray errors. However, the variance of the lower points of the periodic pattern are increased somewhat, since the effects of element errors are more visible at these directions.



(a) **(b)**

Figure 4.1: Optimal radiation pattern for the array setup described in Table 3.1 with (a) Boresight scanning (b) $u_0 = 0.5, v_0 = 0.5$

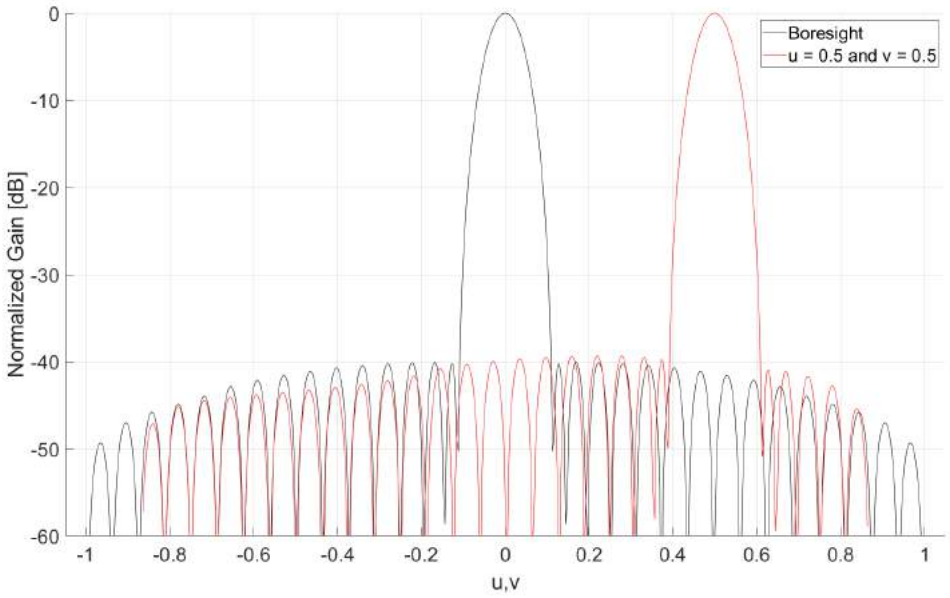
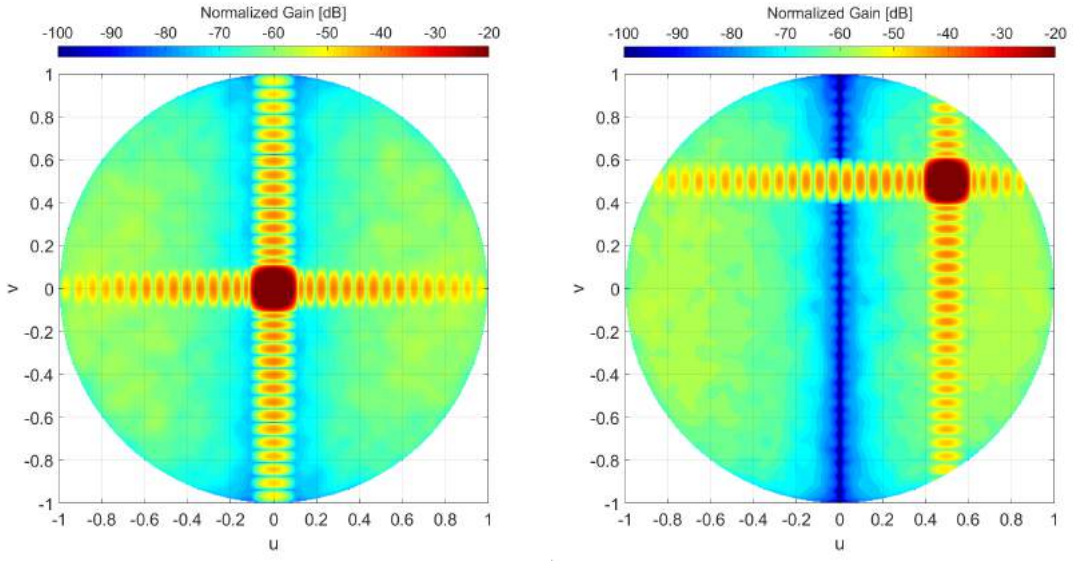


Figure 4.2: Cross-sections through main lobe for optimal radiation pattern for array setup in Table 3.1. Both boresight and the scanning angle have the same pattern along u and v axes.



(a) **(b)**

Figure 4.3: Average radiation pattern for element x-position errors with $\sigma = 0.01\lambda$ for (a) Boresight scanning (b) $u_0 = 0.5, v_0 = 0.5$

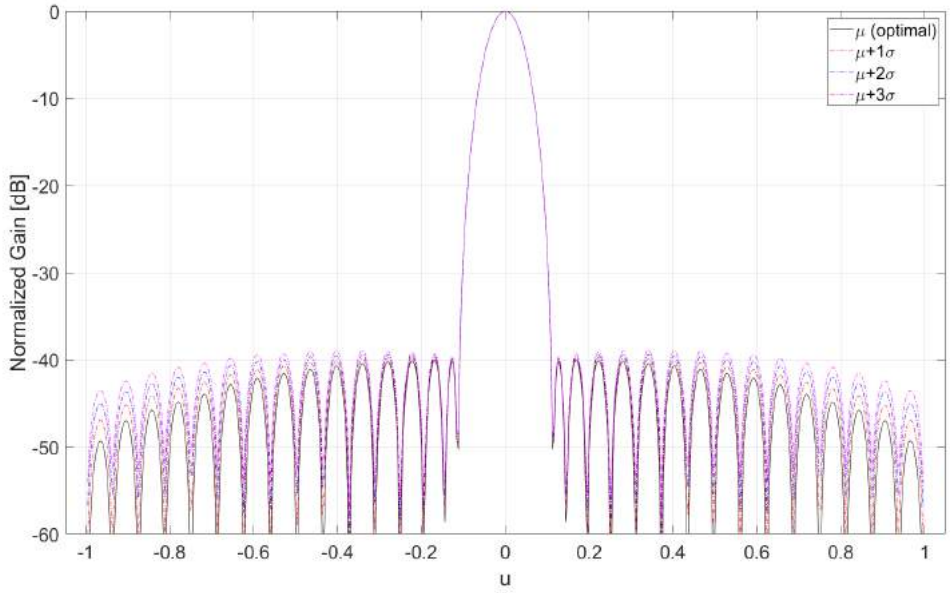


Figure 4.4: Radiation pattern along u-axis through main lobe for element x-position errors with $\sigma = 0.01\lambda$ and boresight scanning. First, second and third standard deviation above optimal pattern shown.

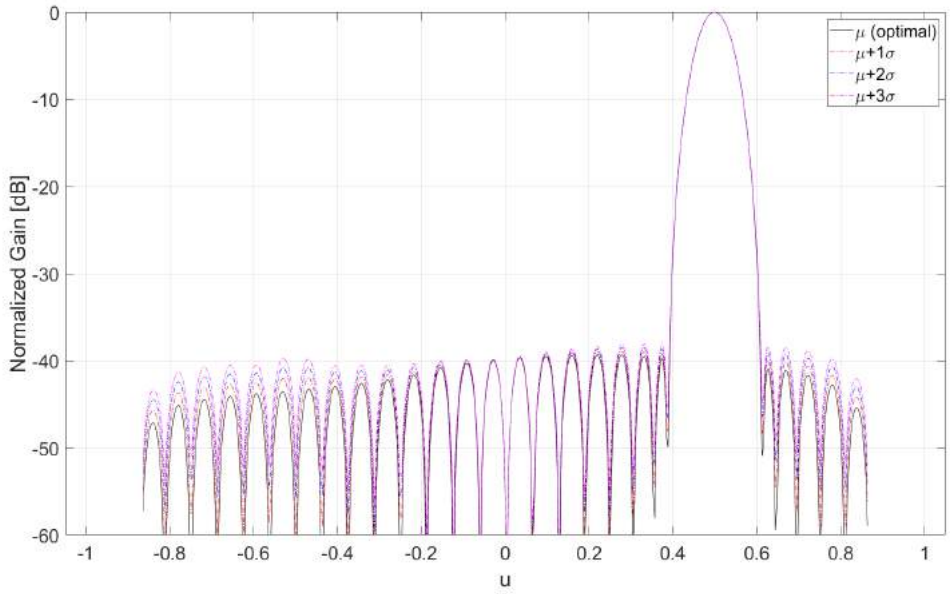


Figure 4.5: Radiation pattern along u-axis through main lobe for element x-position errors with $\sigma = 0.01\lambda$ and scanning angle $u_0 = 0.5, v_0 = 0.5$. First, second and third standard deviation above optimal pattern shown.

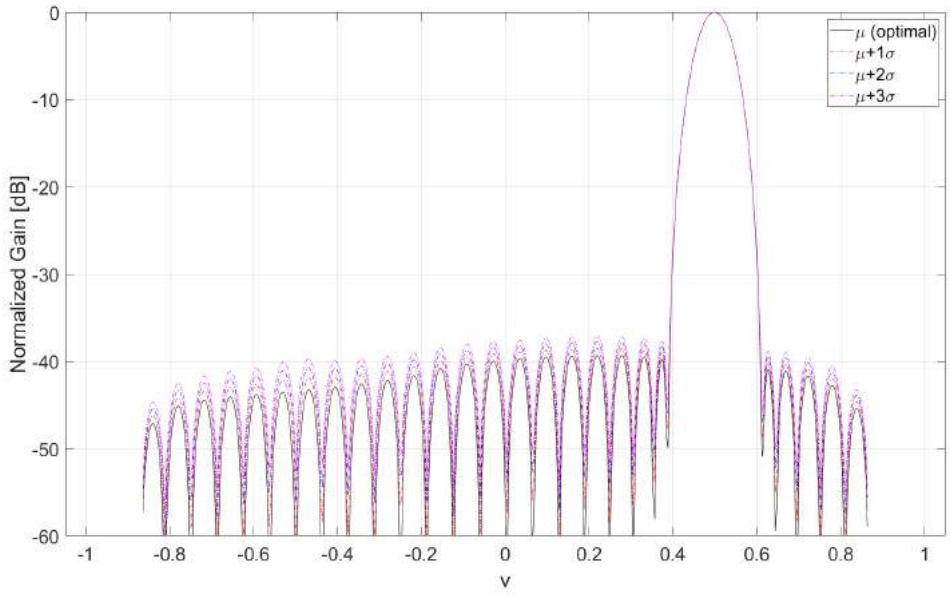
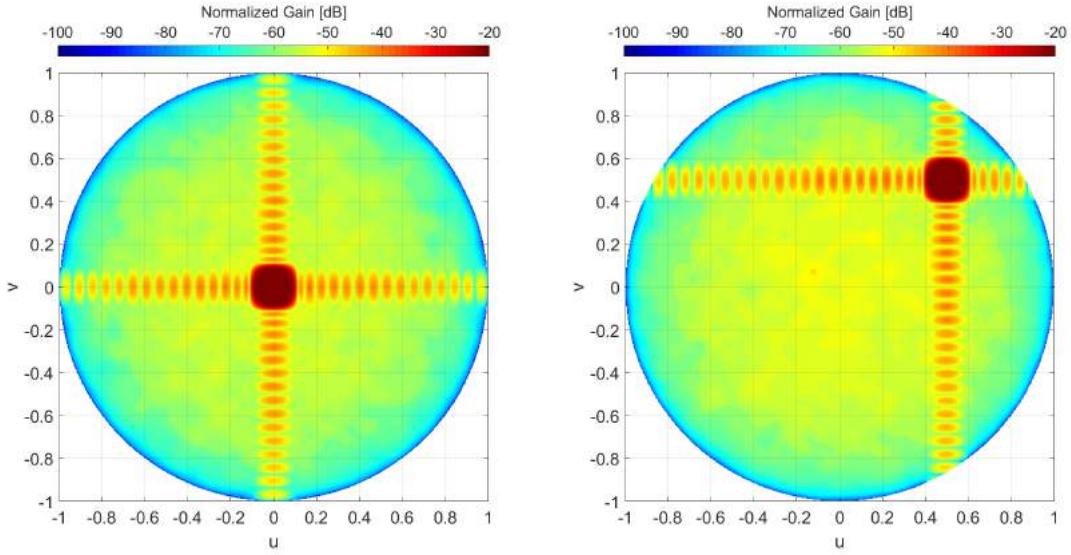


Figure 4.6: Radiation pattern along v-axis through main lobe for element x-position errors with $\sigma = 0.01\lambda$ and scanning angle $u_0 = 0.5, v_0 = 0.5$. First, second and third standard deviation above optimal pattern shown.



(a) **(b)**

Figure 4.7: Average radiation pattern for element z-position errors with $\sigma = 0.01\lambda$ and (a) Boresight scanning (b) $u_0 = 0.5, v_0 = 0.5$

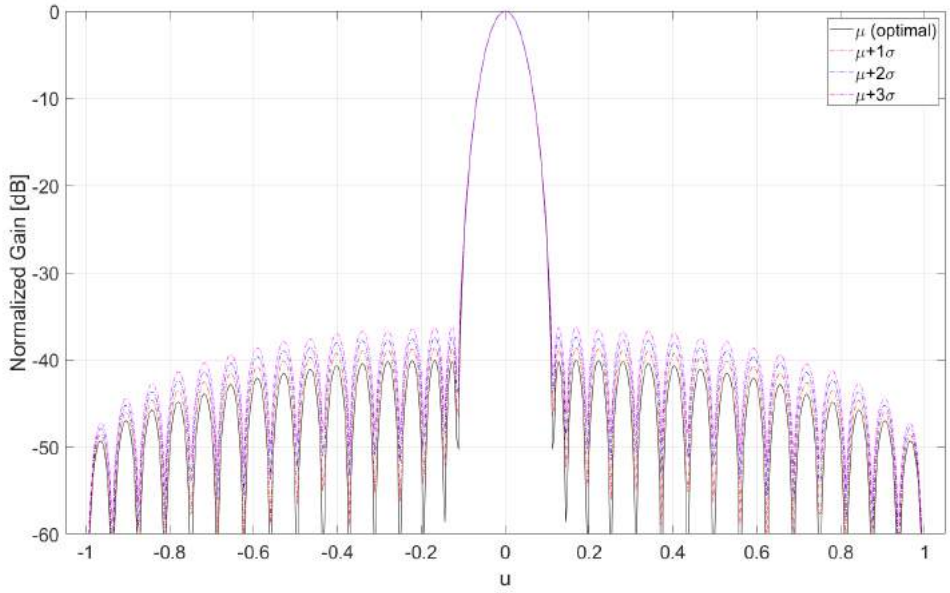


Figure 4.8: Radiation pattern along u-axis through main lobe for element z-position errors with $\sigma = 0.01\lambda$ with boresight scanning. First, second and third standard deviation above optimal pattern shown.

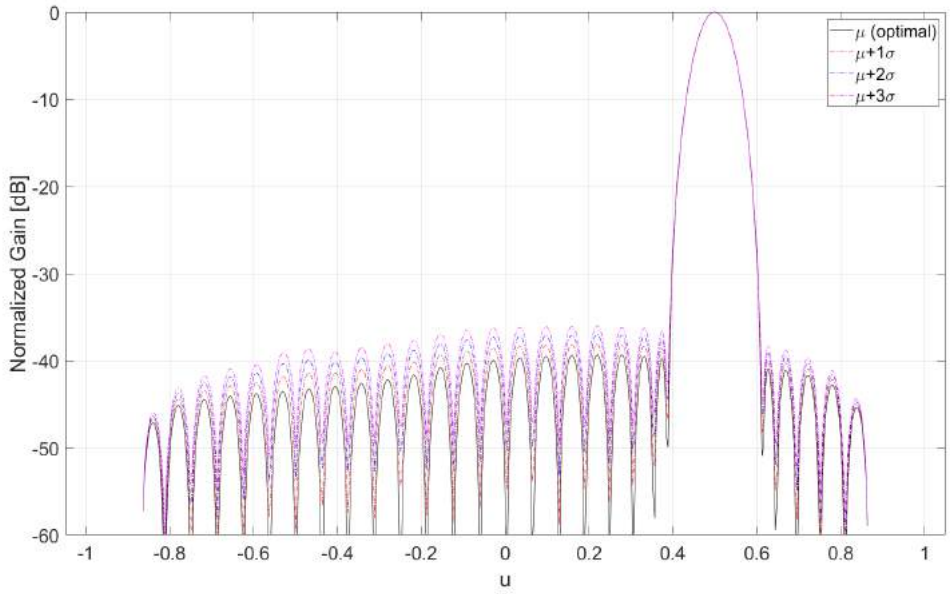
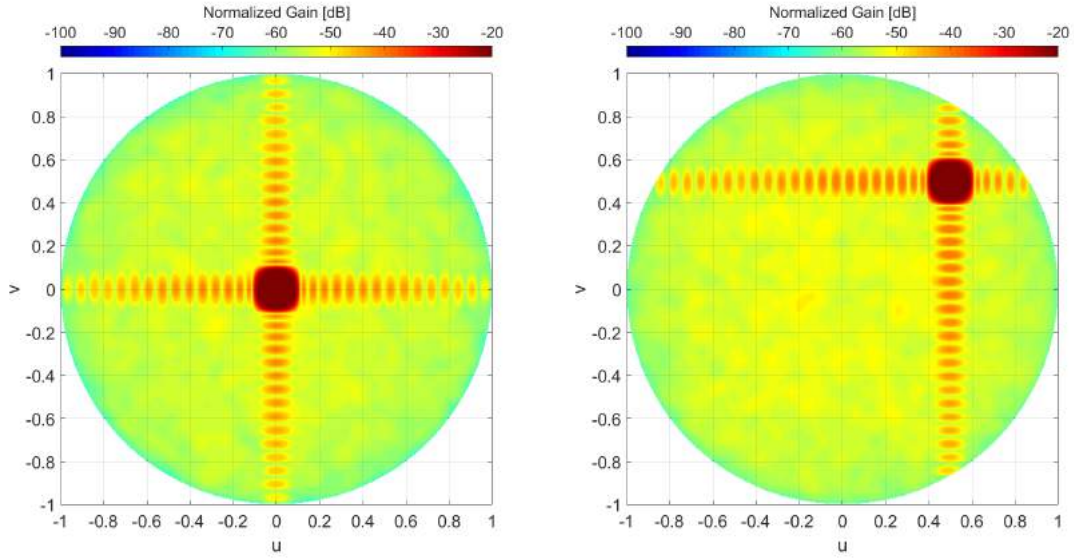


Figure 4.9: Radiation pattern along u-axis through main lobe for element z-position errors with $\sigma = 0.01\lambda$ with scanning angle $u_0 = 0.5, v_0 = 0.5$. First, second and third standard deviation above optimal pattern shown.



(a) **(b)**

Figure 4.10: Average radiation pattern for element x,y,z-position errors with $\sigma = 0.01\lambda$ with (a) Boresight scanning (b) $u_0 = 0.5, v_0 = 0.5$

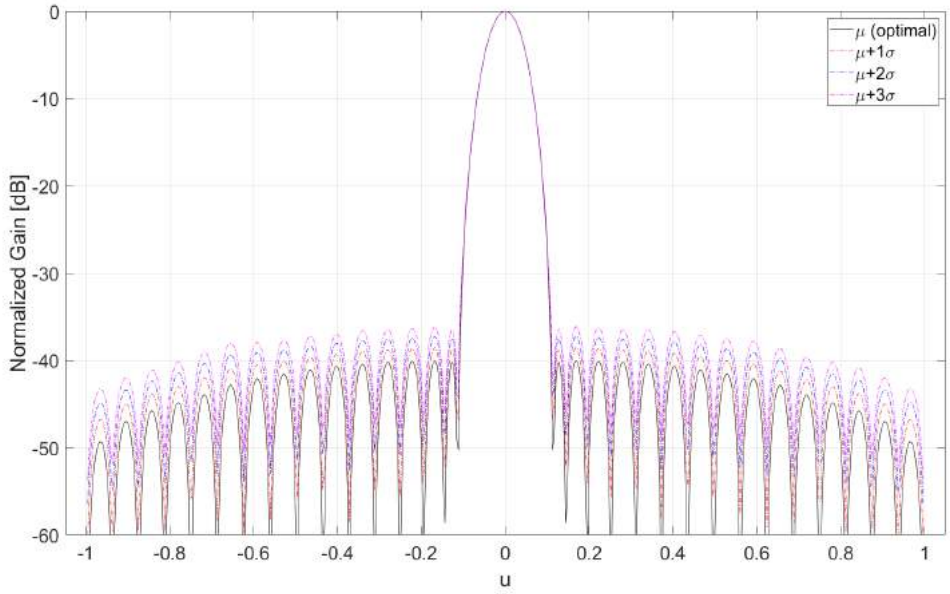


Figure 4.11: Radiation pattern along u-axis through main lobe for element x,y,z-position errors with $\sigma = 0.01\lambda$ with boresight scanning. First, second and third standard deviation above optimal pattern shown.

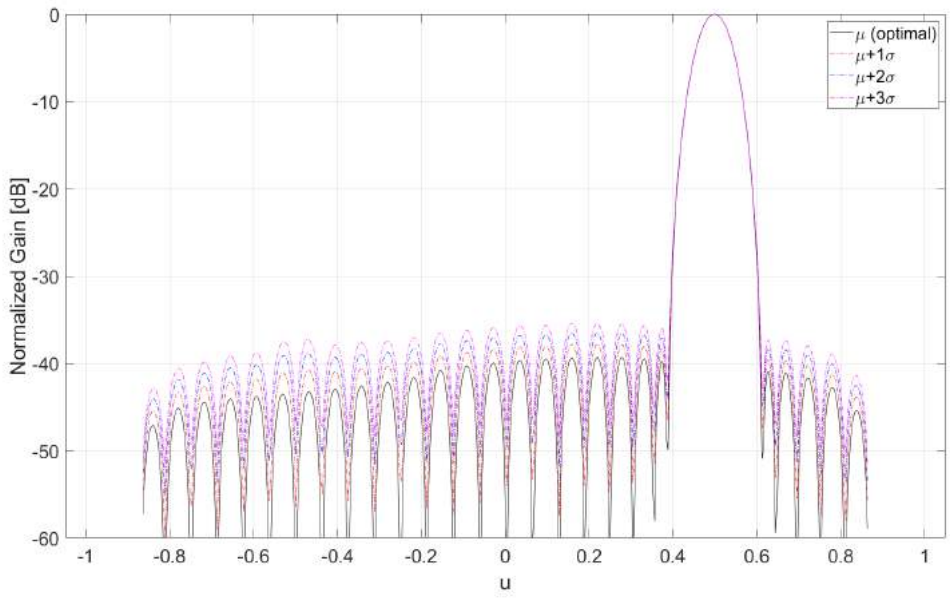
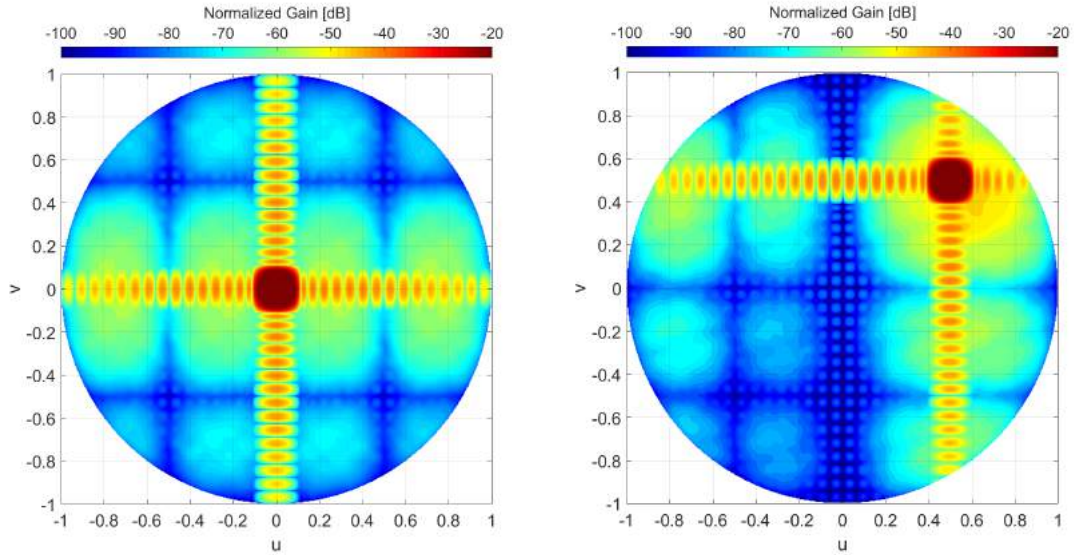


Figure 4.12: Radiation pattern along u-axis through main lobe for element x,y,z-position errors with $\sigma = 0.01\lambda$ with $u_0 = 0.5, v_0 = 0.5$ scanning. First, second and third standard deviation above optimal pattern shown.



(a) **(b)**

Figure 4.13: Average radiation pattern for subarray x-position errors with $\sigma = 0.01\lambda$ with (a) Boresight scanning (b) $u_0 = 0.5, v_0 = 0.5$.

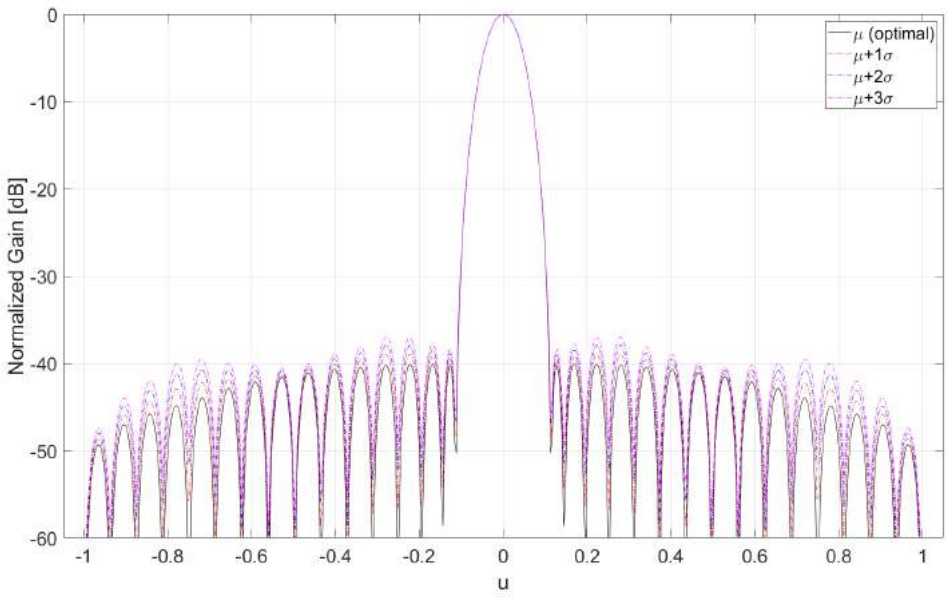


Figure 4.14: Radiation pattern along u-axis through main lobe for subarray x-position errors with $\sigma = 0.01\lambda$ with boresight scanning. First, second and third standard deviation above optimal pattern shown.

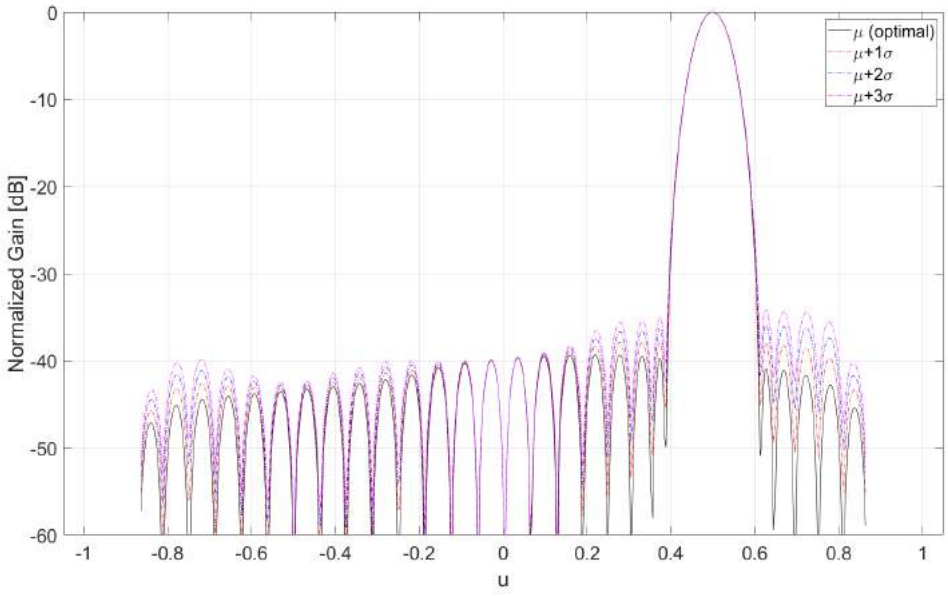


Figure 4.15: Radiation pattern along u-axis through main lobe for subarray x-position errors with $\sigma = 0.01\lambda$ with $u_0 = 0.5, v_0 = 0.5$ scanning. First, second and third standard deviation above optimal pattern shown.

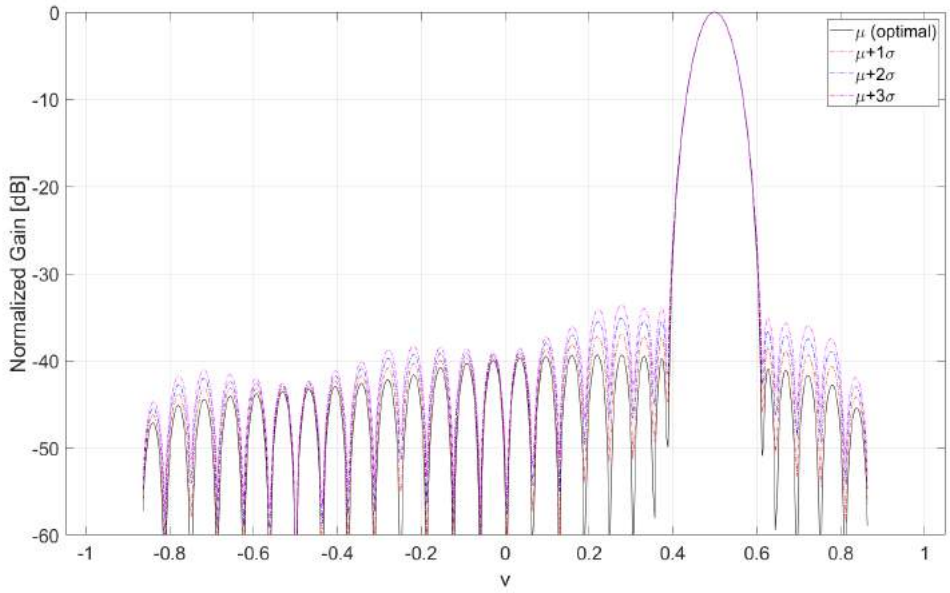
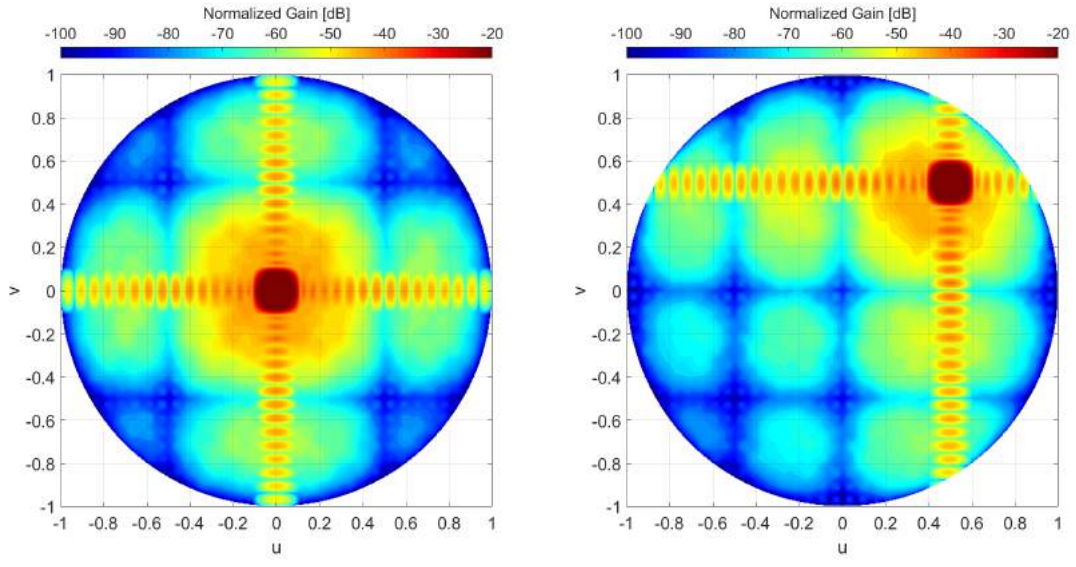


Figure 4.16: Radiation pattern along v-axis through main lobe for subarray x-position errors with $\sigma = 0.01\lambda$ with $u_0 = 0.5, v_0 = 0.5$ scanning. First, second and third standard deviation above optimal pattern shown.



(a) **(b)**

Figure 4.17: Average radiation pattern for subarray z-position errors with $\sigma = 0.01\lambda$ with (a) Boresight scanning (b) $u_0 = 0.5, v_0 = 0.5$

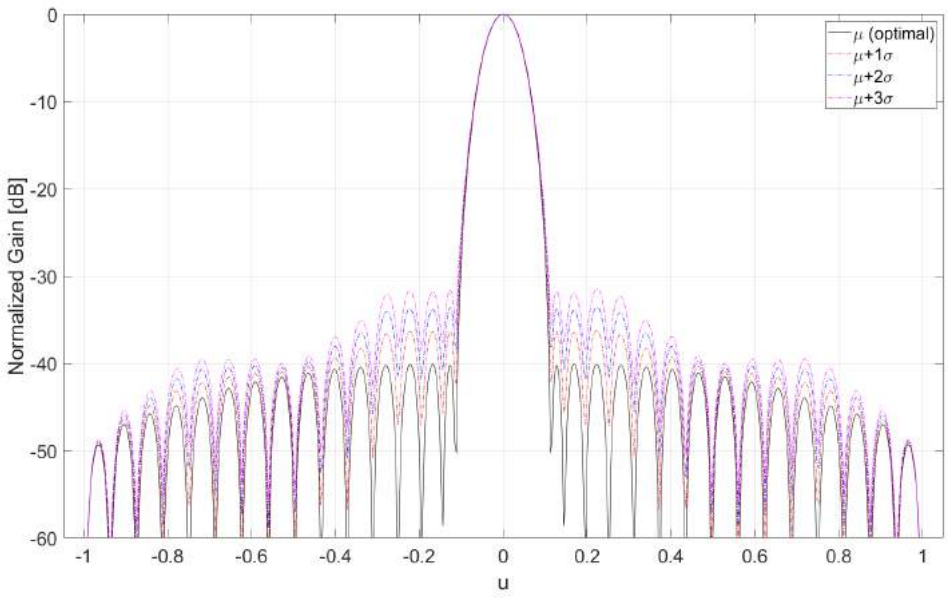


Figure 4.18: Radiation pattern along u-axis through main lobe for subarray z-position errors with $\sigma = 0.01\lambda$ with boresight scanning. First, second and third standard deviation above optimal pattern shown.

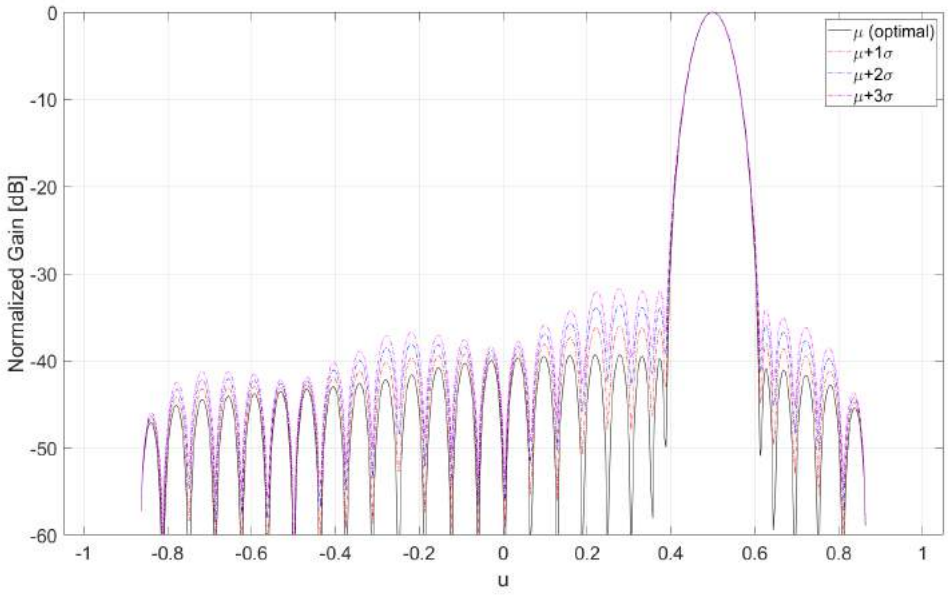
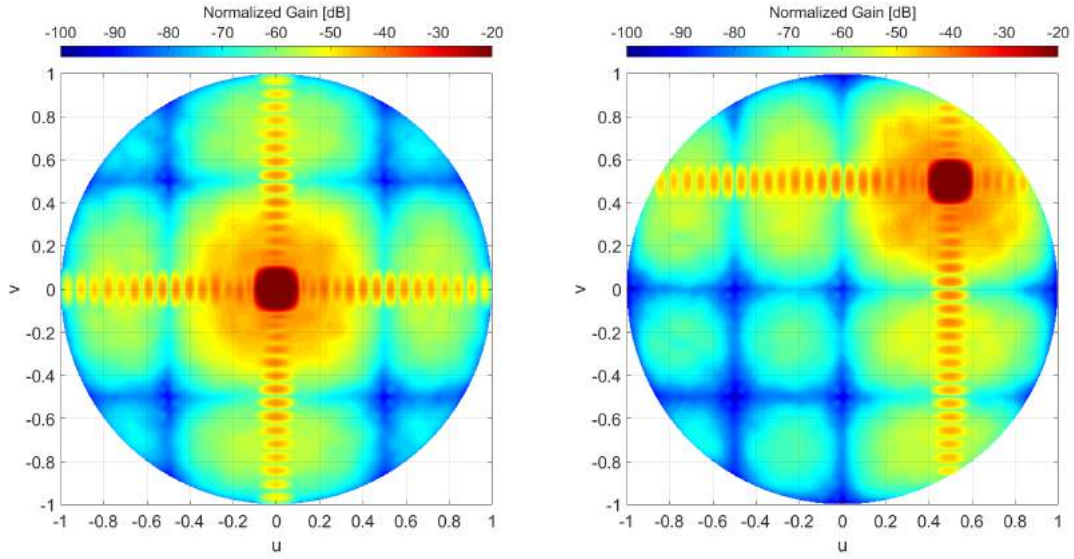


Figure 4.19: Radiation pattern along u-axis through main lobe for subarray z-position errors with $\sigma = 0.01\lambda$ with $u_0 = 0.5, v_0 = 0.5$ scanning. First, second and third standard deviation above optimal pattern shown.



(a) **(b)**

Figure 4.20: Average radiation pattern for subarray x,y,z-position errors with $\sigma = 0.01\lambda$ with (a) Boresight scanning (b) $u_0 = 0.5, v_0 = 0.5$

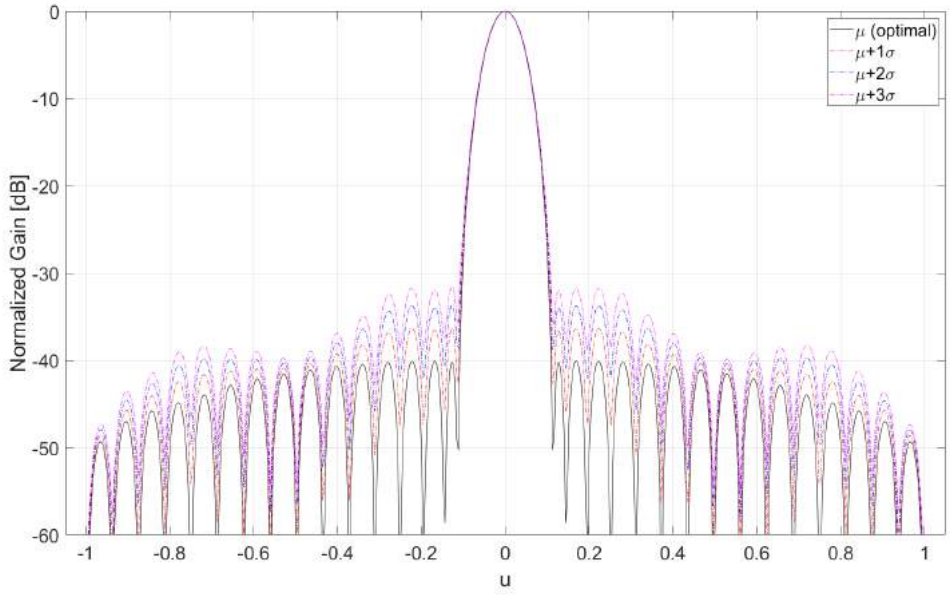


Figure 4.21: Radiation pattern along u-axis through main lobe for subarray x,y,z-position errors with $\sigma = 0.01\lambda$ with boresight scanning. First, second and third standard deviation above optimal pattern shown.

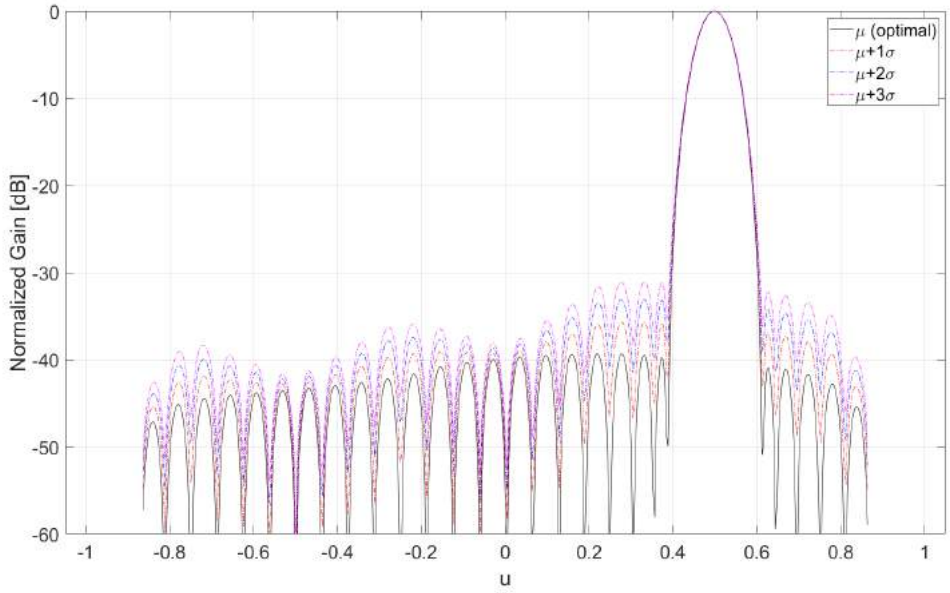
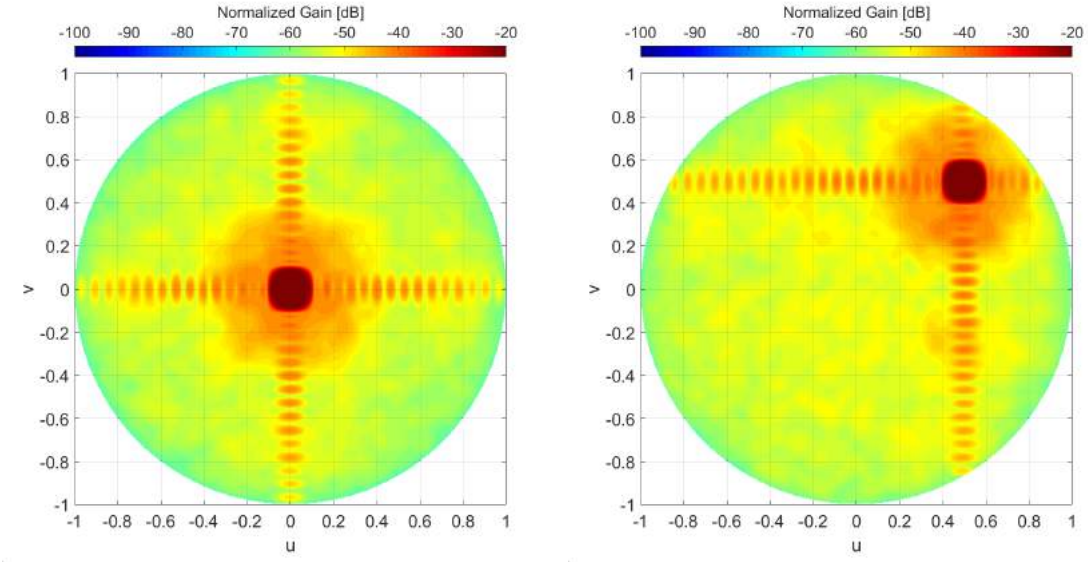


Figure 4.22: Radiation pattern along u-axis through main lobe for subarray x,y,z-position errors with $\sigma = 0.01\lambda$ with $u_0 = 0.5, v_0 = 0.5$ scanning. First, second and third standard deviation above optimal pattern shown.



(a) **(b)**

Figure 4.23: Average radiation pattern for element and subarray x,y,z-position errors with $\sigma = 0.01\lambda$ with (a) Boresight scanning (b) $u_0 = 0.5, v_0 = 0.5$

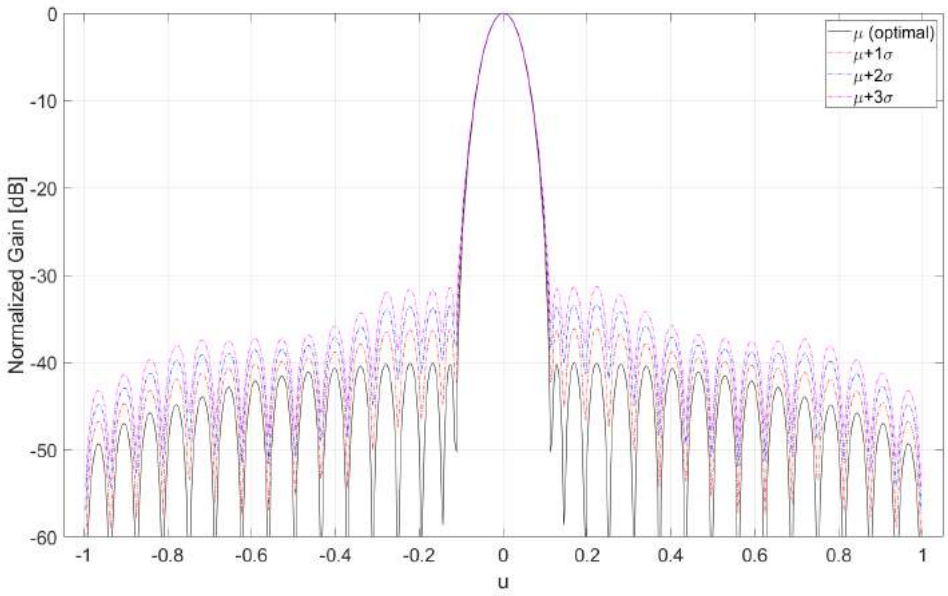


Figure 4.24: Radiation pattern along u-axis through main lobe for element and subarray x,y,z-position errors with $\sigma = 0.01\lambda$ with boresight scanning. First, second and third standard deviation above optimal pattern shown.

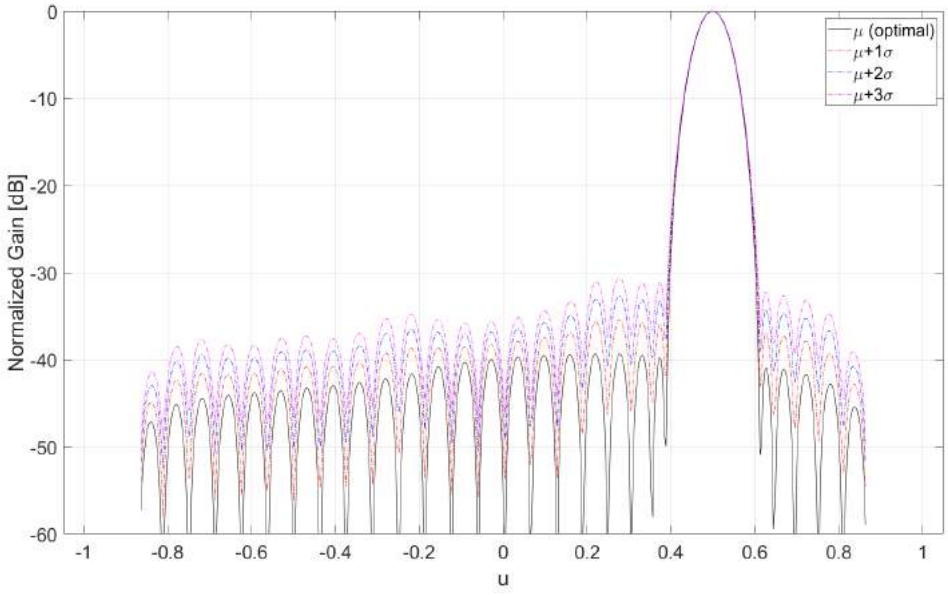


Figure 4.25: Radiation pattern along u-axis through main lobe for element and subarray x,y,z-position errors with $\sigma = 0.01\lambda$ with $u_0 = 0.5, v_0 = 0.5$ scanning. First, second and third standard deviation above optimal pattern shown.

4.1.4 Monte Carlo Tolerance Analysis

Investigating the gain distribution of 1000 iterations of element errors reveals the behaviour shown in Figure 4.26, which displays results from the highest sidelobe (ideal SLL=-40 dB) with boresight scanning at viewing angle u=0.168, v=0. It shows a histogram of the data together with the probability distributions of Equation 2.11, 2.14, and 2.15 generated based on fitting the equations with the data samples. Here we can see a confirmation that the radiation patterns gain does indeed follow a Ricean distribution that approximates a Normal distribution when the errors are small and a Rayleigh distribution when the errors are large. In this case, the standard deviations of the translation errors are 0.005λ, 0.01λ, and 0.02λ for all investigated error types. $\sigma = 0.01\lambda$ is the error investigated in this section and for this case we start to see a clear distinction from a normal distribution. As such, the distribution cannot be assumed gaussian above an average mechanical standard deviation of one percent of lambda for the current setup. Additionally, the assumption made that the radiation pattern errors are normally distributed when calculating standard deviations is not completely correct, when dealing with all types of errors simultaneously. However, the assumption should be operable for fewer types of errors.

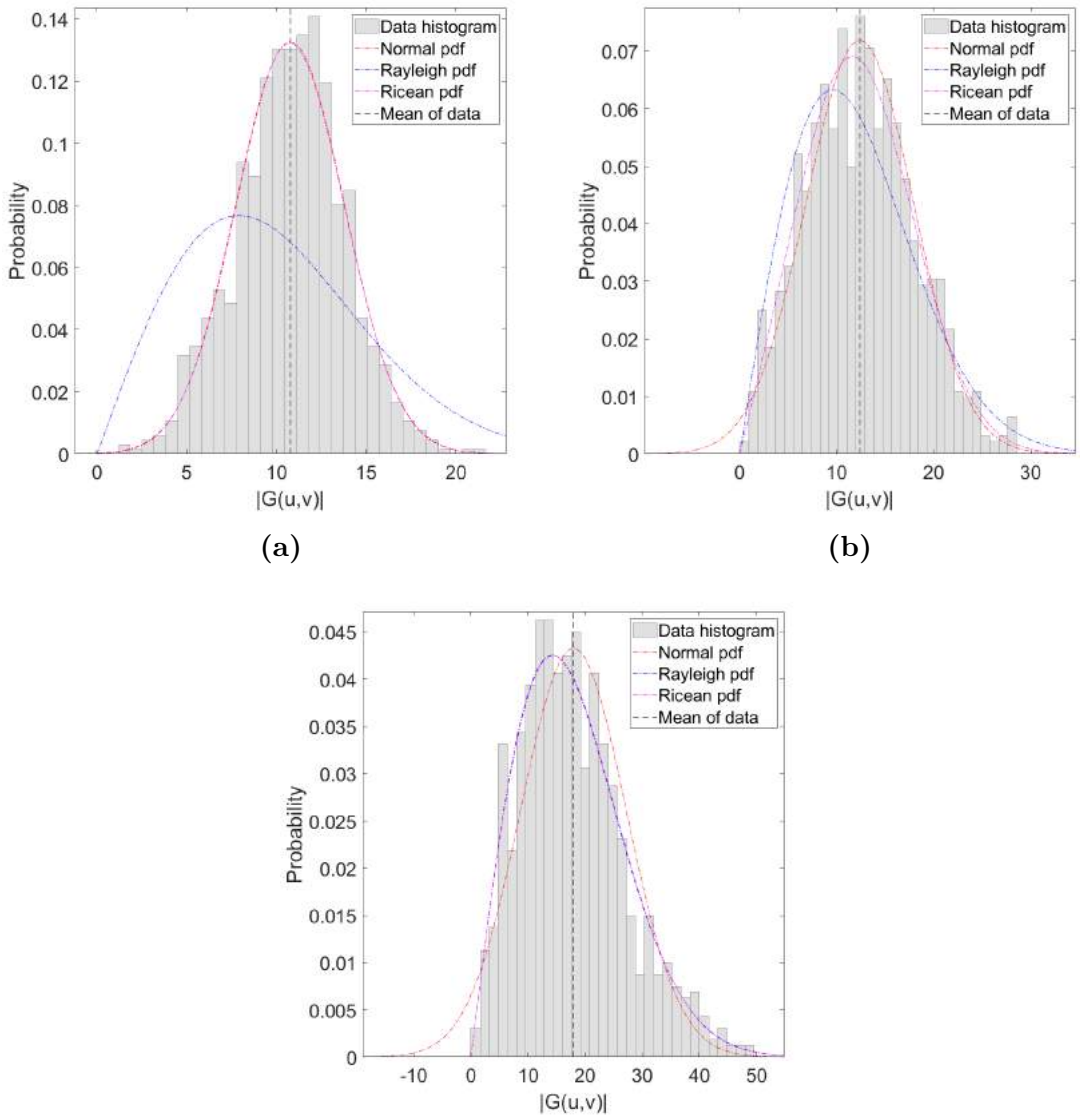


Figure 4.26: PDFs and histograms of gain for 1000 iterations with a) $\sigma = 0.005\lambda$ for all element and subarray errors, b) $\sigma = 0.01\lambda$ for all element and subarray errors, c) $\sigma = 0.02\lambda$ for all element and subarray errors

4.2 Error Compensation

It would be useful to analyse the results of calibration in a similar manner as in section 4.1. Unfortunately, due to time constraints the result will be presented through specific mechanical error instances. As such, these results are simply a rule of thumb for calibrating an array with different types of mechanical errors. A table of the maximum mechanical error possible for each calibration method and a combination of translation errors is presented in Table 4.2.

Table 4.2: Maximum standard deviation before a maximum degradation of 2 dB for each calibration type and error combination. Both element and subarray errors have the same standard deviation on their mechanical tolerances in each scenario.

Calibration type	**x**	**z**	**xz**	**xyz**
Local Calibration	0.04λ	0.19λ	0.05λ	0.03λ
Global Calibration	0.09λ	0.12λ	0.05λ	0.04λ
Hybrid Calibration	0.02λ	0.07λ	0.02λ	0.02λ

4.2.1 Local Calibration

Immediately with the first results we can see that local calibration is essentially unable to compensate for errors in the xy-plane for boresight scanning, as can be seen in Figure 4.27. Therefore, it would be optimal to reduce mechanical errors in these dimensions when employing local calibration. However, it is worth to keep in mind that the errors close to the main beam are small for boresight scanning when only errors in the xy-plane are present. When scanning away from boresight, the calibration capabillities of local calibration is increasing with a larger scanning angle. Looking at Figure 4.28 we can see that in contrast to x- or y-position errors the compensation capacity when dealing with z-position errors is very good. The degradation in the local calibration area does not exceed 2 dB until the mechanical tolerances of both the element and subarray positions exceed 19 percent of the wavelength. We can also conclude that adding more dimensions to the translational errors reduces the maximum standard deviation that can be calibrated as proven by Figure 4.29 and 4.30.

4.2.2 Global Calibration

The global calibration available when a fully digitalized array is used is significantly more powerful when the goal is to compensate for mechanical errors in a large portion of uv-space. Most significantly we can see that calibrating xy-plane errors is possible and yields much better results than local calibration, as shown in Figure 4.31, 4.33, and 4.34. Calibrating the z-position errors however is slightly more difficult, as seen in Figure 4.32. Additionally, the previously observed difficulty with calibrating many dimensions of error at the same time is also present when using global calibration in

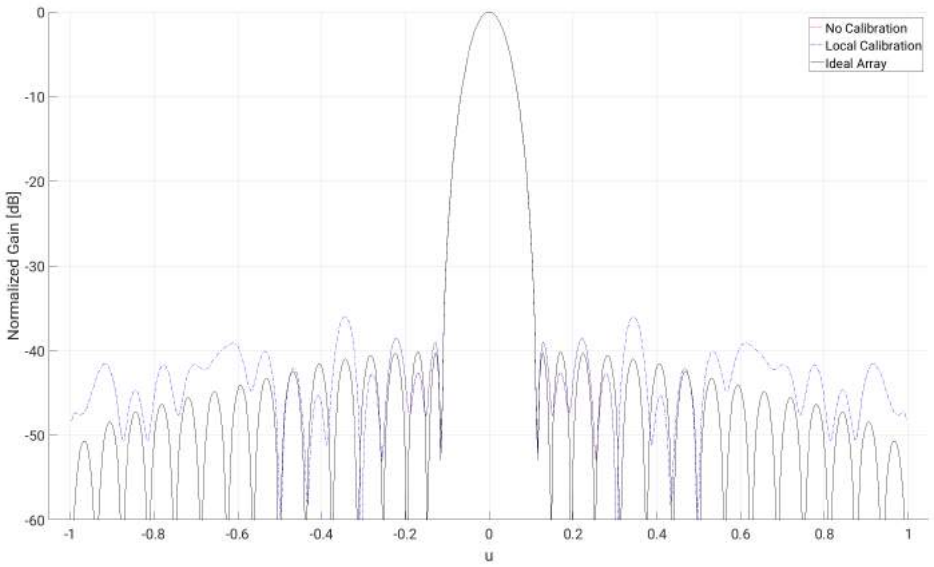


Figure 4.27: Limit of local calibration before a maximum 2 dB degradation in SLL for x-position element and subarray errors at $\sigma = 0.04\lambda$

a fully digital array. Worth noting is that the global calibration produces very large sidelobe for relatively low standard deviations on the mechanical errors at viewing angles wher $\theta \leq 60°$, which is visible in all of Figures 4.31 to 4.34. If the goal is to eliminate sidelobes completely from the half space the calibration will be less effective. Additionally, these sidelobes persist when scanning away from boresight.

4.2.3 Hybrid Calibration

It is clear from simulation results that the best calibration combination when aiming for global calibration is to use global calibration for both within subarrays and between subarrays. When calculating the correction matrix for the subarrays only the diagonal arguments can be used, but the compensation achieved is such that the radiation pattern of each subarray is symmetric around $\theta = 0$ if not optimally close to the ideal radiation pattern. Thist way the global calibration between the subarrays uses 'elements' that have roughly the same radiation pattern as the ideal subarrays. As expected the compensation capacity of the hybrid approach is noticibly less than the fully digital beamforming approach when trying to achieve a global calibration. The local calibration capacity is of course the same as for all other configurations where each elemnent can be individually excited by an amplitude and phase. Looking at Figure 4.35 we can see a similar difficulty as for the local calibration when trying to compensate for errors in the xy-plane. Additionally the same improved results with z-position errors is seen in Figure 4.36. When we look at combinations of xz-errors and xyz-errors as seen in Figure 4.37 and 4.38 the conclusion is that the errors in the xy-plane is the limiting factor since the z-errors are almost completely compensated for.

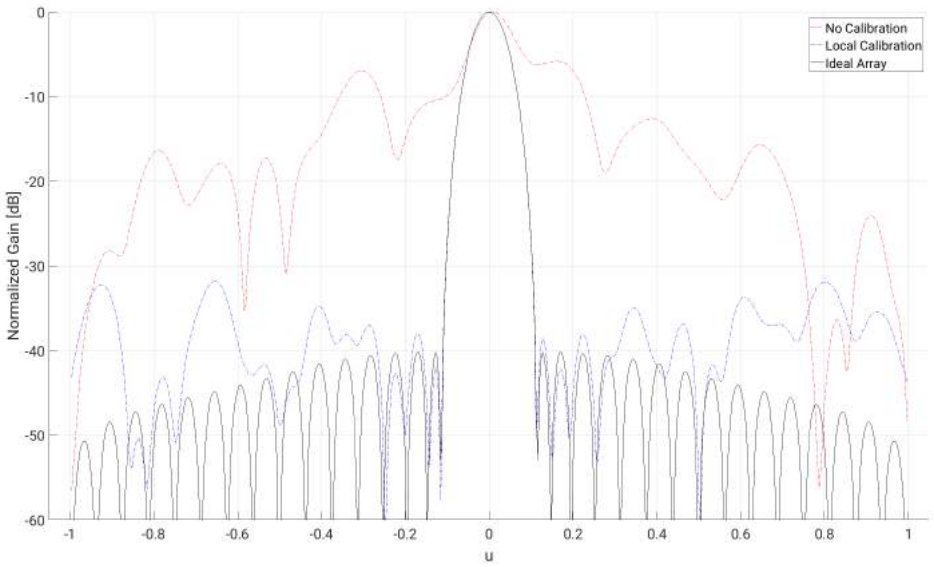


Figure 4.28: Limit of local calibration before a maximum 2 dB degradation in SLL for z-position element and subarray errors at $\sigma = 0.19\lambda$

4.2.4 Correction Matrix Robustness

Once a correction matrix has been calculated it is important to know to what extent the scanning angle can be changed without needing to calculate a new correction matrix, as discussed in section 3.2. An investigation on what level of degradation occurs based on the number of degrees scanned away from boresight when all translational errors ar set to one percent of the wavelength and is presented in Table 4.3. If we assume that the array will not be scanned beyond 60 degrees another table of the number of correction matrices required for maintaining a maximum degradation lower than 1, 2, and 3 dB can be made and is shown in Table 4.4. This is based on how many hexagons are needed to cover the square surrounding the circle in uv-space where $\theta = 60°$. See Figure 4.39. Each hexagon has a distance from the centre to the corners the same as the maximum scanning angles from Table 4.3 in uv coordinates. The hexagons can then simply be replaced by circles with the afformentioned radius to cover the area in an efficient manner. As such this will constitute an upper bound on the number of required Q-matrices. It can be seen that the global calibration offer better compensation of mechanical errors, but the local calibration is far more robust when it comes to how densely the correction matrices must be applied. However, this comparison is somewhat unfair, since the goals for how large space the two types try to compensate for are different. The hybrid calibration seems to be the most robust and this might be because the array is treated as a smaller array of subarrays which would lend some more robustness to the calibration since correction matrices for smaller arrays are more robust.

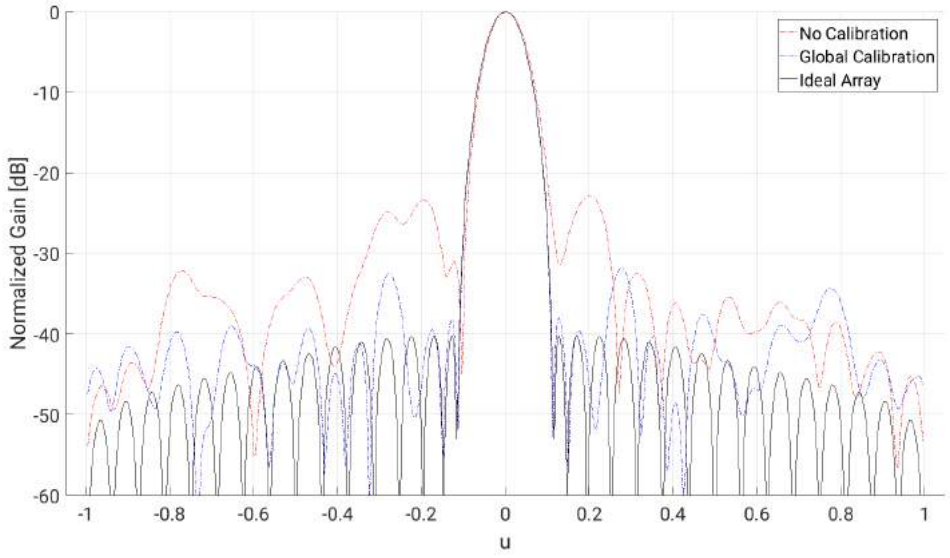


Figure 4.29: Limit of local calibration before a maximum 2 dB degradation in SLL for (x,z)-position element and subarray errors at $\sigma = 0.05\lambda$

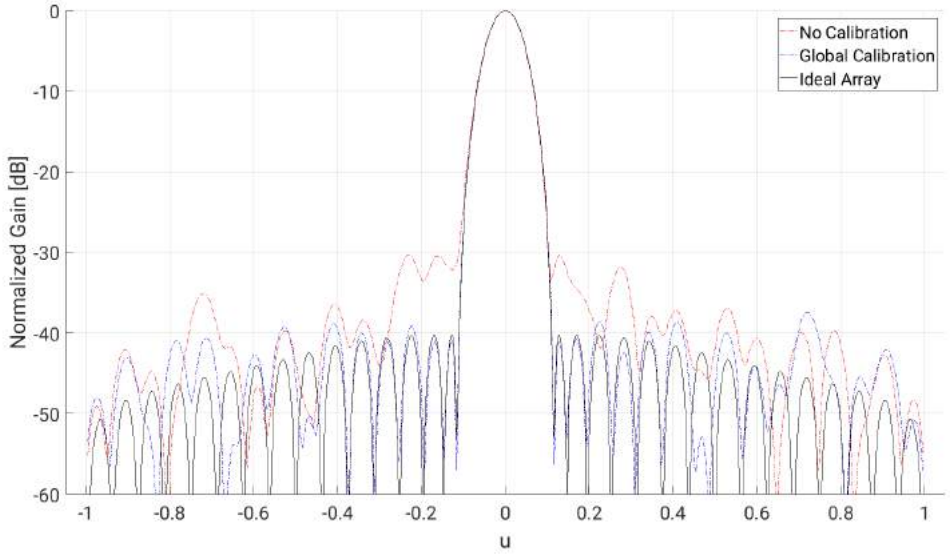


Figure 4.30: Limit of local calibration before a maximum 2 dB degradation in SLL for (x,y,z)-position element and subarray errors at $\sigma = 0.03\lambda$

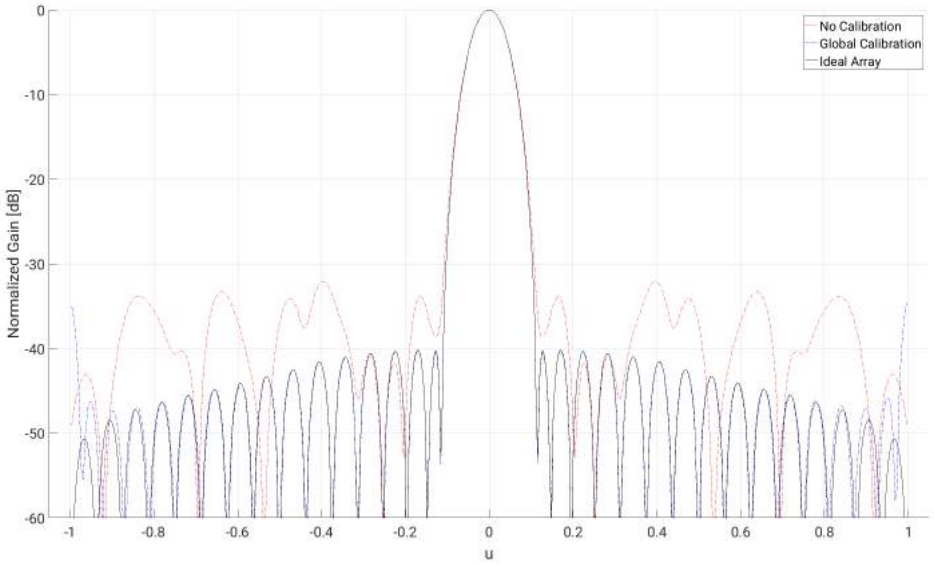


Figure 4.31: Limit of global calibration before a maximum 2 dB degradation in SLL for x-position element and subarray errors at $\sigma = 0.09\lambda$

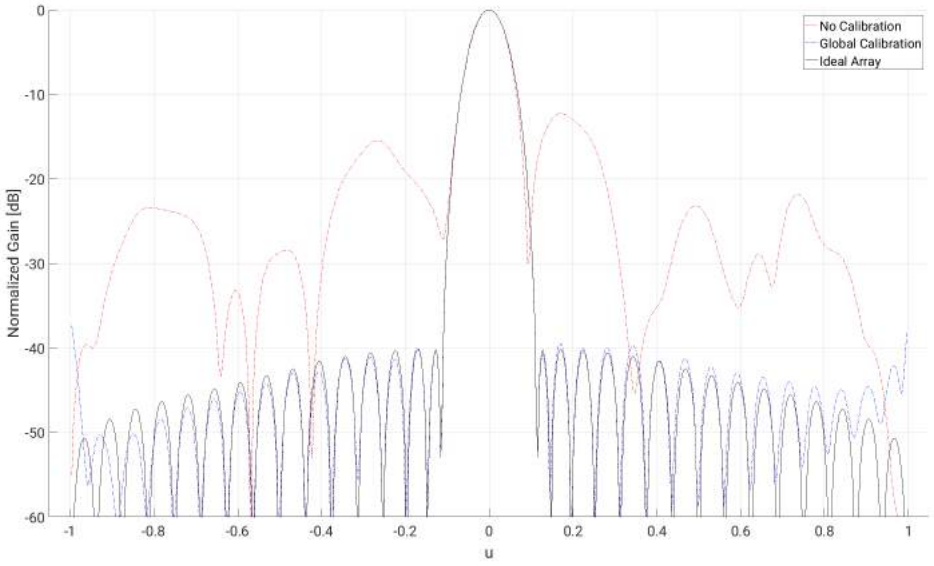


Figure 4.32: Limit of global calibration before a maximum 2 dB degradation in SLL for z-position element and subarray errors at $\sigma = 0.12\lambda$

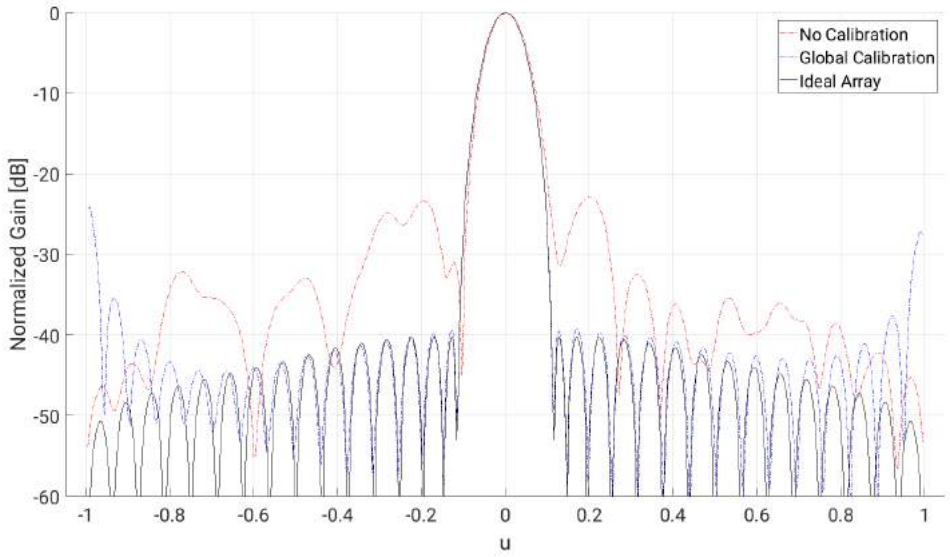


Figure 4.33: Limit of global calibrationbefore a maximum 2 dB degradation in SLL for (x,z)-position element and subarray errors at $\sigma = 0.05\lambda$

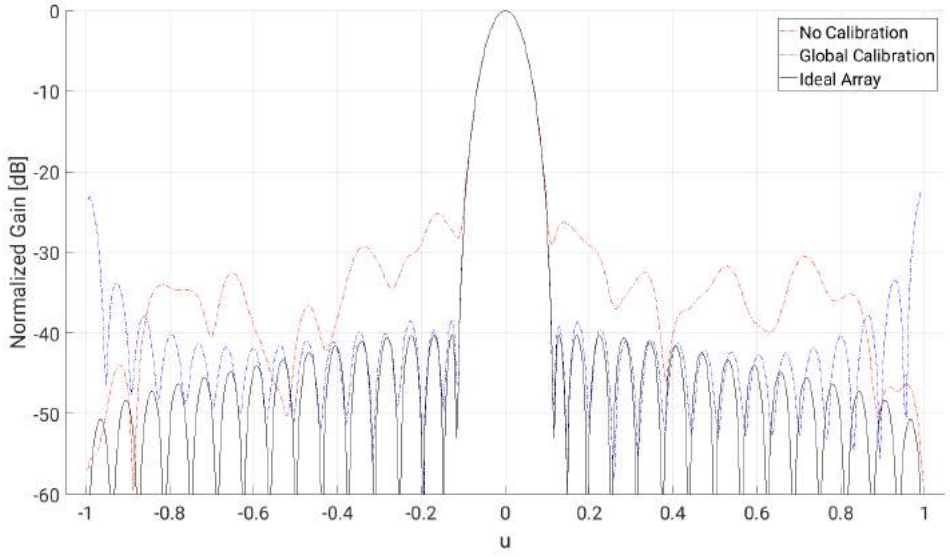


Figure 4.34: Limit of global calibration before a maximum 2 dB degradation in SLL for (x,y,z)-position element and subarray errors at $\sigma = 0.04\lambda$

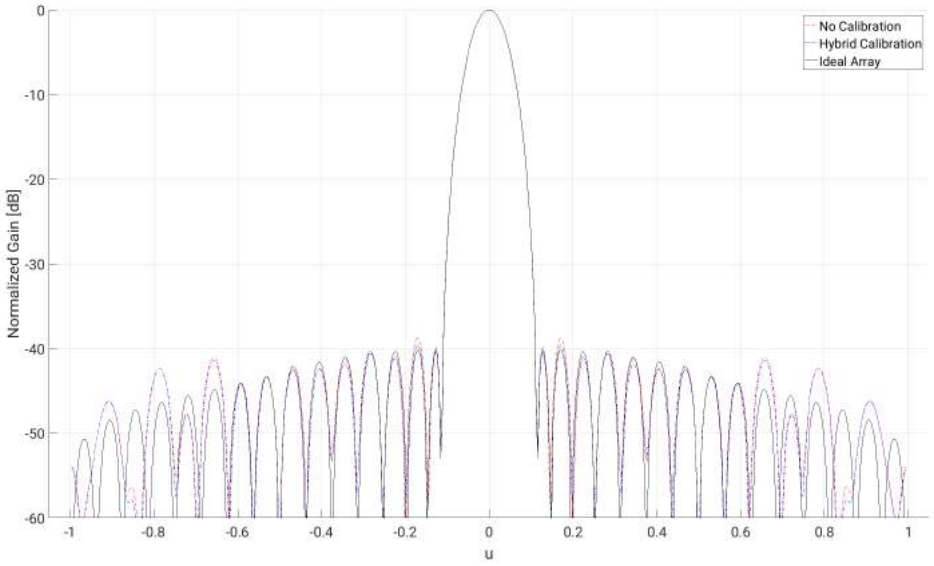


Figure 4.35: Limit of hybrid calibration before a maximum 2 dB degradation in SLL for x-position element and subarray errors at $\sigma = 0.02\lambda$

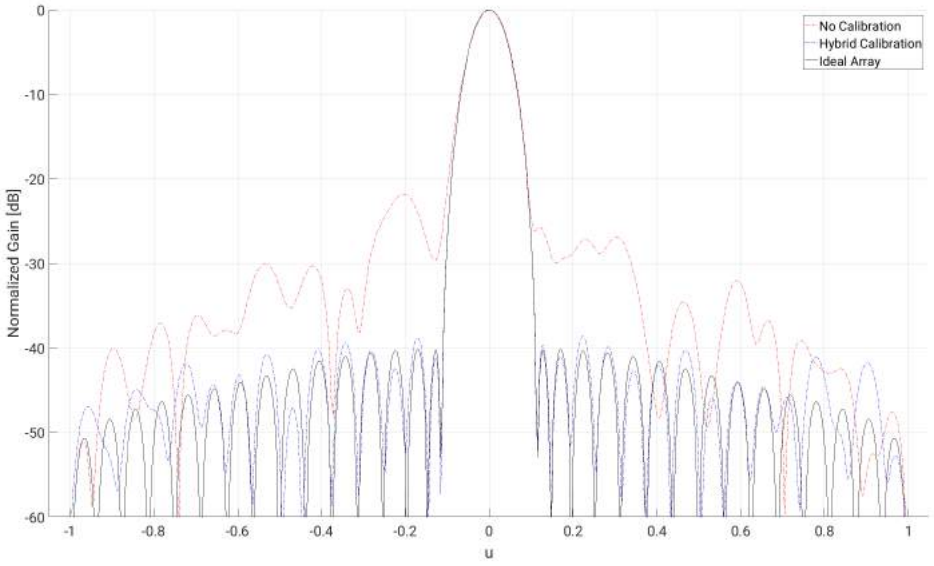


Figure 4.36: Limit of hybrid calibration before a maximum 2 dB degradation in SLL for z-position element and subarray errors at $\sigma = 0.07\lambda$

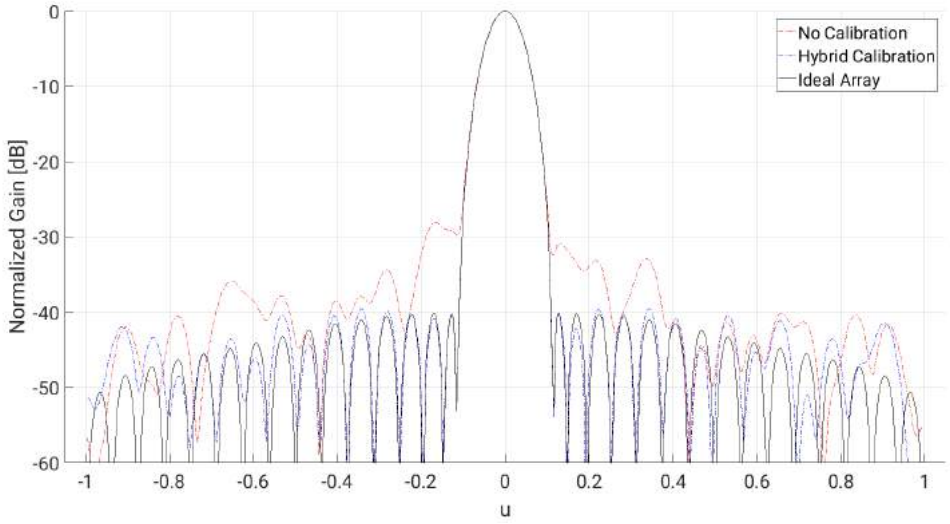


Figure 4.37: Limit of hybrid calibration before a maximum 2 dB degradation in SLL for (x,z)-position element and subarray errors at $\sigma = 0.02\lambda$

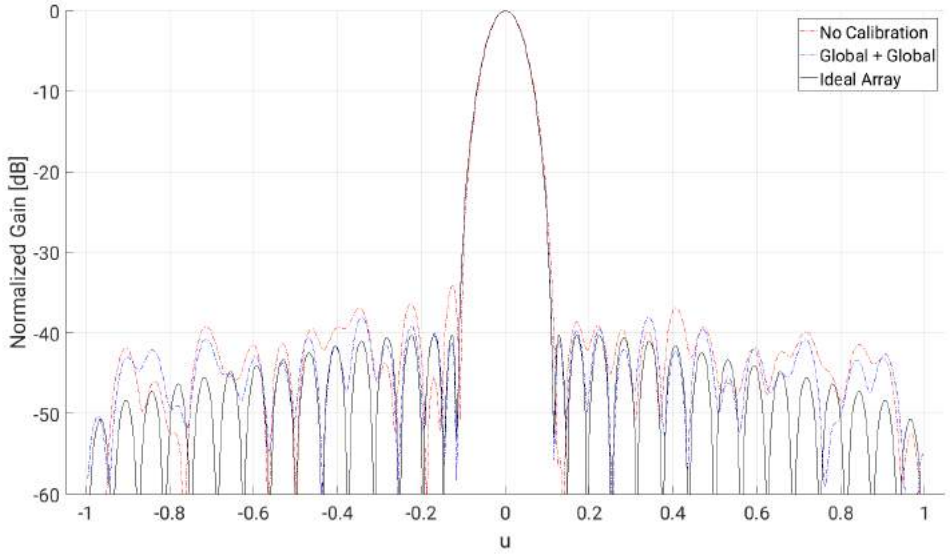


Figure 4.38: Limit of hybrid calibration before a maximum 2 dB degradation in SLL for (x,y,z)-position element and subarray errors at $\sigma = 0.02\lambda$

Table 4.3: Scanning deviation in degrees where degradation is 1, 2, and 3 dB for each calibration type.

Calibration type	1 dB	2 dB	3 dB
Local Cal.	6°	12°	18°
Hybrid Cal.	12°	19°	27°
Global Cal.	3°	5°	7°

Table 4.4: Required number of correction matrices for maintaining a degradation lower than 1, 2, or 3 dB for each calibration type.

Calibration type	1 dB	2 dB	3 dB
Local Cal.	132	42	25
Hybrid Cal.	42	25	16
Global Cal.	504	195	110

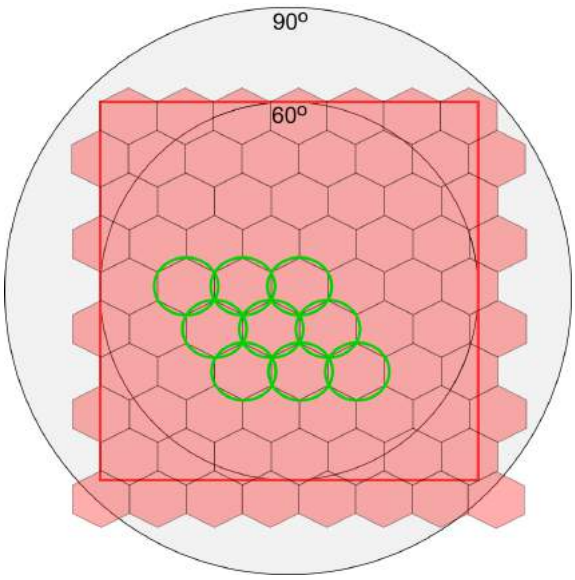


Figure 4.39: Method of packing hexagons to calculate the approximate number of correction matrices required to cover the scanning area.

4.3 Test Antenna

4.3.1 Reflection Coefficients

The test antenna embedded element reflection coefficients ($S_{0101} - S_{1616}$) measured with a vector network analyzer (VNA) are shown in Figure 4.40. We can see that the reflection coefficients have very good values from 8-9.5 GHz, but is worse than -10dB from around 9.5-10.5 GHz. This can be compared with the isolated element reflection coefficients of the HFSS antenna element design and the embedded element and element scan reflection coefficients of the HFSS array design shown in Figure 4.41. We can conclude that the curve fits the simulation, but with a shift towards lower frequencies in the manufactured array antenna compared to the HFSS design. The best performance regarding reflection parameters should be around 9.5 GHz when using all elements simultaneously.

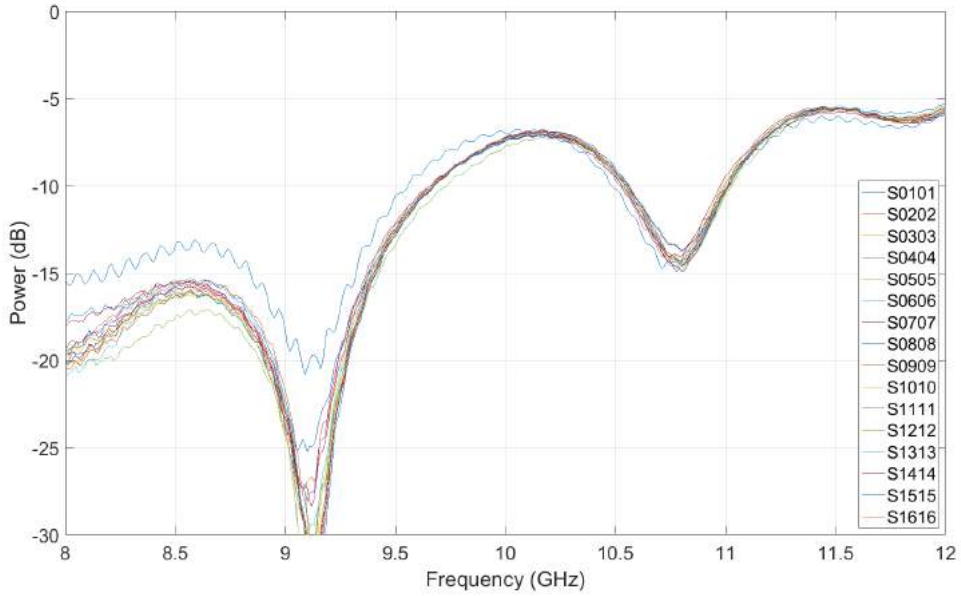


Figure 4.40: Embedded element reflection coefficients for each elements of the test antenna.

4.3.2 Radiation Pattern

The directive gain of each element can be seen in Figure 4.42 for $\varphi = 0°$, which is the plane of the array (xz-plane), and $\varphi = 90°$, which is the plane perpendicular to the array (yz-plane). We can observe that the elements affect each other significantly due to coupling in the plane of the array while the pattern is more similar to the individual element in the perpendicular plane. The elements closer to the edge of the array is exhibiting less ideal patterns and are responsible for the gain dips close to ± 50 degrees in Figure 4.42a, while the centre elements have a better performance. As such, applying the taylor taper when using the array is good for

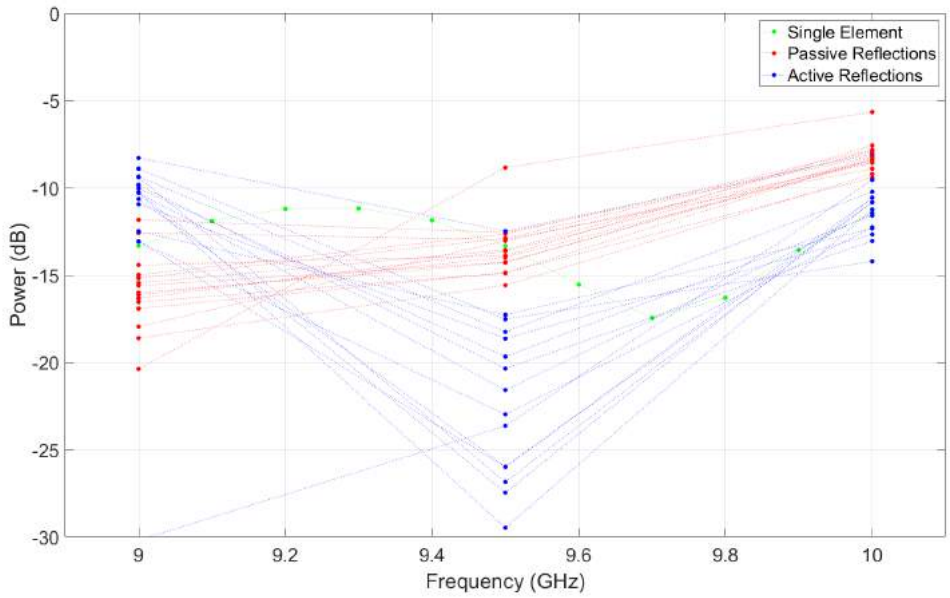


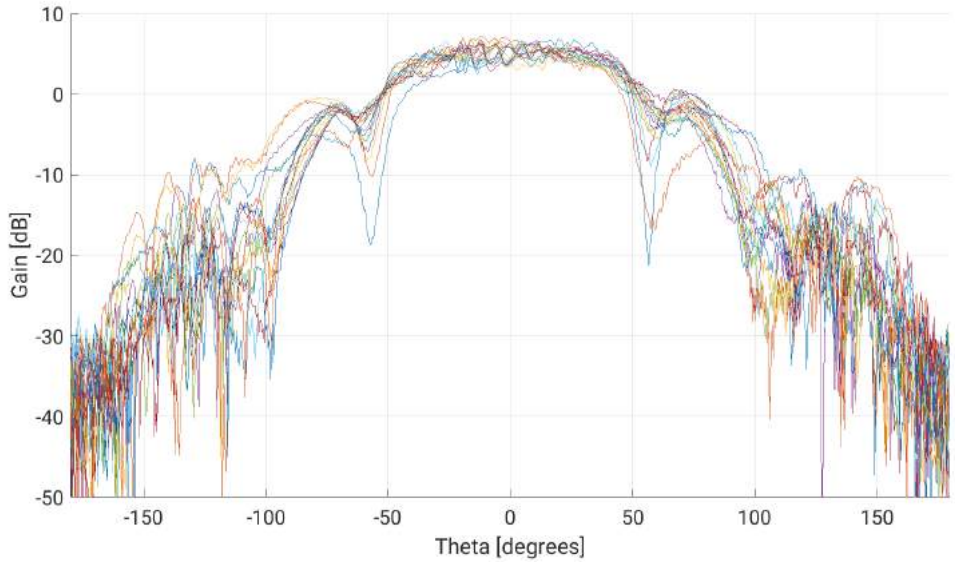
Figure 4.41: Isolated element reflections (single) and embedded element (passive) and element scan (active) reflection coefficients for each elements of the trial array antenna from HFSS design.

reducing the influence of the edge elements in addition to reducing the SLL. The measured radiation pattern of the array is shown in Figure 4.43a. As expected the pattern in the xz-plane is non-ideal due to the position errors of the element and the coupling between them while the pattern in the yz-plane is very close to the optimal pattern. Applying a 30 dB Taylor taper to the array will yield the pattern shown in Figure 4.43b where the discrepancy between the desired SLL and the resulting one is around 5 dB at 10 Ghz. Clearly, some calibration effort is required.

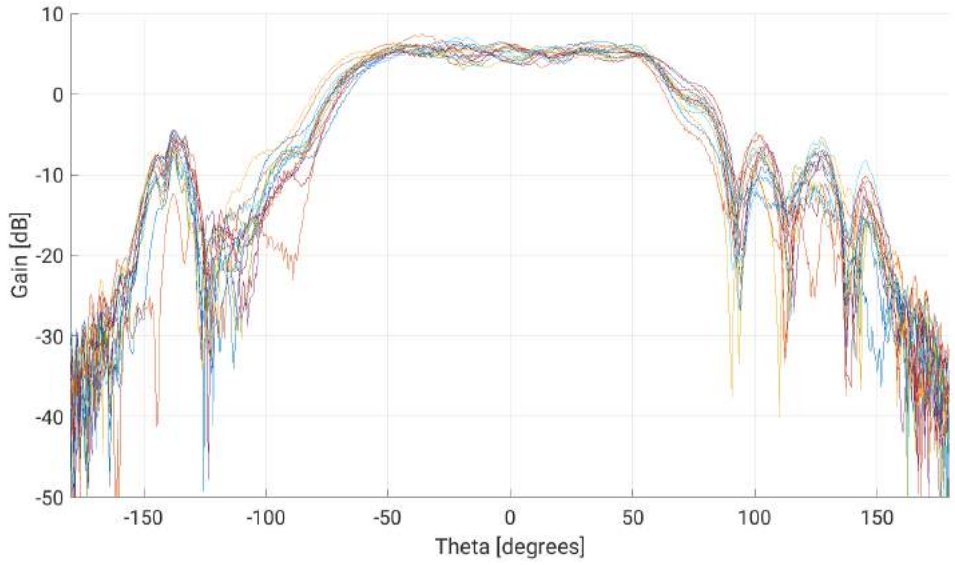
4.3.3 Calibration of Test Antenna

Using the element radiation pattern in Figure 3.2b and element positions in Table 3.3 we can apply the global and local calibration of section 4.2 to calibrate the theoretical test antenna array. This version of the test antenna does not have any coupling between neighboring antennas and no losses due to mismatch, although the radiation efficiency of the element is included, and therefore constitutes the best possible results for the calibration. The optimal array pattern based on the HFSS element pattern might also be a useful goal pattern when calibrating the measured antenna. The results can be seen in Figure 4.44. Both the global an local calibration achieves the goal of the calibration with the global calibration maintaining a 30 dB SLL in the entire uv-space and the local calibration maintaining the same SLL a distance up to 0.5 from the scanned main beam. Only a very minor degradation can be seen for both methods, which can be corrected by applying a slightly stricter excitation taper. Moving on to the calibration of the measured antenna, the results are mixed. The results applied to 9, 9.25, 9.5, 9.75, and 10 GHz and can be seen in Figure 4.45, 4.46, 4.47, 4.48, and 4.49 where the 'weighted pattern' is the pattern when the

elements are weighted with standard 30 dB Taylor weights. Unfortunately, the element pattern from the HFSS simulation proves to be insufficient in making a successful goal pattern for the calibration. Therefore, the goal pattern used is based on one of the centre elements radiation patterns duplicated sixteen times to form an approximately optimal array radiation pattern with -30 dB SLL. As can be seen in the results there is a step in the main lobe when using this method. This is due to the way the measurement of the radiation pattern is made with a delay in collecting the measurement data as the antenna is scanned across different angles. For 9.5 and 9.75 GHz a stronger taper is required for a successful calibration, but for the other frequencies the normal 30 dB taper is enough. Local calibration works well for all frequencies except 9 GHz and global calibration is achieved in the three first frequencies but fail for 9.75 and 10 GHz. These failures could possibly be mitigated by tweaking the goal pattern somewhat or if that fails, a less simplistic way of forming the goal pattern might yield better results. Results for non boresight scanning are similar in nature.

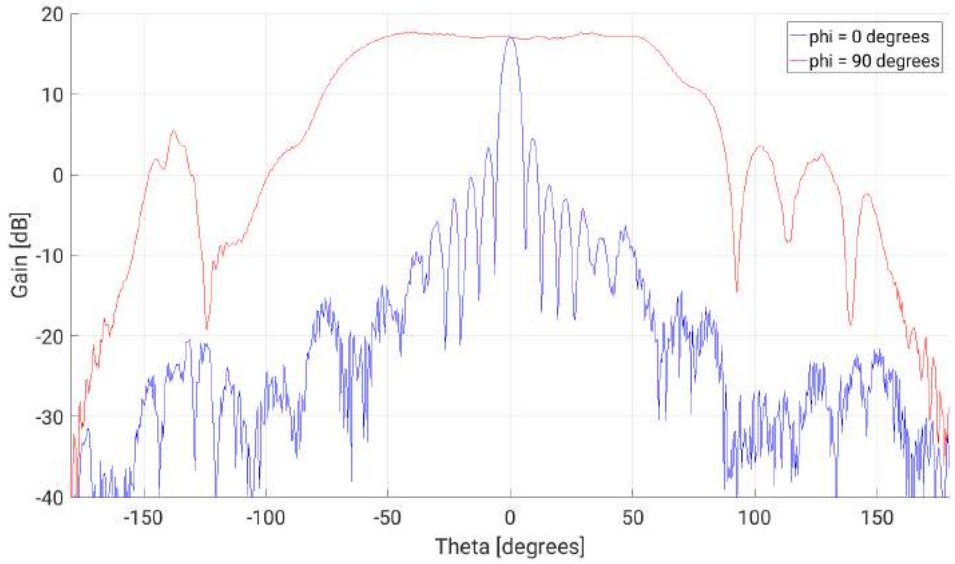


(a)

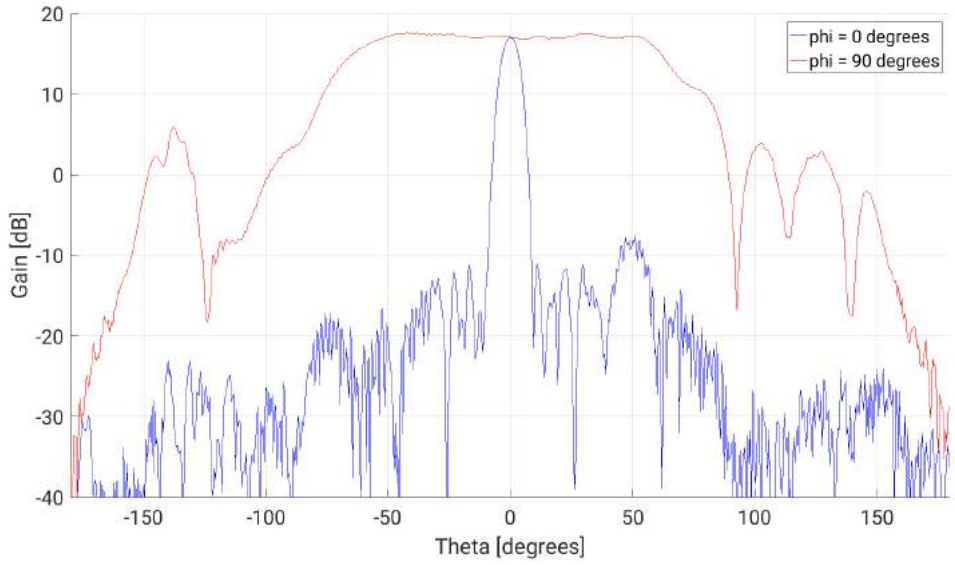


(b)

Figure 4.42: Trial antenna elements radiation pattern at 10 GHz for a) $\varphi = 0°$ and b) $\varphi = 90°$



(a)



(b)

Figure 4.43: Trial antenna array radiation pattern at 10 GHz for $\varphi = 0°$ and $\varphi = 90°$ with a) uniform excitation b) Taylor 30 dB excitation.

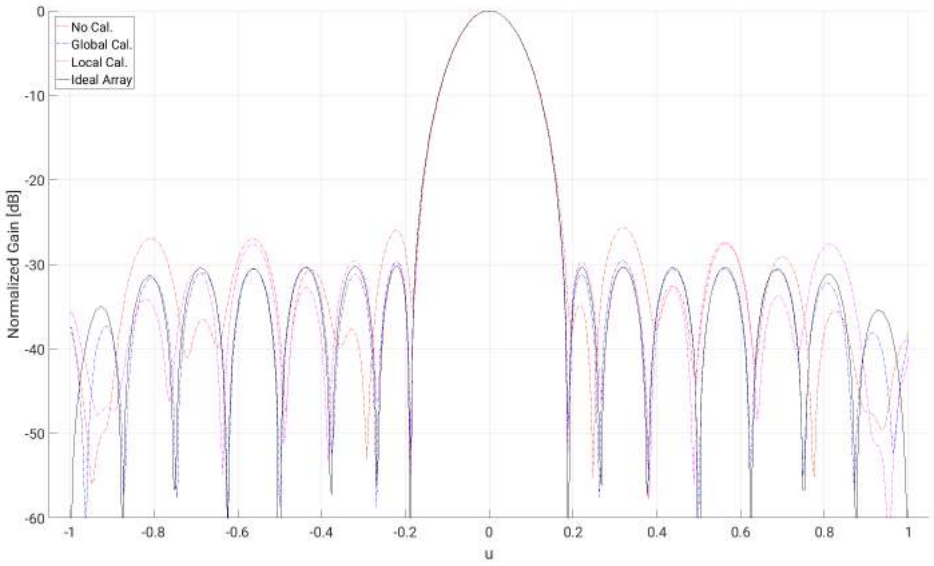


Figure 4.44: Trial antenna theoretical radiation pattern and calibration at v=0. Based on simulated radiation pattern of trial antenna element in HFSS.

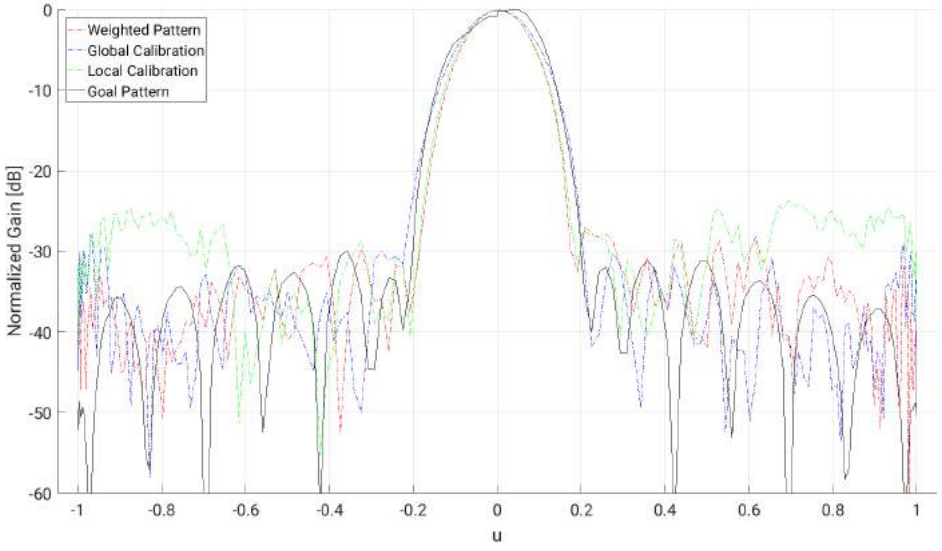


Figure 4.45: Measured trial antenna calibration for 9 GHz.

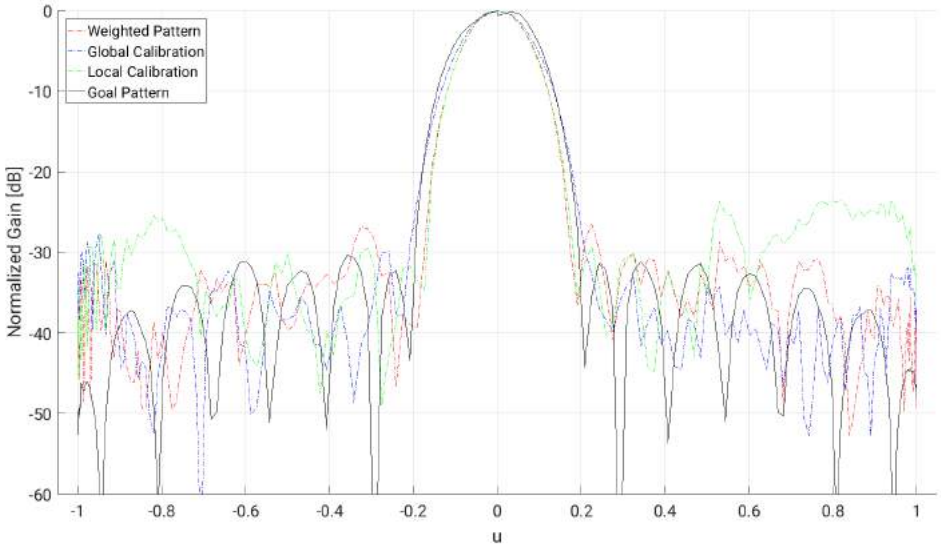


Figure 4.46: Measured trial antenna calibration for 9.25 GHz.

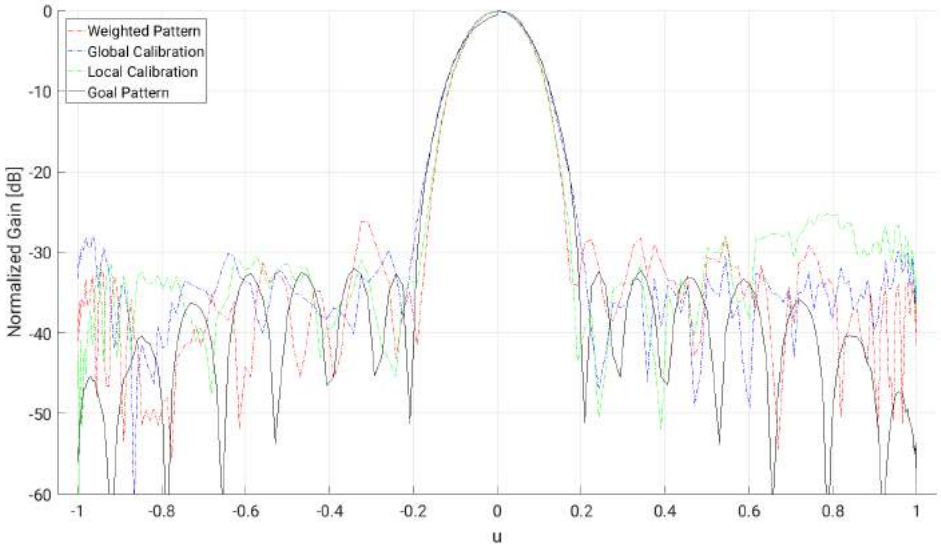


Figure 4.47: Measured trial antenna calibration for 9.5 GHz.

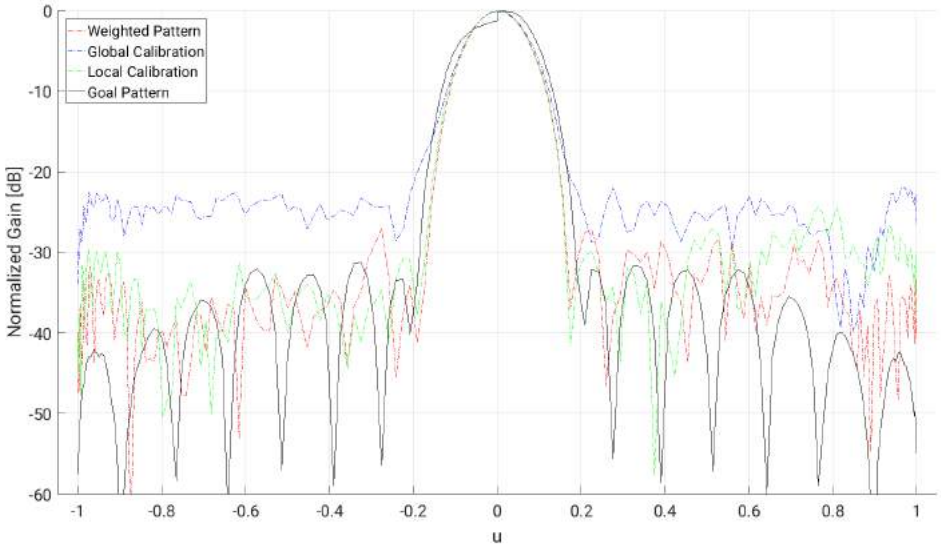


Figure 4.48: Measured trial antenna calibration for 9.75 GHz.

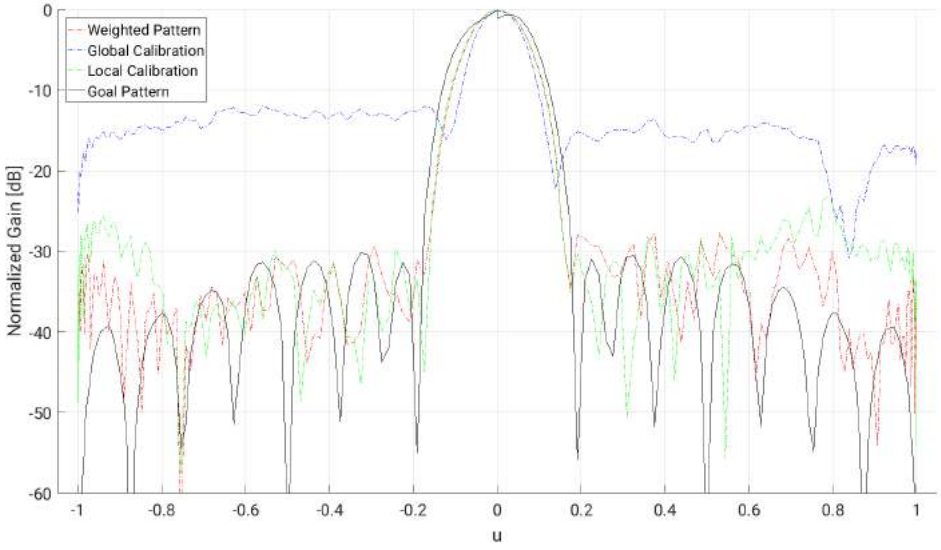


Figure 4.49: Measured trial antenna calibration for 10 GHz.

5
Conclusion

It is clear that the suggested model is a robust method for reducing the SLL and assuring that the desired radiation pattern is fullfilled, given that the mechanical errors are reasonably small. This report only presented the effectiveness when applied to errors caused by translational mechanical tolerances, but there is evidence that the method should work well for any type of mechanical error. Further study needs to be done into what configuration of manufacturing tolerances and error compensation is advisable in these cases.

Despite the largest sources of errors coming from translational errors in the z dimension, these errors are by far the easiest to calibrate. It is more concerning if there are large translational errors in the xy-plane if the presented method is used. Furthermore, reducing the number of types of errors is crucial for each type of calibration and preferably only having a significant z-error left when manufacturing the array antenna. As expected, the fully digital array represent the largest possibility for compensating for mechanical errors. However, there are situations when local calibration might be permissible, given that there is a choice between the two. The method applied to a real manufactured and measured array antenna had mixed results due to the simple derivation of the goal pattern used. If a more successfull attempt is to be made the goal pattern needs to have more accurate models behind it.

A comparison of the results from this method needs to be done with the often used method at SAAB AB, where the phases of each element is aligned when subject to an incident plane wave. Technically, the local calibration with an infinitely narrow weighting should give the same results, but a more rigourus comparison should be made to see if this assumption holds.

In addition to radiation pattern errors caused by mechanical errors there are many other sources for these errors. For a real life active antenna array to be feasably calibrated these other sources must be taken into account. The approach made in this report is determine the amplitude and phase of the far field of each antenna element and then apply signal processing methods to modify them in a manner as to minimize the error between the resulting radiation pattern and the goal pattern. This method is seemingly compatible with these other sources. Simply, combine this method with whatever other methods for handling the other error sources is used and compute the far field function for each of the elements. Then apply the same compensation method used in this report with the new element radiation patterns.

Examples of phenomena that will affect the radiation patterns of the elements other than mechanical errors are coupling, edge effects, nearby radiation sources, radomes, and internal channel errors.

The calculations done in the report has been excluding the influence of noise in the channel and this is something that must be investigate if one wishes to calibrate a real antenna array successfully, especially if it is an active electronically scanned array that have many internal sources of noise. In general, there is much potential in using a fully digital beamforming to compensate for mechanical errors and other error sources, but more research must be done for each source to ensure success.

Bibliography

[1] F. Ugolini, S. Sabatini, and M. E. Marini, "Del_c.6.3.4_1: Antenna position tolerance technical report (1/3)", SELEX Sistemi Integrati S.p.A., Elettronica S.p.A, Saab AB, Gothenburg, Tech. Rep., Feb. 2008.

[2] C. A. Balanis, *Antenna Theory, Analysis and Design*, 4th edition. Hoboken, New Jersey: John Wiley & Sons, Inc., 2016, ISBN: 978-1-118-642060-1.

[3] P. Kildal, *Foundations of Antenna Engineering, A Unified Approach for Line-of-Sight and Multipath*. Gothenburg, Sweden: Kildal Antenn AB, 2015, ISBN: 978-91-637-8515-3.

[4] W. Wirth, *Radar Techniques Using Array Antennas*. London, United Kingdom: The Institution of Electrical Engineers, 2001, ISBN: 0-85296-798-5.

[5] R. J. Mailloux, *Phased Array Antenna Handbook*, 2nd edition. 685 Canton Street, Norwood, MA 02062: Artech House, Inc., 2005, ISBN: 1-58053-689-1.

[6] L. Råde and B. Westergren, *Mathematics Handbook for Science and Engineering*, 4th edition. Lund, Sweden: Studentlitteratur, 1998, ISBN: 91-44-00839-2.

[7] S. O. Rice, "Mathematical analysis of random noise", *The Bell System Technical Journal*, vol. 23, p. 282, Jul. 1944.

[8] G. L. Stüber, *Principles of Modern Communication*, 2nd edition. New York: Kluwer Academic Publishers, 2002, ISBN: 0-306-47315-1.

[9] G. E. P. Box, J. S. Hunter, and W. G. Hunter, *Statistics for Experimenters, Design, Innovation, and Discovery*, 2nd edition. Hoboken, New Jersey: John Wiley & Sons, Inc., 2005, ISBN: 0-471-71813-0.

[10] M. Lanne, A. Lundgren, and M. Viberg, "Calibrating an array with scan dependent errors using a sparse grid", pp. 2242–2246, Oct. 2006, ISSN: 1058-6393. DOI: 10.1109/ACSSC.2006.355168.

[11] ——, "Optimized beamforming calibration in the presence of array imperfections", vol. 2, pp. 973–976, Apr. 2007, ISSN: 1520-6149. DOI: 10.1109/ICASSP.2007.366400.

[12] R. S. Elliott, "Mechanical and electrical tolerances for two-dimensional scanning antenna arrays", *IRE Transactions on Antennas and Propagation*, vol. 6, pp. 114–120, Jan. 1958.

A

MATLAB code

A.1 Error Characterisation Code

```
%% Uniform Planar Array Model - by Josef Ydreborg
% Constants ..............................................................

f0 = 10e9;                          % Central frequency used
c0 = 299792458;                     % Speed of light in vacuum
lam0 = c0/f0;                       % Wavelength of central frequency
k0 = (2*pi)/lam0;                   % Wavenumber of central frequency

%% Array Information _____________________________________________________

M = 32;                             % Number of elements along x-axis
N = 32;                             % Number of elements along y-axis
m = -(M-1)/2:1:(M-1)/2;             % Element indexes along x-axis
n = -(N-1)/2:1:(N-1)/2;             % Element indexes along y-axis

Msub = 4;                           % Number of elements along subarray x-axis
Nsub = 4;                           % Number of elements along subarray y-axis

dx = lam0/2;                        % Desired element spacing along x-axis
Lx = M*dx;                          % Length of array along x-axis
x = m*dx;                           % Element positions along x-axis

dy = lam0/2;                        % Desired element spacing along y-axis
Ly = N*dy;                          % Length of array along y-axis
y = n*dy;                           % Element positions along y-axis

[Xopt,Yopt] = meshgrid(x,y);        % Optimal element positions in x,y-axis
Zopt = zeros(size(Xopt));           % Optimal element positions in z-axis

%% Phase Information _____________________________________________________

res = 250;                          % resolution of angles = (2*res+1)^2
u = -1:1/res:1;                     % angles used in theta-dimension
v = -1:1/res:1;                     % angles used in phi-dimension

[Ui,Vi] = meshgrid(u,v);                % 2D mesh of U,V angles
uv_filt = double((Ui.^2+Vi.^2) <= 1);
uv_filt(uv_filt == 0) = nan;

U = Ui.*uv_filt;    % converts entries to NaN where U^2+V^2 < 1 is not
V = Vi.*uv_filt;    % fullfilled
W = real(sqrt(ones(size(U))-U.^2-V.^2));

% Steering ...............................................................

theta0 = 0;             % steering angle in theta-dimension
phi0 = 0;               % steering angle in phi-dimension

u0 = sin(theta0)*cos(phi0);   % steering angles converted to u-space
v0 = sin(theta0)*sin(phi0);   % steering angles converted to v-space

betaX = -k0*u0;
```

```
53 betaY = -k0*v0;
54
55 % Limit simulation angles to principal side lobes .....................
56
57 [u0val,u0ind] = min(abs(U-u0),[],2);     % Finding columnn index where the
58 u0ind = u0ind(round(length(u0ind)/2)); % principal side lobes are located
59 [v0val,v0ind] = min(abs(U-v0),[],2);     % Finding row index where the
60 v0ind = v0ind(round(length(v0ind)/2)); % principal side lobes are located
61
62 U = [U(u0ind,:);U(:,v0ind)']; % This Restricts U, V and W coordinates to
63 V = [V(u0ind,:);V(:,v0ind)']; % only the principal side lobe axes.
64 W = [W(u0ind,:);W(:,v0ind)'];
65
66 % Other Coordinate Systems ..............................................
67
68 TH = asin(sqrt(U.^2+V.^2));      % angles in spherical coord. theta
69 PH = atan(U./V);                 % angles in spherical coord. phi
70
71 EL = asin(sin(PH).*sin(TH));     % angles in elevation coord.
72 AZ = atan(cos(PH).*tan(TH));     % angles in azimuth coord.
73
74 %% Tapering (R.J.Mailloux, Phase Array Antenna Handbook, p.121-128) _______
75
76 R = -40;                         % desired sidelobe level [dB]
77 r = 10^(-R/20);                  % desired sidelobe ratio [linear]
78
79 A = (1/pi)*acosh(r);
80 ns = 11;
81 dn = 1:1:ns-1;
82 sigma = ns/(A^2+(ns-1/2)^2)^(1/2);
83 zn = sigma*(A^2+(dn-1/2).^2).^(1/2);
84 p = ones(1,ns-1);
85 for i=1:1:ns-1
86     p = p .* (1-dn.^2./zn(i).^2);
87 end
88 Fm = ((factorial(ns-1)^2)./(factorial(ns-1+dn).*factorial(ns-1-dn))).*p;
89
90 wx = ones(1,M);
91 wy = ones(1,N);
92 sx = zeros(1,M);
93 sy = zeros(1,N);
94 for i=1:1:ns-1
95     sx = sx + Fm(i).*cos((2*pi*i*x)/Lx);
96     sy = sy + Fm(i).*cos((2*pi*i*y)/Ly);
97 end
98 wx = wx + 2*sx;                  % tapering values for x-axis
99 wy = wy + 2*sy;                  % tapering values for y-axis
100
101 [IX,IY] = meshgrid(wx,wy);
102 I = IX.*IY;                      % tapering values in xy-plane
103
104 %% Antenna element pattern ____________________________________________
105
106 EP = sqrt(W);          % Pattern of individual element sqrt(cos(theta))
107 EPdB = 10*log10(EP);       % Converting directivity to dB
108
109 %% Optimal Array Factor ____________________________________________
110
111 beta = Xopt*betaX + Yopt*betaY; % phases of element excitations
112 exc = I.*exp(1i*beta);          % matrix of element excitations (amp,phase)
113 AF = zeros(size(U));            % initiating array factor
114
115 % Calculates array factor for each element and summing them
116 for i=1:1:M*N
117         AF = AF + exc(i)*exp(1i*k0*(Xopt(i)*U+Yopt(i)*V+Zopt(i)*W));
118 end
119 Gopt = EP.*AF;
120
121 Gopt_dB = 20*log10(abs(Gopt));              % converts AF to dB
122 Gopt_norm = abs(Gopt)./max(max(abs(Gopt))); % normalizes AF
```

```
123 Gopt_normdB = 20*log10(Gopt_norm);          % normalized AF to dB
124
125 % Removing 'inf' values in dB scale. Needed to plot as contour.
126 Gopt_dB(Gopt_dB==-inf) = min(min(Gopt_dB(Gopt_dB>-inf)))-100;
127 Gopt_normdB(Gopt_normdB==-inf) = ...
128     min(min(Gopt_normdB(Gopt_normdB>-inf)))-100;
129
130 %% Error Loop ____________________________________________________________
131 iter = 1000;              % number of simulations of erronous array patterns
132 % Greg = zeros(length(u),length(v),iter); % register for erronious patterns
133 Greg = zeros(2,length(u),iter); % register for simulating only principal
134                                 % side lobes
135
136 mu = 0;                   % mean value of position errors
137 sigx = 300e-6;            % standard deviation of elements in x-axis [m]
138 sigy = 300e-6;            % standard deviation of elements in y-axis [m]
139 sigz = 300e-6;            % standard deviation of elements in z-axis [m]
140 subx = 300e-6;            % standard deviation of subarrays in x-axis [m]
141 suby = 300e-6;            % standard deviation of subarrays in y-axis [m]
142 subz = 300e-6;            % standard deviation of subarrays in z-axis [m]
143
144 beta = Xopt*betaX + Yopt*betaY; % phases of element excitations
145 exc = I.*exp(1i*beta);          % matrix of element excitations (amp,phase)
146
147 for k=1:iter
148 % Generating Position Errors for pattern k
149 X = Xopt; Y = Yopt; Z = Zopt;
150
151 subX = normrnd(mu,subx,N/Nsub,M/Msub);
152 subY = normrnd(mu,suby,N/Nsub,M/Msub);
153 subZ = normrnd(mu,subz,N/Nsub,M/Msub);
154 subX = kron(subX,ones(Nsub,Msub));      % x-axis errors of subarrays
155 subY = kron(subY,ones(Nsub,Msub));      % y-axis errors of subarrays
156 subZ = kron(subZ,ones(Nsub,Msub));      % z-axis errors of subarrays
157
158 sigX = normrnd(mu,sigx,N,M);            % x-axis errors of elements
159 sigY = normrnd(mu,sigy,N,M);            % y-axis errors of elements
160 sigZ = normrnd(mu,sigz,N,M);            % z-axis errors of elements
161
162 X = X + sigX + subX;     % erroneous x-position of each element
163 Y = Y + sigY + subY;     % erroneous y-position of each element
164 Z = Z + sigZ + subZ;     % erroneous z-position of each element
165
166 % Calculates and sums the array factor for each element
167 AF = zeros(size(U));     % initiating array factor
168 for i=1:1:M*N
169         AF = AF + exc(i)*exp(1i*k0*(X(i)*U+Y(i)*V+Z(i)*W));
170 end
171 G = EP.*AF;              % Applying antenna element pattern to array factor
172 Greg(:,:,k) = G;         % Assigning pattern k
173
174 end
175
176 %% Calculations __________________________________________________________
177
178 Gavg = mean(abs(Greg),3); % Average o the erronous simulations
179 Gsum = sum((abs(Greg)-repmat(Gavg,1,1,iter)).^2,3);
180 Gsig = sqrt(Gsum/(iter-1)); % The standard deviation of each set of
181                             % samples for each viewinfg angle
182 Grel = Gsig./max(max(abs(Gopt))); % Normalized standard deviation
183
184 Gs1 = abs(Gopt) + abs(Gsig);     % mu + sigma
185 Gs2 = abs(Gopt) + 2*abs(Gsig);   % mu + 2*sigma
186 Gs3 = abs(Gopt) + 3*abs(Gsig);   % mu + 3*sigma
187
188 % Converting all sets into normalized and dB versions and removing 'inf'
189 % values to make plotting easier:
190
191 Gavg_dB = 20*log10(abs(Gavg));
192 Gavg_norm = abs(Gavg)./max(max(abs(Gavg)));
```

```
193 Gavg_normdB = 20*log10(Gavg_norm);
194 Gavg_dB(Gavg_dB==-inf) = min(min(Gavg_dB(Gavg_dB>-inf)))-100;
195 Gavg_normdB(Gavg_normdB==-inf) = ...
196     min(min(Gavg_normdB(Gavg_normdB>-inf)))-100;
197
198 Gsig_dB = 20*log10(abs(Gsig));
199 Gsig_norm = abs(Gsig)./max(max(abs(Gsig)));
200 Gsig_normdB = 20*log10(Gsig_norm);
201
202 Gs1_dB = 20*log10(abs(Gs1));
203 Gs1_norm = abs(Gs1)./max(max(abs(Gs1)));
204 Gs1_normdB = 20*log10(Gs1_norm);
205
206 Gs2_dB = 20*log10(abs(Gs2));
207 Gs2_norm = abs(Gs2)./max(max(abs(Gs2)));
208 Gs2_normdB = 20*log10(Gs2_norm);
209
210 Gs3_dB = 20*log10(abs(Gs3));
211 Gs3_norm = abs(Gs3)./max(max(abs(Gs3)));
212 Gs3_normdB = 20*log10(Gs3_norm);
213
214 Grel_dB = 20*log10(abs(Grel));
215 Grel_norm = abs(Grel)./max(max(abs(Grel)));
216 Grel_normdB = 20*log10(Grel_norm);
217 Grel_dB(Grel_dB==-inf) = min(min(Grel_dB(Grel_dB>-inf)))-100;
218 Grel_normdB(Grel_normdB==-inf) = ...
219     min(min(Grel_normdB(Grel_normdB>-inf)))-100;
```

A.2 Error Compensation Code

A.2.1 Analog/Digital Array

```
1  %% Hybrid analog/digital Planar Array Calibration Model
2  % by Josef Ydreborg
3  %% Constants ____________________________________________________________
4  tic
5  f0 = 10e9;                          % Central frequency
6  c0 = 299792458;                     % Speed of light in vacuum
7  lam0 = c0/f0;                       % Wavelength of central frequency
8  k0 = (2*pi)/lam0;                   % Wavenumber of central frequency
9
10 %% ____________________ Section 1: Array Information ____________________
11 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
12 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
13 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
14 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
15 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
16 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
17 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
18 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
19 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
20 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
21
22 %% Ideal Array Information ..............................................
23 % Sensor Array is in the x-y plane and the direction of propagation is
24 % z-axis. As long as the number of elements in the array and subarrays
25 % are powers of two, the code should work well. Subarray size of one
26 % element works but keep in mind that calibrations based on subarrays
27 % are invalid in this case.
28
29 M = 2^5;                            % Number of elements along x-axis
30 N = 2^5;                            % Number of elements along y-axis
31 m = -(M-1)/2:1:(M-1)/2;             % Element indeces along x-axis
32 n = -(N-1)/2:1:(N-1)/2;             % Element indeces along y-axis
33
34 NumSubX = 2^3;                      % Number of Subarrays along x-axis
```

```
35  NumSubY = 2^3;                % Number of Subarrays along y-axis
36  NumSub = NumSubX*NumSubY;
37
38  Msub = M/NumSubX;             % Number of elements in subarray along x-axis
39  Nsub = N/NumSubY;             % Number of elements in subarray along y-axis
40
41  dx = lam0/2;                  % Ideal element spacing along x-axis
42  dy = lam0/2;                  % Ideal element spacing along y-axis
43  Lx = M*dx;                    % Length of array along x-axis
44  Ly = N*dy;                    % Length of array along y-axis
45
46  x = m*dx;                     % Ideal element positions along x-axis
47  y = n*dy;                     % Ideal element positions along y-axis
48  z = zeros(size(x));
49
50  [X0,Y0] = meshgrid(x,y);
51  Z0 = zeros(size(X0));
52
53  %% Position Errors ..........................................................
54
55  X = X0;
56  Y = Y0;
57  Z = Z0;
58
59  mu = 0;                       % mean value of position errors
60  sigx = 0.01*lam0;             % standard deviation of elements in x-axis
61  sigy = 0.01*lam0;             % standard deviation of elements in y-axis
62  sigz = 0.01*lam0;             % standard deviation of elements in z-axis
63  subx = 0.01*lam0;             % standard deviation of subarrays in x-axis
64  suby = 0.01*lam0;             % standard deviation of subarrays in y-axis
65  subz = 0.01*lam0;             % standard deviation of subarrays in z-axis
66
67  subX = normrnd(mu,subx,N/Nsub,M/Msub);
68  subY = normrnd(mu,suby,N/Nsub,M/Msub);
69  subZ = normrnd(mu,subz,N/Nsub,M/Msub);
70  subX = kron(subX,ones(Nsub,Msub));      % x-axis errors of subarrays
71  subY = kron(subY,ones(Nsub,Msub));      % y-axis errors of subarrays
72  subZ = kron(subZ,ones(Nsub,Msub));      % z-axis errors of subarrays
73
74  sigX = normrnd(mu,sigx,N,M);            % x-axis errors of elements
75  sigY = normrnd(mu,sigy,N,M);            % y-axis errors of elements
76  sigZ = normrnd(mu,sigz,N,M);            % z-axis errors of elements
77
78  X = X + sigX + subX;     % erroneous x-position of each element
79  Y = Y + sigY + subY;     % erroneous y-position of each element
80  Z = Z + sigZ + subZ;     % erroneous z-position of each element
81
82  %% Weighting Scheme .........................................................
83  % This code uses a taylor taper weighting scheme.
84
85  R = -40;                      % desired sidelobe level [dB]
86  r = 10^(-R/20);               % desired sidelobe ratio [linear]
87
88  A = (1/pi)*acosh(r);
89  nbar = ceil(2*A^2+0.5);       % SLL=-30dB; 4<nbar<7 for monotonic taper
90                                % SLL=-40dB; 7<nbar<11 for monotonic taper
91  dn = 1:1:nbar-1;
92  sigma = nbar/(A^2+(nbar-1/2)^2)^(1/2);
93  zn = sigma*(A^2+(dn-1/2).^2).^(1/2);
94  p = ones(1,nbar-1);
95  for i=1:1:nbar-1
96      p = p .* (1-dn.^2./zn(i).^2);
97  end
98  Fm = ((factorial(nbar-1)^2)./(factorial(nbar-1+dn).* ...
99       factorial(nbar-1-dn))).*p;
100
101 wx = ones(1,M); sx = zeros(1,M);
102 wy = ones(1,N); sy = zeros(1,N);
103 for i=1:1:nbar-1
104      sx = sx + Fm(i).*cos((2*pi*i*x)/Lx);
```

```
105         sy = sy + Fm(i).*cos((2*pi*i*y)/Ly);
106     end
107     wx = wx + 2*sx;
108     wy = wy + 2*sy;
109     [IX,IY] = meshgrid(wx,wy);
110     I = IX.*IY;                     % tapering values in xy-plane
111
112     %% Vectorizing positions and tapering ..................................
113     % Puts all element positions in vectors with subarray positions lumped
114     % together.
115
116     X0v = zeros(1,M*N);
117     Y0v = zeros(1,M*N);
118     Z0v = zeros(1,M*N);
119     Xv = zeros(1,M*N);
120     Yv = zeros(1,M*N);
121     Zv = zeros(1,M*N);
122     Iv = zeros(1,M*N);
123
124     count = 0;
125     for i=1:NumSubX
126         for j=1:NumSubY
127
128         temp = X0((i-1)*Msub+1:(i-1)*Msub+Msub,(j-1)*Nsub+1:(j-1)*Nsub+Nsub);
129         temp = temp(:); X0v(count*Msub*Nsub+1:(count+1)*Msub*Nsub) = temp;
130
131         temp = Y0((i-1)*Msub+1:(i-1)*Msub+Msub,(j-1)*Nsub+1:(j-1)*Nsub+Nsub);
132         temp = temp(:); Y0v(count*Msub*Nsub+1:(count+1)*Msub*Nsub) = temp;
133
134         temp = Z0((i-1)*Msub+1:(i-1)*Msub+Msub,(j-1)*Nsub+1:(j-1)*Nsub+Nsub);
135         temp = temp(:); Z0v(count*Msub*Nsub+1:(count+1)*Msub*Nsub) = temp;
136
137         temp = X((i-1)*Msub+1:(i-1)*Msub+Msub,(j-1)*Nsub+1:(j-1)*Nsub+Nsub);
138         temp = temp(:); Xv(count*Msub*Nsub+1:(count+1)*Msub*Nsub) = temp;
139
140         temp = Y((i-1)*Msub+1:(i-1)*Msub+Msub,(j-1)*Nsub+1:(j-1)*Nsub+Nsub);
141         temp = temp(:); Yv(count*Msub*Nsub+1:(count+1)*Msub*Nsub) = temp;
142
143         temp = Z((i-1)*Msub+1:(i-1)*Msub+Msub,(j-1)*Nsub+1:(j-1)*Nsub+Nsub);
144         temp = temp(:); Zv(count*Msub*Nsub+1:(count+1)*Msub*Nsub) = temp;
145
146         temp = I((i-1)*Msub+1:(i-1)*Msub+Msub,(j-1)*Nsub+1:(j-1)*Nsub+Nsub);
147         temp = temp(:); Iv(count*Msub*Nsub+1:(count+1)*Msub*Nsub) = temp;
148
149         count = count + 1;
150
151         end
152     end
153
154     %% _____________________ Section 2: Angle Information _____________________
155     % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
156     % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
157     % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
158     % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
159     % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
160     % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
161     % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
162     % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
163     % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
164     % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
165
166     % This piece of code defines what angles will be used for displaying the
167     % data and which angles are used for calibration. Additionally, the
168     % scanning angle is defined.
169
170     %% Visualisation angles ................................................
171
172     resp = 250;                     % resolution of angles = (2*res+1)^2
173     u = -1:1/resp:1;                % angles used in u-space
174     v = -1:1/resp:1;                % angles used in v-space
```

```
175
176 [Ui,Vi] = meshgrid(u,v);          % 2D mesh of U,V angles
177 uv_filt = double((Ui.^2+Vi.^2) <= 1);
178 uv_filt(uv_filt == 0) = nan;     % filter for removing illegal angles
179
180 % converts entries to NaN where U^2+V^2 < 1 is not fullfilled
181 U = Ui.*uv_filt;
182 V = Vi.*uv_filt;
183 W = abs(sqrt(ones(size(U))-U.^2-V.^2));
184
185 % vectorize non-NaN entries
186 nanind = find(~isnan(U));
187 Uv = U(nanind);
188 Vv = V(nanind);
189 Wv = W(nanind);
190
191 %% Calibration angles ..................................................
192
193 resc = 100;                        % resolution of angles = (2*res+1)^2
194 uc = -1:1/resc:1;                  % angles used in u-space
195 vc = -1:1/resc:1;                  % angles used in v-space
196
197 [Ui,Vi] = meshgrid(uc,vc);         % 2D mesh of U,V angles
198 uv_filt = double((Ui.^2+Vi.^2) <= 1);
199 uv_filt(uv_filt == 0) = nan;       % filter for removing illegal angles
200
201 % converts entries to NaN where U^2+V^2 < 1 is not fullfilled
202 Uc = Ui.*uv_filt;
203 Vc = Vi.*uv_filt;
204 Wc = real(sqrt(ones(size(Uc))-Uc.^2-Vc.^2));
205
206 % vectorize non-NaN entries
207 nanindc = find(~isnan(Uc));
208 Ucv = Uc(nanindc);
209 Vcv = Vc(nanindc);
210 Wcv = Wc(nanindc);
211
212 %% Scanning angle ......................................................
213
214 theta0 = (0/180)*pi;          % scanning angle in theta-dimension
215 phi0 = (0/180)*pi;            % scanning angle in phi-dimension
216
217 u0 = sin(theta0)*cos(phi0);    % scanning angles converted to u-space
218 v0 = sin(theta0)*sin(phi0);    % scanning angles converted to v-space
219
220 betaX = -k0*u0;
221 betaY = -k0*v0;
222
223 %% _____________________ Section 3: Gain Calculation _____________________
224 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
225 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
226 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
227 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
228 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
229 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
230 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
231 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
232 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
233 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
234
235 %% Subarray Gain ......................................................
236 % Here the radiation pattern of each element is calculated, This is
237 % then sorted into subarrays. From the subsequent matrices the total
238 % radiation pattern can be exracted before or after a certain
239 % calibration is implemented on the subarray elements or the entire
240 % array itself.
241
242 EPv = sqrt(Wv);  % Amplitude pattern for vectorized visualisation angles
243 EPcv = sqrt(Wcv);% Amplitude pattern for vectorized calibration angles
244
```

```
245 excv = Iv.*exp(1i*(X0v*betaX+Y0v*betaY)); % Ideal element excitation
246
247 vsize = [NumSub,numel(Uv),Msub*Nsub];   % size of visualisation matrices
248 cvsize = [NumSub,numel(Ucv),Msub*Nsub]; % size of calibration matrices
249
250 Gopt_subv = zeros(vsize);   % optimal array pattern for visualisation
251 Gopt_subcv = zeros(cvsize); % optimal array pattern for calibration
252
253 Gerr_subv = zeros(vsize);   % erronous array pattern for visualisation
254 Gerr_subcv = zeros(cvsize); % erronous array pattern for calibration
255
256
257 for i=1:NumSub
258     for j=1:Msub*Nsub
259
260     k = (i-1)*Msub*Nsub+j;
261
262     Gopt_subv(i,:,j) = EPv.*(excv(k) * ...
263                         exp(1i*k0*(X0v(k)*Uv+Y0v(k)*Vv+Z0v(k)*Wv)));
264
265     Gopt_subcv(i,:,j) = EPcv.*(excv(k) * ...
266                          exp(1i*k0*(X0v(k)*Ucv+Y0v(k)*Vcv+Z0v(k)*Wcv)));
267
268     Gerr_subv(i,:,j) = EPv.*(excv(k) * ...
269                         exp(1i*k0*(Xv(k)*Uv+Yv(k)*Vv+Zv(k)*Wv)));
270
271     Gerr_subcv(i,:,j) = EPcv.*(excv(k) * ...
272                          exp(1i*k0*(Xv(k)*Ucv+Yv(k)*Vcv+Zv(k)*Wcv)));
273
274     end
275 end
276
277 % Optimal pattern for full array
278 GOPT = zeros(size(U))*nan;
279 GOPT(nanind) = sum(sum(Gopt_subv,3),1);
280 GOPT_dB = 20*log10(abs(GOPT));
281 GOPT_normdB = GOPT_dB - max(max(GOPT_dB));
282
283 % Erronous pattern for full array
284 GERR = zeros(size(U))*nan;
285 GERR(nanind) = sum(sum(Gerr_subv,3),1);
286 GERR_dB = 20*log10(abs(GERR));
287 GERR_normdB = GERR_dB - max(max(GERR_dB));
288
289 % Patterns if subarrays are treated like an array of antennas.
290 % Here, the subarrays are not calibrated at all.
291 Gerr_v = sum(Gerr_subv,3);
292 Gerr_cv = sum(Gerr_subcv,3);
293 Gopt_v = sum(Gopt_subv,3);
294 Gopt_cv = sum(Gopt_subcv,3);
295
296 % indeces for main lobe in final pattern
297 [row,col] = find(GOPT == max(max(GOPT)));
298
299 %% ____________________ Section 4: Subarray Calibration ____________________
300 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
301 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
302 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
303 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
304 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
305 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
306 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
307 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
308 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
309 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
310
311 % Here, the individual subarrays are calibrated. It is assumed that only
312 % excitation changes of the elements are possible. Therefore, only
313 % diagonal entries are allowed in the calibration matrices.
314
```

```
315 %% Subarray Correction - Global Calibration ..........................
316
317 qglo = ones(NumSub,Msub*Nsub);
318 Gglo_subv = zeros(vsize);
319 Gglo_subcv = zeros(cvsize);
320
321 for i=1:NumSub
322
323 for j=1:Msub*Nsub
324     qglo(i,j) = (Gopt_subcv(i,:,j)*Gerr_subcv(i,:,j)')' / ...
325                 (Gopt_subcv(i,:,j)*Gopt_subcv(i,:,j)');
326 end
327
328 temp = repmat(conj(qglo(i,:)),numel(Uv),1);
329 Gglo_subv(i,:,:) = reshape(temp,[1,size(temp)]).*Gerr_subv(i,:,:);
330 temp = repmat(conj(qglo(i,:)),numel(Ucv),1);
331 Gglo_subcv(i,:,:) = reshape(temp,[1,size(temp)]).*Gerr_subcv(i,:,:);
332
333 end
334
335 %% Subarray Correction - Local Calibration ..........................
336
337 h = 50;
338 D = sqrt(abs(Ucv-u0).^2+abs(Vcv-v0).^2);
339 weight = exp(-h*D.^2);
340 weight = weight';
341
342 qloc = ones(NumSub,Msub*Nsub);
343 Gloc_subv = zeros(vsize);
344 Gloc_subcv = zeros(cvsize);
345
346 for i=1:NumSub
347
348 for j=1:Msub*Nsub
349
350 qloc(i,j)=sum(conj(Gopt_subcv(i,:,j)).*weight.*Gerr_subcv(i,:,j))./ ...
351           sum(conj(Gopt_subcv(i,:,j)).*weight.*Gopt_subcv(i,:,j));
352
353 end
354
355 temp = repmat(conj(qloc(i,:)),numel(Uv),1);
356 Gloc_subv(i,:,:) = reshape(temp,[1,size(temp)]).*Gerr_subv(i,:,:);
357 temp = repmat(conj(qloc(i,:)),numel(Ucv),1);
358 Gloc_subcv(i,:,:) = reshape(temp,[1,size(temp)]).*Gerr_subcv(i,:,:);
359
360 end
361
362 %% Global + Nothing Array Gain ..........................................
363
364 Ggn_v = sum(Gglo_subv,3);
365 Ggn_cv = sum(Gglo_subcv,3);
366
367 GGN = zeros(size(U))*nan;
368 GGN(nanind) = sum(Ggn_v,1);
369 GGN_dB = 20*log10(abs(GGN));
370 GGN_normdB = GGN_dB - max(max(GGN_dB));
371
372 %% Local + Nothing Array Gain ..........................................
373
374 Gln_v = sum(Gloc_subv,3);
375 Gln_cv = sum(Gloc_subcv,3);
376
377 GLN = zeros(size(U))*nan;
378 GLN(nanind) = sum(Gln_v,1);
379 GLN_dB = 20*log10(abs(GLN));
380 GLN_normdB = GLN_dB - max(max(GLN_dB));
381
382 %% Subarray patterns comparison ........................................
383
384 Gopt_sub = zeros([size(U),NumSub]);
```

```
385 Gopt_sub_dB = zeros([size(U),NumSub]);
386 Gopt_sub_normdB = zeros([size(U),NumSub]);
387
388 Gerr_sub = zeros([size(U),NumSub]);
389 Gerr_sub_dB = zeros([size(U),NumSub]);
390 Gerr_sub_normdB = zeros([size(U),NumSub]);
391
392 Gglo_sub = zeros([size(U),NumSub]);
393 Gglo_sub_dB = zeros([size(U),NumSub]);
394 Gglo_sub_normdB = zeros([size(U),NumSub]);
395
396 Gloc_sub = zeros([size(U),NumSub]);
397 Gloc_sub_dB = zeros([size(U),NumSub]);
398 Gloc_sub_normdB = zeros([size(U),NumSub]);
399
400 for i=1:NumSub
401
402 temp = zeros(size(U))*nan;
403 temp(nanind) = Gopt_v(i,:,:);
404 Gopt_sub(:,:,i) = temp;
405
406 Gopt_sub_dB(:,:,i) = 20*log10(abs(Gopt_sub(:,:,i)));
407 Gopt_sub_normdB(:,:,i)=Gopt_sub_dB(:,:,i)-max(max(Gopt_sub_dB(:,:,i)));
408
409 temp = zeros(size(U))*nan;
410 temp(nanind) = Gerr_v(i,:,:);
411 Gerr_sub(:,:,i) = temp;
412
413 Gerr_sub_dB(:,:,i) = 20*log10(abs(Gerr_sub(:,:,i)));
414 Gerr_sub_normdB(:,:,i)=Gerr_sub_dB(:,:,i)-max(max(Gerr_sub_dB(:,:,i)));
415
416 temp = zeros(size(U))*nan;
417 temp(nanind) = Ggn_v(i,:,:);
418 Gglo_sub(:,:,i) = temp;
419
420 Gglo_sub_dB(:,:,i) = 20*log10(abs(Gglo_sub(:,:,i)));
421 Gglo_sub_normdB(:,:,i)=Gglo_sub_dB(:,:,i)-max(max(Gglo_sub_dB(:,:,i)));
422
423 temp = zeros(size(U))*nan;
424 temp(nanind) = Gln_v(i,:,:);
425 Gloc_sub(:,:,i) = temp;
426
427 Gloc_sub_dB(:,:,i) = 20*log10(abs(Gloc_sub(:,:,i)));
428 Gloc_sub_normdB(:,:,i)=Gloc_sub_dB(:,:,i)-max(max(Gloc_sub_dB(:,:,i)));
429
430 end
431
432 %% _____________ Section 5: Inter Subarray Calibration _____________
433 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
434 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
435 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
436 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
437 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
438 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
439 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
440 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
441 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
442 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
443
444 % Here, the radiation patterns after the calibration between subarrays
445 % has been applied are calculated. For this section we are assuming that
446 % full digitalization between subarrays is available. Therefore, the
447 % calibration matrices are allowed to use all entries.
448
449 %% Global + Global Calibration ..........................................
450
451 Qgg = Ggn_v/Gopt_v;
452
453 GGG = zeros(size(U))*nan;
454 GGG(nanind) = sum(Qgg\Ggn_v,1);
```

```
455 GGG_dB = 20*log10(abs(GGG));
456 GGG_normdB = GGG_dB - max(max(GGG_dB));
457
458 %% Local + Global Calibration ..........................................
459
460 Qlg = Gln_v/Gopt_v;
461
462 GLG = zeros(size(U))*nan;
463 GLG(nanind) = sum(Qlg\Gln_v,1);
464 GLG_dB = 20*log10(abs(GLG));
465 GLG_normdB = GLG_dB - max(max(GLG_dB));
466
467 %% Local + Local Calibration ..........................................
468
469 h = 50;
470 D = sqrt(abs(Ucv-u0).^2+abs(Vcv-v0).^2);
471 weight = exp(-h*D.^2);
472 weight = weight';
473 qll = ones(1,NumSub);
474
475 for i=1:NumSub
476     qll(i) = sum(conj(Gopt_cv(i,:)).*weight.*Gln_cv(i,:)) ./ ...
477                 sum(conj(Gopt_cv(i,:)).*weight.*Gopt_cv(i,:));
478 end
479 Qll = diag(qll);
480
481 GLL = zeros(size(U))*nan;
482 GLL(nanind) = sum(Qll\Gln_v,1);
483 GLL_dB = 20*log10(abs(GLL));
484 GLL_normdB = GLL_dB - max(max(GLL_dB));
485
486 %% Global + Local Calibration ..........................................
487
488 h = 50;
489 D = sqrt(abs(Ucv-u0).^2+abs(Vcv-v0).^2);
490 weight = exp(-h*D.^2);
491 weight = weight';
492 qgl = ones(1,NumSub);
493
494 for i=1:NumSub
495     qgl(i) = sum(conj(Gopt_cv(i,:)).*weight.*Ggn_cv(i,:)) ./ ...
496                 sum(conj(Gopt_cv(i,:)).*weight.*Gopt_cv(i,:));
497 end
498 Qgl = diag(qgl);
499
500 GGL = zeros(size(U))*nan;
501 GGL(nanind) = sum(Qgl\Ggn_v,1);
502 GGL_dB = 20*log10(abs(GGL));
503 GGL_normdB = GGL_dB - max(max(GGL_dB));
504
505 %% Nothing + Global Calibration ..........................................
506
507 Qng = Gerr_v/Gopt_v;
508
509 GNG = zeros(size(U))*nan;
510 GNG(nanind) = sum(Qng\Gerr_v,1);
511 GNG_dB = 20*log10(abs(GNG));
512 GNG_normdB = GNG_dB - max(max(GNG_dB));
513
514 %% Nothing + Local Calibration ..........................................
515
516 h = 50;
517 D = sqrt(abs(Ucv-u0).^2+abs(Vcv-v0).^2);
518 weight = exp(-h*D.^2);
519 weight = weight';
520 qnl = ones(1,NumSub);
521
522 for i=1:NumSub
523     qnl(i) = sum(conj(Gopt_cv(i,:)).*weight.*Gerr_cv(i,:)) ./ ...
524                 sum(conj(Gopt_cv(i,:)).*weight.*Gopt_cv(i,:));
```

```
end
Qnl = diag(qnl);

GNL = zeros(size(U))*nan;
GNL(nanind) = sum(Qnl\Gerr_v,1);
GNL_dB = 20*log10(abs(GNL));
GNL_normdB = GNL_dB - max(max(GNL_dB));

%%
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
toc
```

A.2.2 Hybrid Array

```
%% Hybrid analog/digital Planar Array Calibration Model
% by Josef Ydreborg
%% Constants ______________________________________________________________
tic
f0 = 10e9;                          % Central frequency
c0 = 299792458;                     % Speed of light in vacuum
lam0 = c0/f0;                       % Wavelength of central frequency
k0 = (2*pi)/lam0;                   % Wavenumber of central frequency

%% ____________________ Section 1: Array Information ____________________
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

%% Ideal Array Information .........................................
% Sensor Array is in the x-y plane and the direction of propagation is
% z-axis. As long as the number of elements in the array and subarrays
% are powers of two, the code should work well. Subarray size of one
% element works but keep in mind that calibrations based on subarrays
% are invalid in this case.

M = 2^5;                    % Number of elements along x-axis
N = 2^5;                    % Number of elements along y-axis
m = -(M-1)/2:1:(M-1)/2;     % Element indeces along x-axis
n = -(N-1)/2:1:(N-1)/2;     % Element indeces along y-axis

NumSubX = 2^3;              % Number of Subarrays along x-axis
NumSubY = 2^3;              % Number of Subarrays along y-axis
NumSub = NumSubX*NumSubY;

Msub = M/NumSubX;           % Number of elements in subarray along x-axis
Nsub = N/NumSubY;           % Number of elements in subarray along y-axis

dx = lam0/2;                % Ideal element spacing along x-axis
dy = lam0/2;                % Ideal element spacing along y-axis
Lx = M*dx;                  % Length of array along x-axis
Ly = N*dy;                  % Length of array along y-axis

x = m*dx;                   % Ideal element positions along x-axis
```

```
47  y = n*dy;                    % Ideal element positions along y-axis
48  z = zeros(size(x));
49
50  [X0,Y0] = meshgrid(x,y);
51  Z0 = zeros(size(X0));
52
53  %% Position Errors ..................................................
54
55  X = X0;
56  Y = Y0;
57  Z = Z0;
58
59  mu = 0;                      % mean value of position errors
60  sigx = 0.01*lam0;            % standard deviation of elements in x-axis
61  sigy = 0.01*lam0;            % standard deviation of elements in y-axis
62  sigz = 0.01*lam0;            % standard deviation of elements in z-axis
63  subx = 0.01*lam0;            % standard deviation of subarrays in x-axis
64  suby = 0.01*lam0;            % standard deviation of subarrays in y-axis
65  subz = 0.01*lam0;            % standard deviation of subarrays in z-axis
66
67  subX = normrnd(mu,subx,N/Nsub,M/Msub);
68  subY = normrnd(mu,suby,N/Nsub,M/Msub);
69  subZ = normrnd(mu,subz,N/Nsub,M/Msub);
70  subX = kron(subX,ones(Nsub,Msub));      % x-axis errors of subarrays
71  subY = kron(subY,ones(Nsub,Msub));      % y-axis errors of subarrays
72  subZ = kron(subZ,ones(Nsub,Msub));      % z-axis errors of subarrays
73
74  sigX = normrnd(mu,sigx,N,M);            % x-axis errors of elements
75  sigY = normrnd(mu,sigy,N,M);            % y-axis errors of elements
76  sigZ = normrnd(mu,sigz,N,M);            % z-axis errors of elements
77
78  X = X + sigX + subX;     % erroneous x-position of each element
79  Y = Y + sigY + subY;     % erroneous y-position of each element
80  Z = Z + sigZ + subZ;     % erroneous z-position of each element
81
82  %% Weighting Scheme ..................................................
83  % This code uses a taylor taper weighting scheme.
84
85  R = -40;                     % desired sidelobe level [dB]
86  r = 10^(-R/20);              % desired sidelobe ratio [linear]
87
88  A = (1/pi)*acosh(r);
89  nbar = ceil(2*A^2+0.5);      % SLL=-30dB; 4<nbar<7 for monotonic taper
90                               % SLL=-40dB; 7<nbar<11 for monotonic taper
91  dn = 1:1:nbar-1;
92  sigma = nbar/(A^2+(nbar-1/2)^2)^(1/2);
93  zn = sigma*(A^2+(dn-1/2).^2).^(1/2);
94  p = ones(1,nbar-1);
95  for i=1:1:nbar-1
96      p = p .* (1-dn.^2./zn(i).^2);
97  end
98  Fm = ((factorial(nbar-1)^2)./(factorial(nbar-1+dn).* ...
99       factorial(nbar-1-dn))).*p;
100
101 wx = ones(1,M); sx = zeros(1,M);
102 wy = ones(1,N); sy = zeros(1,N);
103 for i=1:1:nbar-1
104     sx = sx + Fm(i).*cos((2*pi*i*x)/Lx);
105     sy = sy + Fm(i).*cos((2*pi*i*y)/Ly);
106 end
107 wx = wx + 2*sx;
108 wy = wy + 2*sy;
109 [IX,IY] = meshgrid(wx,wy);
110 I = IX.*IY;                  % tapering values in xy-plane
111
112 %% Vectorizing positions and tapering ................................
113 % Puts all element positions in vectors with subarray positions lumped
114 % together.
115
116 X0v = zeros(1,M*N);
```

```
Y0v = zeros(1,M*N);
Z0v = zeros(1,M*N);
Xv = zeros(1,M*N);
Yv = zeros(1,M*N);
Zv = zeros(1,M*N);
Iv = zeros(1,M*N);

count = 0;
for i=1:NumSubX
    for j=1:NumSubY

    temp = X0((i-1)*Msub+1:(i-1)*Msub+Msub,(j-1)*Nsub+1:(j-1)*Nsub+Nsub);
    temp = temp(:); X0v(count*Msub*Nsub+1:(count+1)*Msub*Nsub) = temp;

    temp = Y0((i-1)*Msub+1:(i-1)*Msub+Msub,(j-1)*Nsub+1:(j-1)*Nsub+Nsub);
    temp = temp(:); Y0v(count*Msub*Nsub+1:(count+1)*Msub*Nsub) = temp;

    temp = Z0((i-1)*Msub+1:(i-1)*Msub+Msub,(j-1)*Nsub+1:(j-1)*Nsub+Nsub);
    temp = temp(:); Z0v(count*Msub*Nsub+1:(count+1)*Msub*Nsub) = temp;

    temp = X((i-1)*Msub+1:(i-1)*Msub+Msub,(j-1)*Nsub+1:(j-1)*Nsub+Nsub);
    temp = temp(:); Xv(count*Msub*Nsub+1:(count+1)*Msub*Nsub) = temp;

    temp = Y((i-1)*Msub+1:(i-1)*Msub+Msub,(j-1)*Nsub+1:(j-1)*Nsub+Nsub);
    temp = temp(:); Yv(count*Msub*Nsub+1:(count+1)*Msub*Nsub) = temp;

    temp = Z((i-1)*Msub+1:(i-1)*Msub+Msub,(j-1)*Nsub+1:(j-1)*Nsub+Nsub);
    temp = temp(:); Zv(count*Msub*Nsub+1:(count+1)*Msub*Nsub) = temp;

    temp = I((i-1)*Msub+1:(i-1)*Msub+Msub,(j-1)*Nsub+1:(j-1)*Nsub+Nsub);
    temp = temp(:); Iv(count*Msub*Nsub+1:(count+1)*Msub*Nsub) = temp;

    count = count + 1;

    end
end

%% ____________________ Section 2: Angle Information ____________________
% ..........................................................................
% ..........................................................................
% ..........................................................................
% ..........................................................................
% ..........................................................................
% ..........................................................................
% ..........................................................................
% ..........................................................................
% ..........................................................................
% ..........................................................................

% This piece of code defines what angles will be used for displaying the
% data and which angles are used for calibration. Additionally, the
% scanning angle is defined.

%% Visualisation angles ..................................................

resp = 250;                         % resolution of angles = (2*res+1)^2
u = -1:1/resp:1;                    % angles used in u-space
v = -1:1/resp:1;                    % angles used in v-space

[Ui,Vi] = meshgrid(u,v);            % 2D mesh of U,V angles
uv_filt = double((Ui.^2+Vi.^2) <= 1);
uv_filt(uv_filt == 0) = nan;        % filter for removing illegal angles

% converts entries to NaN where U^2+V^2 < 1 is not fullfilled
U = Ui.*uv_filt;
V = Vi.*uv_filt;
W = abs(sqrt(ones(size(U))-U.^2-V.^2));

% vectorize non-NaN entries
nanind = find(~isnan(U));
```

```
187 Uv = U(nanind);
188 Vv = V(nanind);
189 Wv = W(nanind);
190
191 %% Calibration angles ..................................................
192
193 resc = 100;                          % resolution of angles = (2*res+1)^2
194 uc = -1:1/resc:1;                    % angles used in u-space
195 vc = -1:1/resc:1;                    % angles used in v-space
196
197 [Ui,Vi] = meshgrid(uc,vc);           % 2D mesh of U,V angles
198 uv_filt = double((Ui.^2+Vi.^2) <= 1);
199 uv_filt(uv_filt == 0) = nan;         % filter for removing illegal angles
200
201 % converts entries to NaN where U^2+V^2 < 1 is not fullfilled
202 Uc = Ui.*uv_filt;
203 Vc = Vi.*uv_filt;
204 Wc = real(sqrt(ones(size(Uc))-Uc.^2-Vc.^2));
205
206 % vectorize non-NaN entries
207 nanindc = find(~isnan(Uc));
208 Ucv = Uc(nanindc);
209 Vcv = Vc(nanindc);
210 Wcv = Wc(nanindc);
211
212 %% Scanning angle ......................................................
213
214 theta0 = (0/180)*pi;           % scanning angle in theta-dimension
215 phi0 = (0/180)*pi;             % scanning angle in phi-dimension
216
217 u0 = sin(theta0)*cos(phi0);    % scanning angles converted to u-space
218 v0 = sin(theta0)*sin(phi0);    % scanning angles converted to v-space
219
220 betaX = -k0*u0;
221 betaY = -k0*v0;
222
223 %% _____________________ Section 3: Gain Calculation _____________________
224 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
225 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
226 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
227 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
228 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
229 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
230 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
231 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
232 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
233 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
234
235 %% Subarray Gain ......................................................
236 % Here the radiation pattern of each element is calculated, This is
237 % then sorted into subarrays. From the subsequent matrices the total
238 % radiation pattern can be exracted before or after a certain
239 % calibration is implemented on the subarray elements or the entire
240 % array itself.
241
242 EPv = sqrt(Wv);  % Amplitude pattern for vectorized visualisation angles
243 EPcv = sqrt(Wcv);% Amplitude pattern for vectorized calibration angles
244
245 excv = Iv.*exp(1i*(X0v*betaX+Y0v*betaY)); % Ideal element excitation
246
247 vsize = [NumSub,numel(Uv),Msub*Nsub];   % size of visualisation matrices
248 cvsize = [NumSub,numel(Ucv),Msub*Nsub]; % size of calibration matrices
249
250 Gopt_subv = zeros(vsize);   % optimal array pattern for visualisation
251 Gopt_subcv = zeros(cvsize); % optimal array pattern for calibration
252
253 Gerr_subv = zeros(vsize);   % erronous array pattern for visualisation
254 Gerr_subcv = zeros(cvsize); % erronous array pattern for calibration
255
256
```

```
257 for i=1:NumSub
258     for j=1:Msub*Nsub
259
260     k = (i-1)*Msub*Nsub+j;
261
262     Gopt_subv(i,:,j) = EPv.*(excv(k) * ...
263                         exp(1i*k0*(X0v(k)*Uv+Y0v(k)*Vv+Z0v(k)*Wv)));
264
265     Gopt_subcv(i,:,j) = EPcv.*(excv(k) * ...
266                          exp(1i*k0*(X0v(k)*Ucv+Y0v(k)*Vcv+Z0v(k)*Wcv)));
267
268     Gerr_subv(i,:,j) = EPv.*(excv(k) * ...
269                         exp(1i*k0*(Xv(k)*Uv+Yv(k)*Vv+Zv(k)*Wv)));
270
271     Gerr_subcv(i,:,j) = EPcv.*(excv(k) * ...
272                          exp(1i*k0*(Xv(k)*Ucv+Yv(k)*Vcv+Zv(k)*Wcv)));
273
274     end
275 end
276
277 % Optimal pattern for full array
278 GOPT = zeros(size(U))*nan;
279 GOPT(nanind) = sum(sum(Gopt_subv,3),1);
280 GOPT_dB = 20*log10(abs(GOPT));
281 GOPT_normdB = GOPT_dB - max(max(GOPT_dB));
282
283 % Erronous pattern for full array
284 GERR = zeros(size(U))*nan;
285 GERR(nanind) = sum(sum(Gerr_subv,3),1);
286 GERR_dB = 20*log10(abs(GERR));
287 GERR_normdB = GERR_dB - max(max(GERR_dB));
288
289 % Patterns if subarrays are treated like an array of antennas.
290 % Here, the subarrays are not calibrated at all.
291 Gerr_v = sum(Gerr_subv,3);
292 Gerr_cv = sum(Gerr_subcv,3);
293 Gopt_v = sum(Gopt_subv,3);
294 Gopt_cv = sum(Gopt_subcv,3);
295
296 % indeces for main lobe in final pattern
297 [row,col] = find(GOPT == max(max(GOPT)));
298
299 %% _____________________ Section 4: Subarray Calibration _____________________
300 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
301 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
302 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
303 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
304 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
305 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
306 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
307 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
308 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
309 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
310
311 % Here, the individual subarrays are calibrated. It is assumed that only
312 % excitation changes of the elements are possible. Therefore, only
313 % diagonal entries are allowed in the calibration matrices.
314
315 %% Subarray Correction - Global Calibration .............................
316
317 qglo = ones(NumSub,Msub*Nsub);
318 Gglo_subv = zeros(vsize);
319 Gglo_subcv = zeros(cvsize);
320
321 for i=1:NumSub
322
323 for j=1:Msub*Nsub
324     qglo(i,j) = (Gopt_subcv(i,:,j)*Gerr_subcv(i,:,j)')' / ...
325                 (Gopt_subcv(i,:,j)*Gopt_subcv(i,:,j)');
326 end
```

```
327
328 temp = repmat(conj(qglo(i,:)),numel(Uv),1);
329 Gglo_subv(i,:,:) = reshape(temp,[1,size(temp)]).*Gerr_subv(i,:,:);
330 temp = repmat(conj(qglo(i,:)),numel(Ucv),1);
331 Gglo_subcv(i,:,:) = reshape(temp,[1,size(temp)]).*Gerr_subcv(i,:,:);
332
333 end
334
335 %% Subarray Correction - Local Calibration ..............................
336
337 h = 50;
338 D = sqrt(abs(Ucv-u0).^2+abs(Vcv-v0).^2);
339 weight = exp(-h*D.^2);
340 weight = weight';
341
342 qloc = ones(NumSub,Msub*Nsub);
343 Gloc_subv = zeros(vsize);
344 Gloc_subcv = zeros(cvsize);
345
346 for i=1:NumSub
347
348 for j=1:Msub*Nsub
349
350 qloc(i,j)=sum(conj(Gopt_subcv(i,:,j)).*weight.*Gerr_subcv(i,:,j))./ ...
351           sum(conj(Gopt_subcv(i,:,j)).*weight.*Gopt_subcv(i,:,j));
352
353 end
354
355 temp = repmat(conj(qloc(i,:)),numel(Uv),1);
356 Gloc_subv(i,:,:) = reshape(temp,[1,size(temp)]).*Gerr_subv(i,:,:);
357 temp = repmat(conj(qloc(i,:)),numel(Ucv),1);
358 Gloc_subcv(i,:,:) = reshape(temp,[1,size(temp)]).*Gerr_subcv(i,:,:);
359
360 end
361
362 %% Global + Nothing Array Gain ..........................................
363
364 Ggn_v = sum(Gglo_subv,3);
365 Ggn_cv = sum(Gglo_subcv,3);
366
367 GGN = zeros(size(U))*nan;
368 GGN(nanind) = sum(Ggn_v,1);
369 GGN_dB = 20*log10(abs(GGN));
370 GGN_normdB = GGN_dB - max(max(GGN_dB));
371
372 %% Local + Nothing Array Gain ..........................................
373
374 Gln_v = sum(Gloc_subv,3);
375 Gln_cv = sum(Gloc_subcv,3);
376
377 GLN = zeros(size(U))*nan;
378 GLN(nanind) = sum(Gln_v,1);
379 GLN_dB = 20*log10(abs(GLN));
380 GLN_normdB = GLN_dB - max(max(GLN_dB));
381
382 %% Subarray patterns comparison .........................................
383
384 Gopt_sub = zeros([size(U),NumSub]);
385 Gopt_sub_dB = zeros([size(U),NumSub]);
386 Gopt_sub_normdB = zeros([size(U),NumSub]);
387
388 Gerr_sub = zeros([size(U),NumSub]);
389 Gerr_sub_dB = zeros([size(U),NumSub]);
390 Gerr_sub_normdB = zeros([size(U),NumSub]);
391
392 Gglo_sub = zeros([size(U),NumSub]);
393 Gglo_sub_dB = zeros([size(U),NumSub]);
394 Gglo_sub_normdB = zeros([size(U),NumSub]);
395
396 Gloc_sub = zeros([size(U),NumSub]);
```

```
Gloc_sub_dB = zeros([size(U),NumSub]);
Gloc_sub_normdB = zeros([size(U),NumSub]);

for i=1:NumSub

temp = zeros(size(U))*nan;
temp(nanind) = Gopt_v(i,:,:);
Gopt_sub(:,:,i) = temp;

Gopt_sub_dB(:,:,i) = 20*log10(abs(Gopt_sub(:,:,i)));
Gopt_sub_normdB(:,:,i)=Gopt_sub_dB(:,:,i)-max(max(Gopt_sub_dB(:,:,i)));

temp = zeros(size(U))*nan;
temp(nanind) = Gerr_v(i,:,:);
Gerr_sub(:,:,i) = temp;

Gerr_sub_dB(:,:,i) = 20*log10(abs(Gerr_sub(:,:,i)));
Gerr_sub_normdB(:,:,i)=Gerr_sub_dB(:,:,i)-max(max(Gerr_sub_dB(:,:,i)));

temp = zeros(size(U))*nan;
temp(nanind) = Ggn_v(i,:,:);
Gglo_sub(:,:,i) = temp;

Gglo_sub_dB(:,:,i) = 20*log10(abs(Gglo_sub(:,:,i)));
Gglo_sub_normdB(:,:,i)=Gglo_sub_dB(:,:,i)-max(max(Gglo_sub_dB(:,:,i)));

temp = zeros(size(U))*nan;
temp(nanind) = Gln_v(i,:,:);
Gloc_sub(:,:,i) = temp;

Gloc_sub_dB(:,:,i) = 20*log10(abs(Gloc_sub(:,:,i)));
Gloc_sub_normdB(:,:,i)=Gloc_sub_dB(:,:,i)-max(max(Gloc_sub_dB(:,:,i)));

end

%% ________________ Section 5: Inter Subarray Calibration _________________
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::

% Here, the radiation patterns after the calibration between subarrays
% has been applied are calculated. For this section we are assuming that
% full digitalization between subarrays is available. Therefore, the
% calibration matrices are allowed to use all entries.

%% Global + Global Calibration .......................................

Qgg = Ggn_v/Gopt_v;

GGG = zeros(size(U))*nan;
GGG(nanind) = sum(Qgg\Ggn_v,1);
GGG_dB = 20*log10(abs(GGG));
GGG_normdB = GGG_dB - max(max(GGG_dB));

%% Local + Global Calibration ........................................

Qlg = Gln_v/Gopt_v;

GLG = zeros(size(U))*nan;
GLG(nanind) = sum(Qlg\Gln_v,1);
GLG_dB = 20*log10(abs(GLG));
GLG_normdB = GLG_dB - max(max(GLG_dB));

```

```
%% Local + Local Calibration ..........................................

h = 50;
D = sqrt(abs(Ucv-u0).^2+abs(Vcv-v0).^2);
weight = exp(-h*D.^2);
weight = weight';
qll = ones(1,NumSub);

for i=1:NumSub
    qll(i) = sum(conj(Gopt_cv(i,:)).*weight.*Gln_cv(i,:)) ./ ...
                sum(conj(Gopt_cv(i,:)).*weight.*Gopt_cv(i,:));
end
Qll = diag(qll);

GLL = zeros(size(U))*nan;
GLL(nanind) = sum(Qll\Gln_v,1);
GLL_dB = 20*log10(abs(GLL));
GLL_normdB = GLL_dB - max(max(GLL_dB));

%% Global + Local Calibration ..........................................

h = 50;
D = sqrt(abs(Ucv-u0).^2+abs(Vcv-v0).^2);
weight = exp(-h*D.^2);
weight = weight';
qgl = ones(1,NumSub);

for i=1:NumSub
    qgl(i) = sum(conj(Gopt_cv(i,:)).*weight.*Ggn_cv(i,:)) ./ ...
                sum(conj(Gopt_cv(i,:)).*weight.*Gopt_cv(i,:));
end
Qgl = diag(qgl);

GGL = zeros(size(U))*nan;
GGL(nanind) = sum(Qgl\Ggn_v,1);
GGL_dB = 20*log10(abs(GGL));
GGL_normdB = GGL_dB - max(max(GGL_dB));

%% Nothing + Global Calibration ..........................................

Qng = Gerr_v/Gopt_v;

GNG = zeros(size(U))*nan;
GNG(nanind) = sum(Qng\Gerr_v,1);
GNG_dB = 20*log10(abs(GNG));
GNG_normdB = GNG_dB - max(max(GNG_dB));

%% Nothing + Local Calibration ..........................................

h = 50;
D = sqrt(abs(Ucv-u0).^2+abs(Vcv-v0).^2);
weight = exp(-h*D.^2);
weight = weight';
qnl = ones(1,NumSub);

for i=1:NumSub
    qnl(i) = sum(conj(Gopt_cv(i,:)).*weight.*Gerr_cv(i,:)) ./ ...
                sum(conj(Gopt_cv(i,:)).*weight.*Gopt_cv(i,:));
end
Qnl = diag(qnl);

GNL = zeros(size(U))*nan;
GNL(nanind) = sum(Qnl\Gerr_v,1);
GNL_dB = 20*log10(abs(GNL));
GNL_normdB = GNL_dB - max(max(GNL_dB));

%%
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
% ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
```

```
537 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
538 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
539 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
540 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
541 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
542 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
543 % ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
544 toc
```

B

CATR Measurement Setup

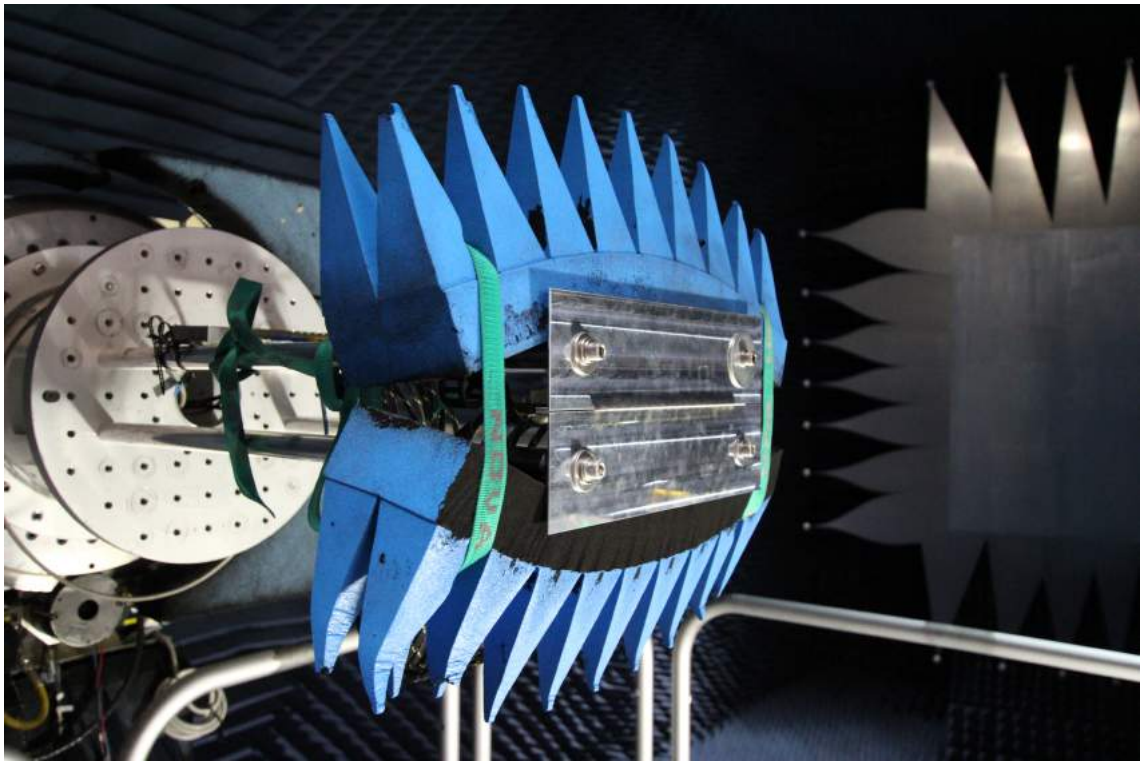


Figure B.1: Frontside of CATR measurement setup.

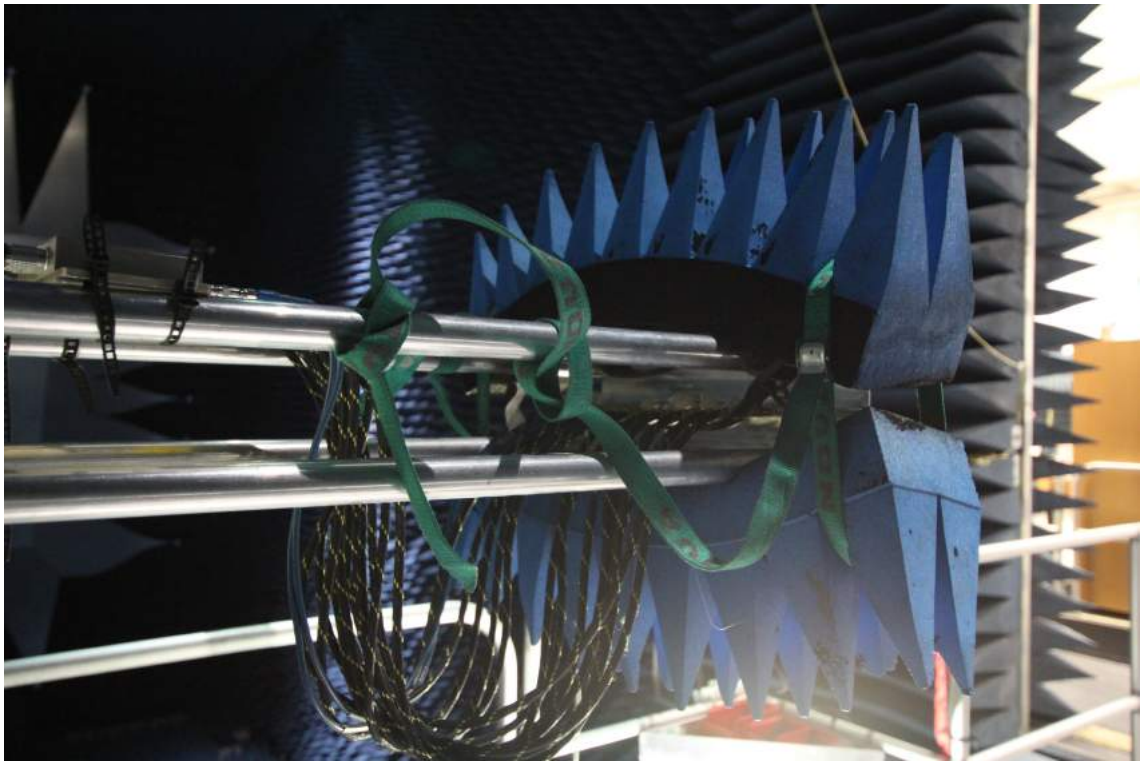


Figure B.2: Backside of CATR measurement setup.

Figure B.3: Overview of CATR measurement setup.